Final System And System Report

**Team: Access-Ability Innovators**

El Shammas, Lama Rita

Li, Jiaying

Truong, Tam K

Srinivasan, Siddharth

**CS-6795: Introduction to Cognitive Science**

**Dr. Michael Helms**

**Pseudo-code**

1. **The complexity of the key decision task you select (non-trivial), 5 points**

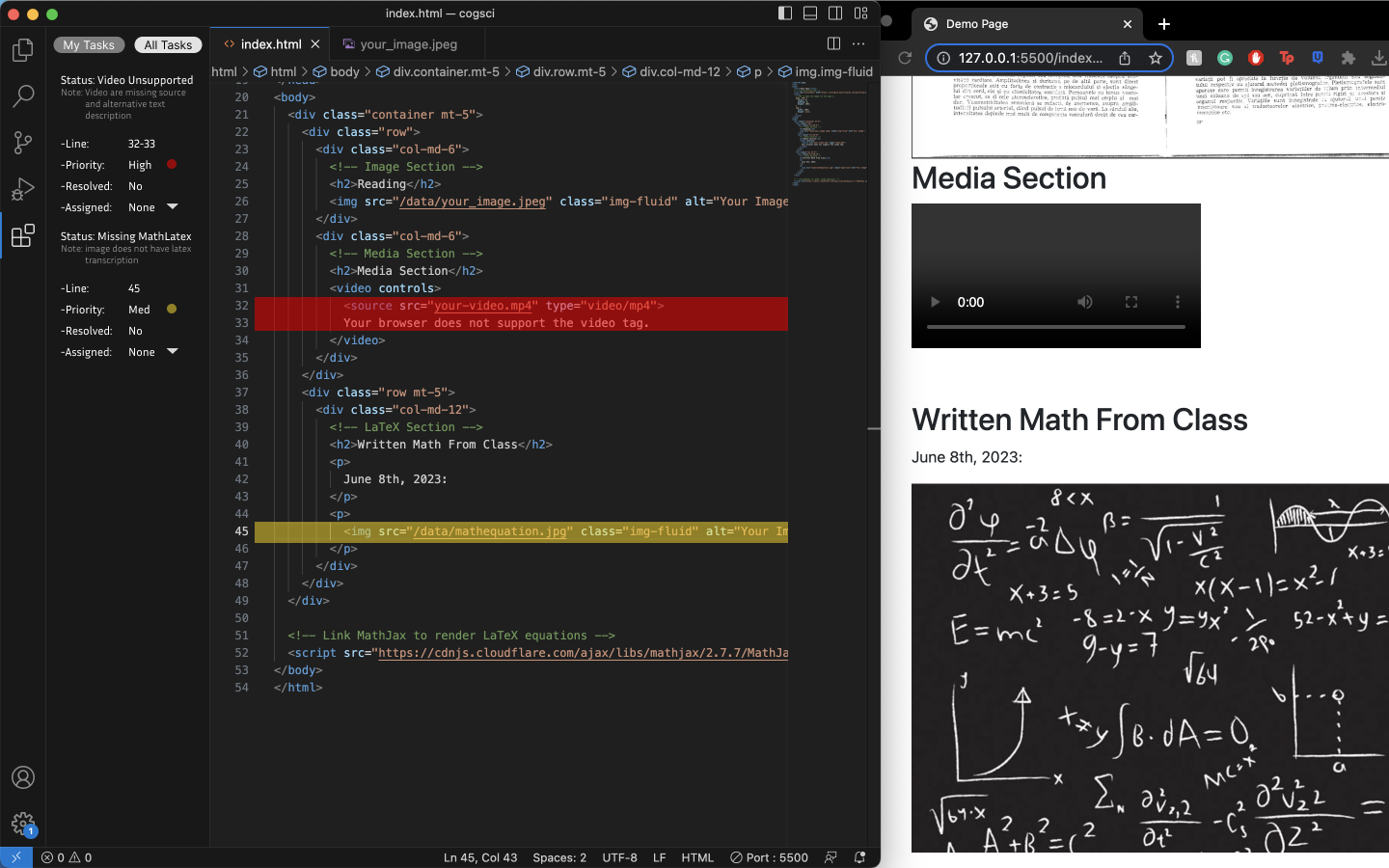
Typically, the designers must go through every image, pdf file, and audio file on the canvas pages to ensure students can view them with their assistive technologies. This process can take very long as instructors often upload new materials to their classrooms, which requires the designers to stay on top of their tasks. The designers also work with other people who specialize in other aspects of designing, so delegating tasks to each other is necessary to ensure the studying materials are accessible to visually impaired students.

Supported by our interviews and findings, we chose the above mentioned tasks as our crucial decision because it is a familiar problem designers face when designing accessible materials. Our application is tailored toward the designer, suggesting to them where the problems might occur and who is the best person on the team to delegate the problem.

1. **The sophistication of the pseudo-code explanation of reasoning, with respect to the key decision, 15 points**

Our application highlights two essential features that help designers make critical decisions: Flagging and Assigning. Flagging refers to the application automatically looking for common errors and summarizing the problem to be fixed. Assigning is a feature that allows the designers to assign the task to the best-fit person on the team. Below is a pseudo-code and the prototype of how these features would work in the application.

1. Flagging:
   1. Prototype



* 1. Pseudo code
     1. Html is loaded
        1. Database are pulled
           1. Common errors:

Document

Document html

Media files

Video

Images

Audio

Error handling

missing description

files not supported

* + - * 1. Assistive Technologies

Screen reader

Speaking tools

Writing tools

Reading tools

Math tools

* + - 1. HTML files are checked
         1. Functionality
         2. Accessibility
      2. Common errors are checked against the database
         1. Errors and warning codes are flagged

UI generate a ticket

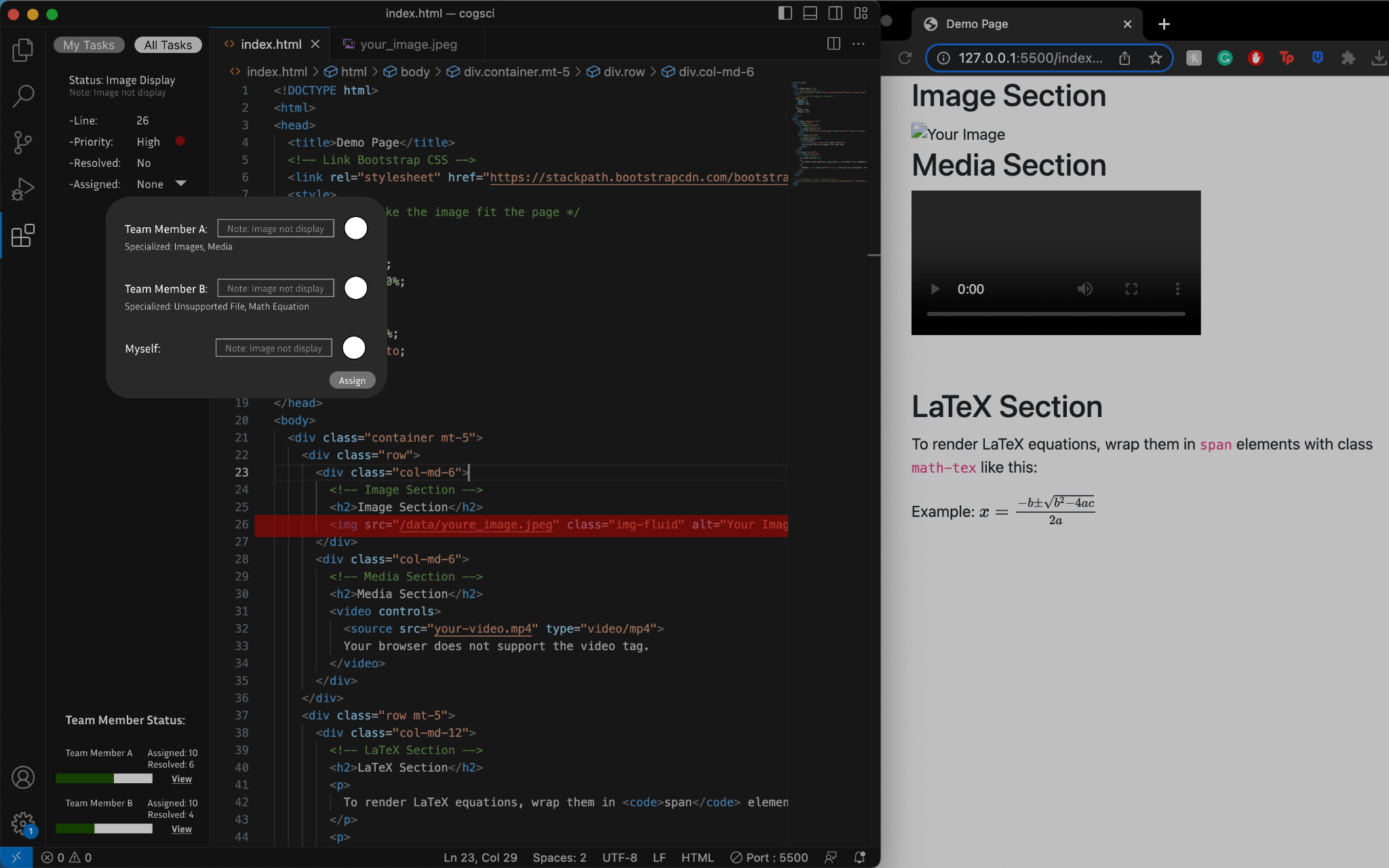
Type:

Line number

Priority

Brief description generated

1. Assigning
   1. Prototype



* 1. Pseudo Code
     1. “All Tasks” received new ticket(s)
        1. Assigned button clicked
        2. User Interface with the assigning feature pop-up
           1. Front end:

Problem description is passed through from ticket flagging

Team member status

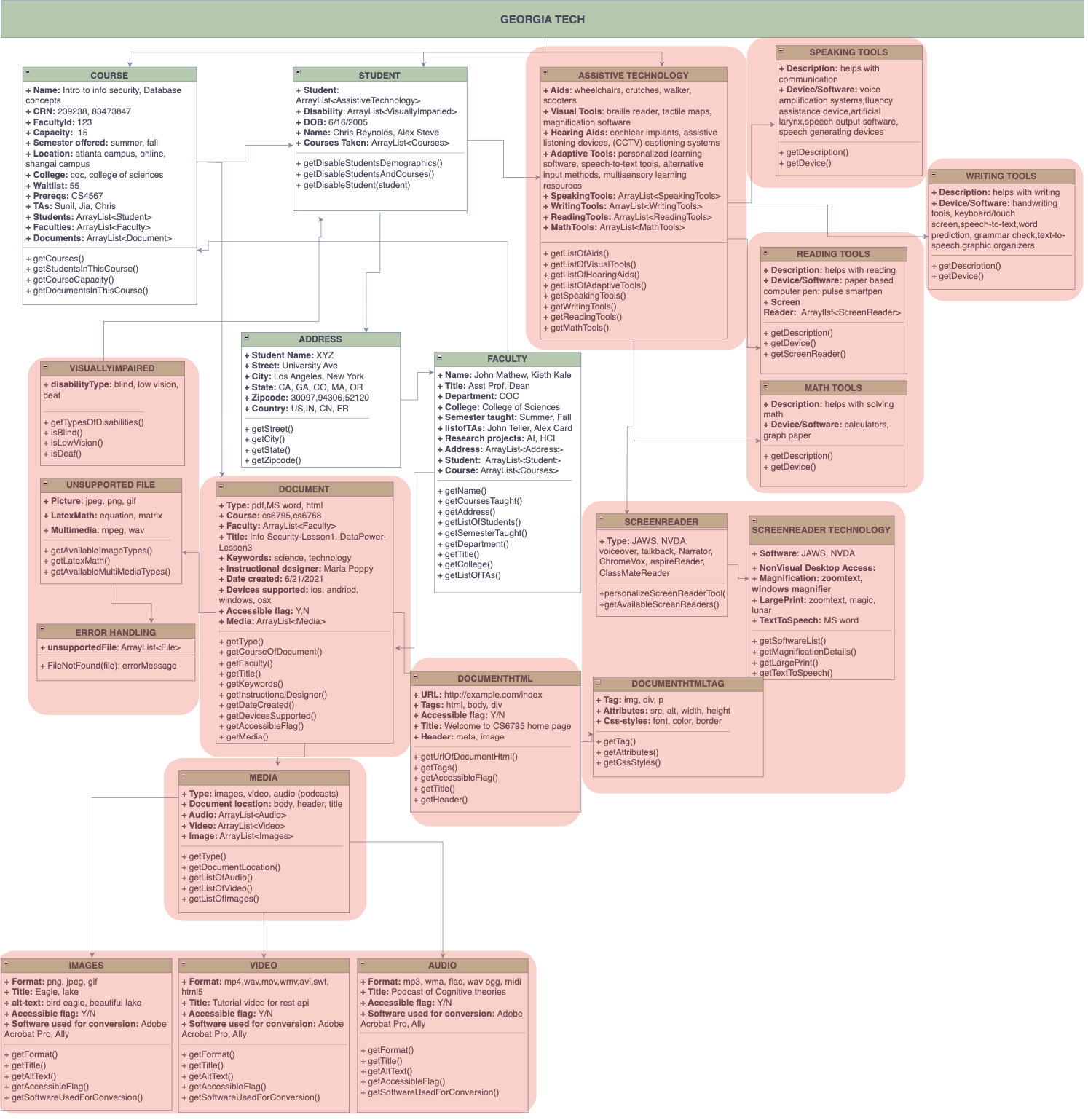
* + - * 1. Back end:

Sorting the team member based on the problem and how familiarized they are

* + - 1. Assigned button clicked
         1. The person status bar is updated
         2. Pushed to their “My Task” bar
         3. “View” to see changes
    1. “My taskbar received new ticket
       1. Resolving the issue
       2. Confirming the problem is solved
       3. History and status bar is updated

1. **The extent to which your pseudo-code leverages your conceptual schema, 10 points**

The pseudo-code utilizes our conceptual schema with common problems that the designers encounter. The below conceptual schema image is highlighted with the sections that we are incorporating into our pseudo code:



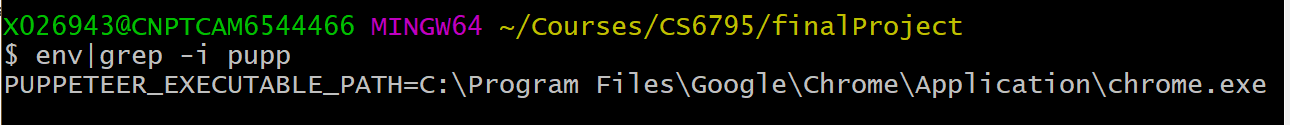
We support an automatic flagging system and collaboration feature in our application based on the outcomes of our interview process.

Specifically:

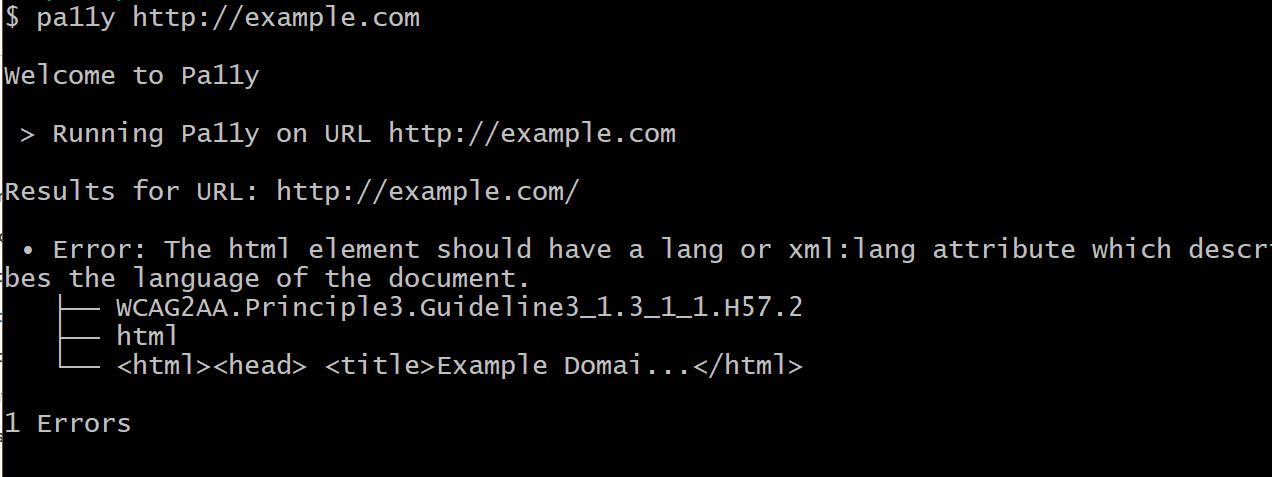
1. Our data analysis showed that designers must consider the constraints of tools used by students with disabilities and the challenges of accommodating unexpected formats in their designs. Hence, the flagging feature in our application will help the designers by considering the student's individual needs and the dynamic nature of available resources during operation to form suggestions for the designers to improve the system.
2. And collaborating with professionals from diverse backgrounds enables designers to gain valuable insights not easily obtainable when working alone. The assigning feature in our application utilizes this finding to allow the designers to assign tasks to the best-fit person on the team. Each team member has specific skills and expertise that can meaningfully contribute to the design process. The assigning feature optimizes collaboration by matching the right person to the right task, maximizing the team's collective potential, and driving successful outcomes.

**Application**

1. **1. The application code compiles and executes, 10 points**
2. **2. The application code is documented, and easy to run, 10 points**
   * 1. Create a directory where the application will be run. On windows: mkdir finalProject
     2. Install npm and node js on to your system. Choose the installation appropriate for your operating system using this link <https://docs.npmjs.com/downloading-and-installing-node-js-and-npm>
     3. Install the following node packages
        1. npm install puppeteer
        2. npm install express
        3. npm install express-xml-bodyparser
        4. npm install fs
        5. All the above packages are also included in the finalProject.zip/node\_modules directory.
     4. From the directory created in step 2.1.1, Install pa11y package using this link: <https://www.npmjs.com/package/pa11y>.
        1. OR: cd finalProject
        2. On windows 10, to install pa11y run: npm install pa11y
     5. Set environment variable to path to your browser executable. On Windows bash shell: export PUPPETEER\_EXECUTABLE\_PATH=C:\Program Files\Google\Chrome\Application\chrome.exe
        1. Use your path to the browser executable for the 2.1.5 step

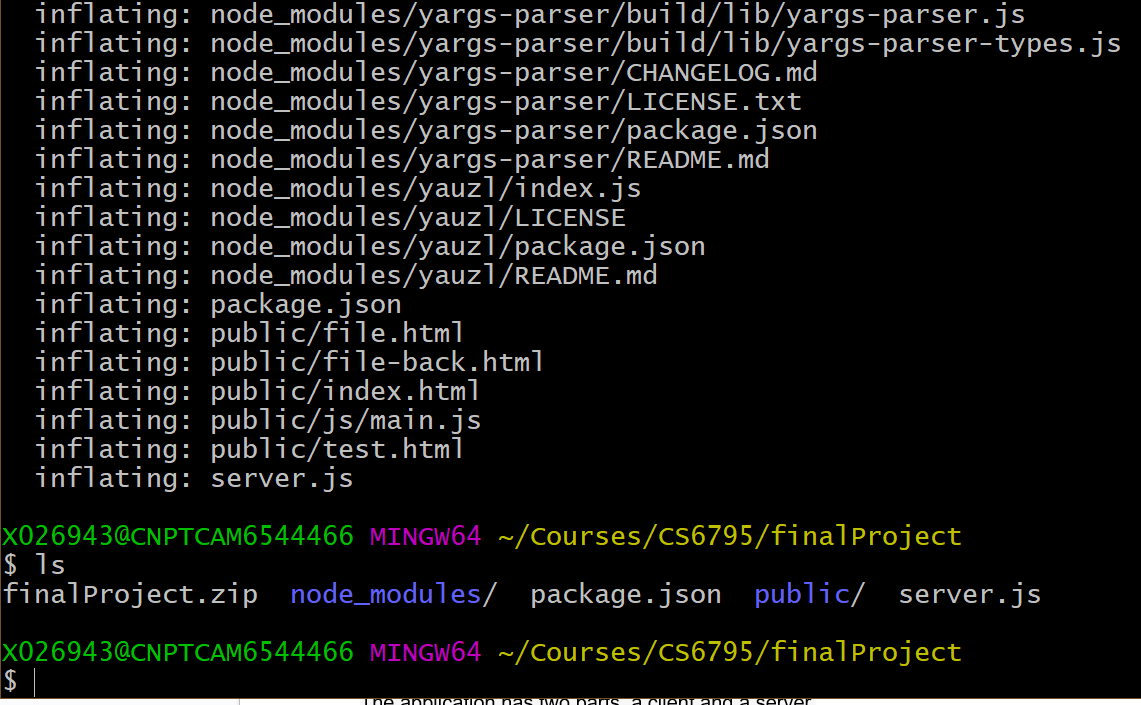


* + 1. To test if pa11y was installed properly run this example on the command line: pa11y <http://example.com>. You should see error messages on the standard output.



* + 1. Unzip the finalProject.zip on the finalProject directory created in step 2.1.1.

You should see the following directory structure.

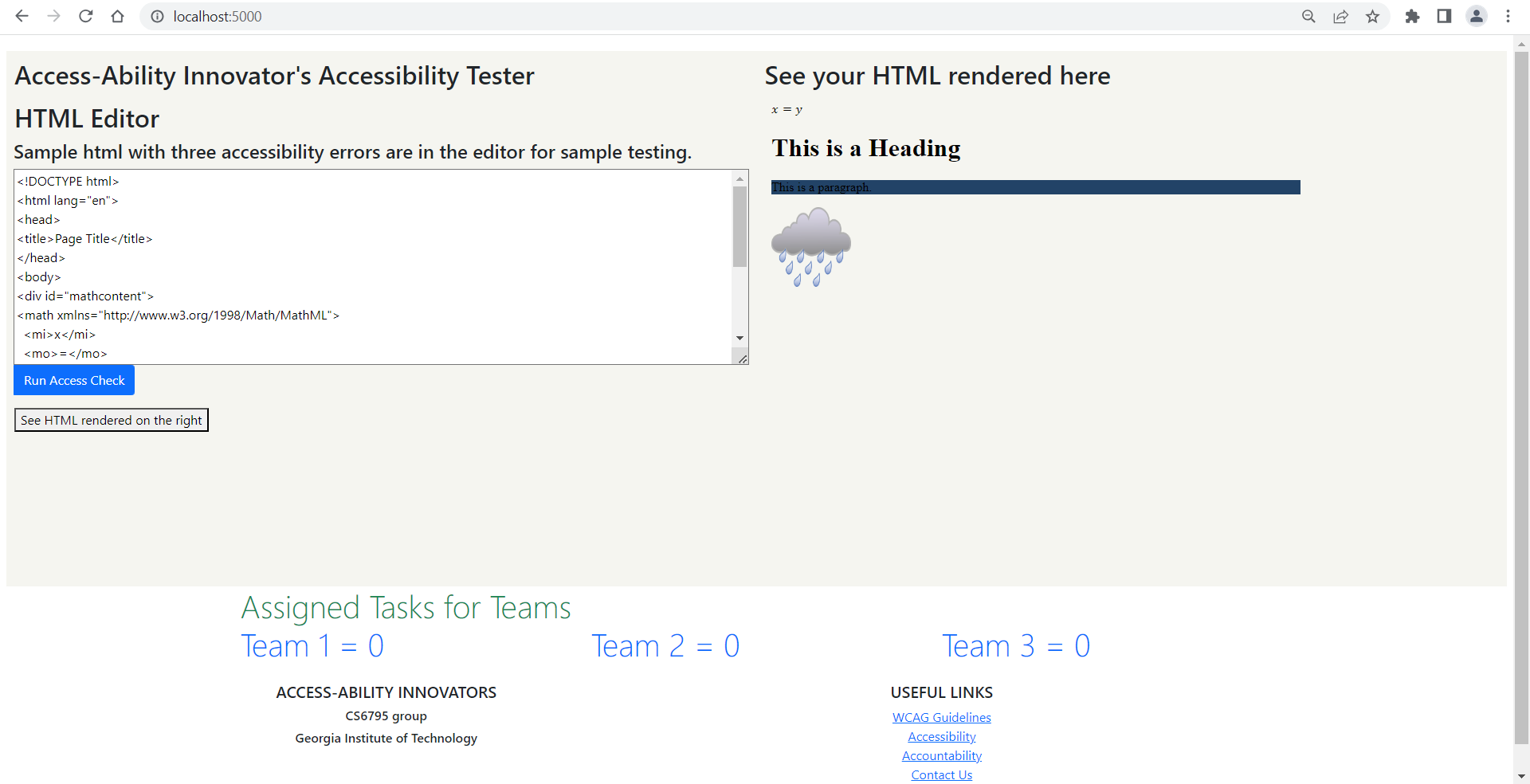


The application has two parts, a client and a server.

* + 1. To start the server run: node server.js



* + 1. To start the client, point your browser to http://localhost:5000



1. **3. The application code leverages elements from your conceptual schema, 20 points**

The final stage of the schema focuses on optimizing the tool's design by considering key elements that enhance accessibility. The application provides accessibility checks on the html document. The conceptual schema had several elements. We mainly focussed on some of the major problems that were discovered during the interview process. They are as follows:

* + 1. Color and contrast
    2. Text Description for Images (jpeg, png, gif, etc.)
    3. Error Handling
    4. Text Explanation for Mathematical Formulas
    5. Text-to-Audio Technology (Screen Reader including JAWS, NVDA, etc.)
    6. Haptic Feedback Assistant Technology

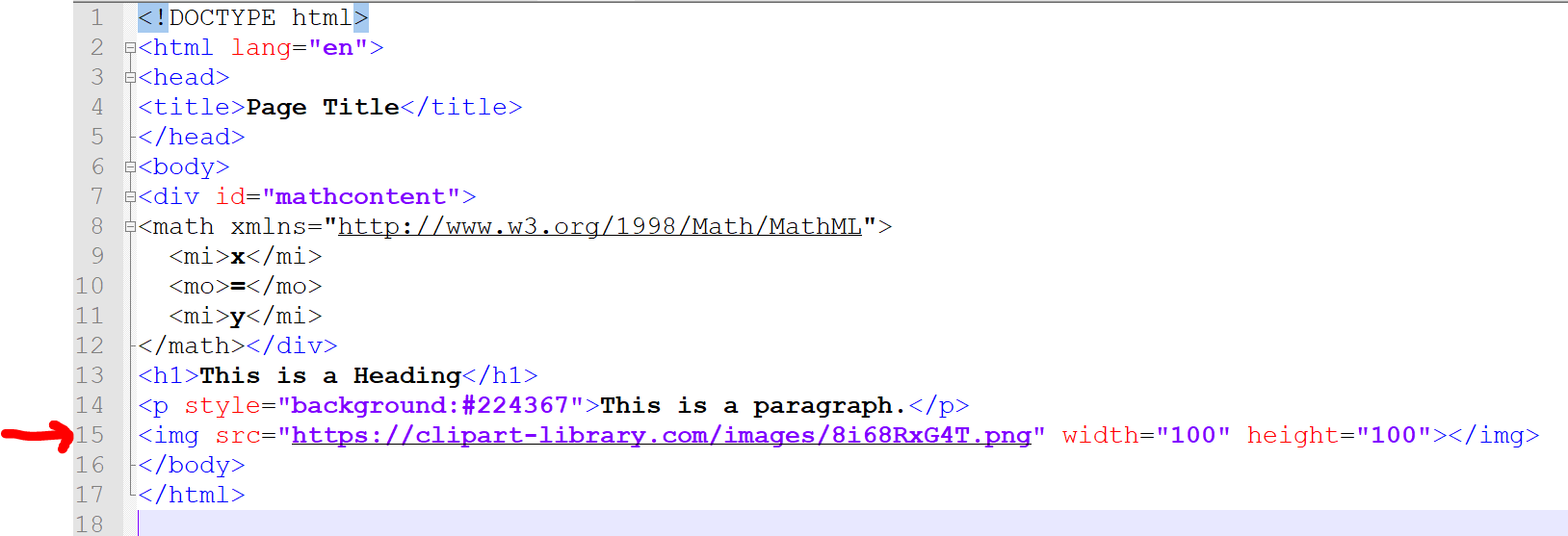
We focussed on delivering an application that would report errors on 3.1.1, 3.1.2, and 3.1.3. Future enhancements would handle 3.1.4, 3.1.5, and 3.1.6. Other future enhancements would be handling other file formats like pdf, media, images, audio, and video content identified during the interviews.

1. **4. The application intervenes appropriately (as explained in your report) in the decision making process, 40 points**
2. **5. Report and analysis of multiple runs through your application, exploring a variety of scenarios, 20 points**

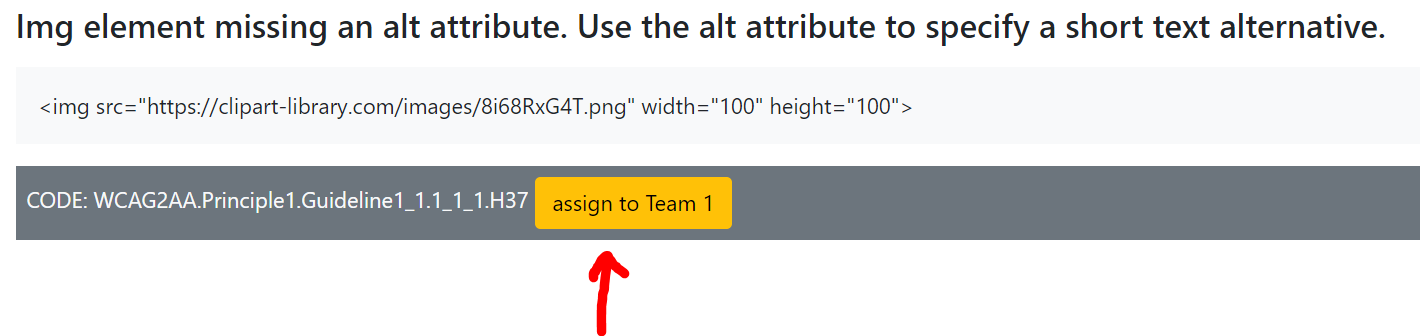
The application provides immediate feedback for the HTML source code used to construct the course material. The application is configurable to assign errors to appropriate teams based on team expertise. We will demonstrate multiple runs through the application.

Start with a sample html source code that has some WCAG-accessible errors.

1. **Scenario 1:** An image element missing the alt attribute with description in line 15 below is assigned to ‘Team 1.’

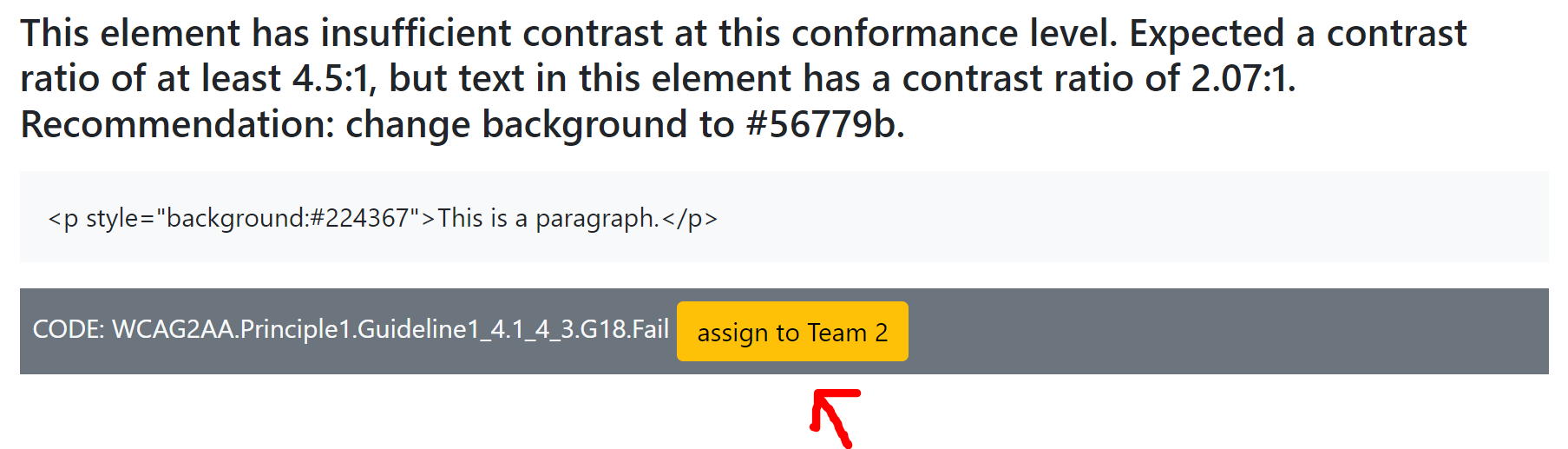


After running the ‘run access check,’ you will see this error reported by the application. The application is configured to assign image-related errors to ‘Team 1’.



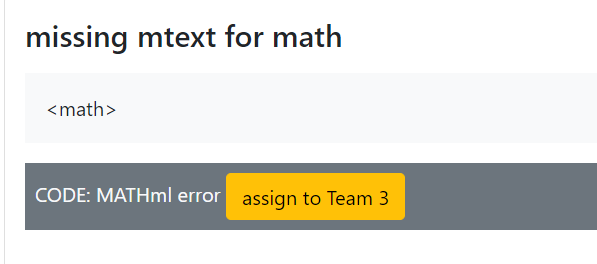
1. **Scenario 2:** A <p> element having a high color and contrast background in line 14 error below is assigned to ‘Team 2’.



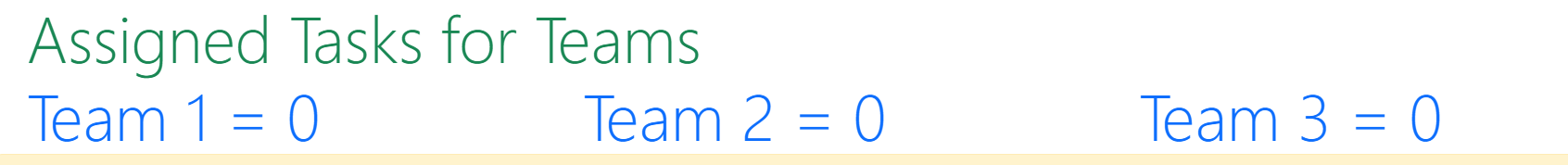


1. **Scenario 3:** Math ml related elements missing a description tag is flagged as error and is shown in the html code in lines 8 through 12 is assigned to ‘Team 3’.

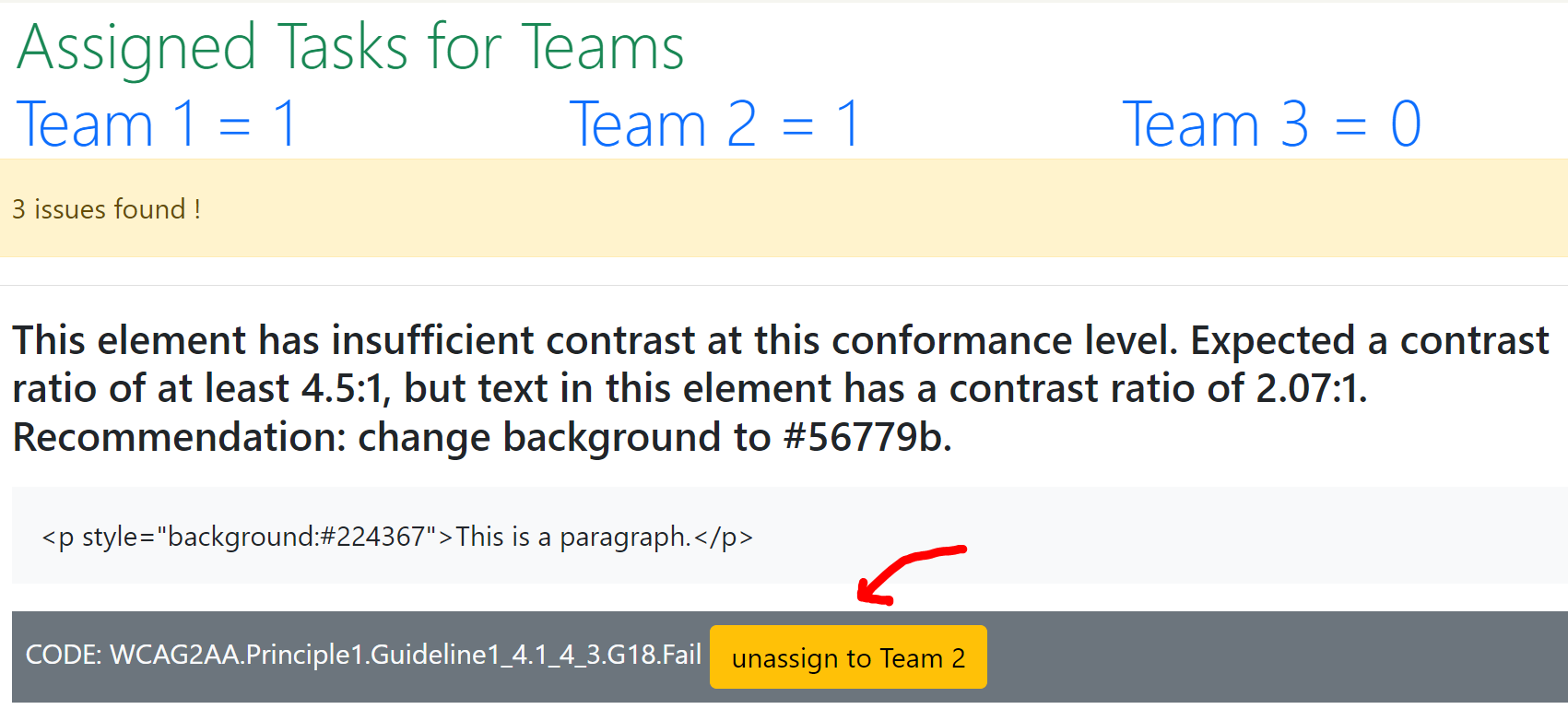




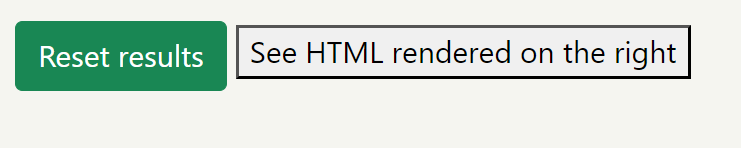
1. **Scenario 4:** Error handling support is provided in the application that is configurable to assign to a specific team for collaboration and feedback, as depicted by the assign/unassign buttons next to each error scenario. Once the errors are assigned, the task list shows how many errors are assigned to each team.



Once an error has been assigned the total number of tasks assigned to a team is reflected as shown below.



In the above example, Team 1 and Team 2 have one task assigned. Hence the yellow assign button next to the WCAG error is assigned, and the button text changes to ‘unassign to Team 2’. Designers can now click on ‘unassign to Team 2’ to remove the error from Team 2’s list.

****

**Reset Results button:** clicking this button clears the error messages, and the designer can either edit the HTML code and click the ‘Run Access Check’ button OR just click the ‘Run Access Check’ button to run on the same HTML code for re-assignment of errors.

**See HTML rendered on the right:** This button helps the designer preview the HTML source code generated as an HTML document in the browser.

The designer can fix the HTML source code during development and re-run the access checker until the HTML document is clear of all errors or assign the errors to specific configurable teams. The application, as coded, assigns all errors related to:

1. All errors related to “Text Description for Images (jpeg, png, gif, etc.)” to Team 1.
2. All errors related to “Color and Contrast” to Team 2.
3. All errors related to “Math ML” to Team 3.
4. Assign/Unassign button for feedback and collaboration