Software Requirements Specification and Design Description

for

WALA: Web Accessibility Evaluation Tool for Autism

Prepared by y Shayan N. Bhutta (2542413), Ali M. Chatkha (2486843), Batuhan Boynukalın (2315141), Nurlan Ildırımlı (2528396)

Middle East Technical University Northern Cyprus Campus

Computer Engineering

Supervised by Yeliz Yesilada, Sukru Eraslan

14/01/2024

Contents

1 Introduction

1.1 Purpose

In today's digital age, where online accessibility is considered a fundamental right, ensuring inclusivity for individuals with autism is more significant than ever. Our web browser extension, known as WALA (Web Accessibility for Autism), serves as a pioneering tool dedicated to evaluating webpages for autism friendliness according to established accessibility guidelines. With a primary focus on critical factors such as visual complexity, distinguishability, and text-to-image ratio, WALA becomes an indispensable resource for web developers striving to create websites that adhere to the unique needs of users on the autism spectrum.

Innovatively, WALA stands out in a landscape where tools for assessing autism friendliness are scarce. The dearth of comprehensive solutions to evaluate webpages against multiple metrics highlights the novelty and importance of our extension, which specifically aligns with recognized accessibility guidelines for people with autism. Moreover, our approach is grounded in empirical studies [1], drawing on insights provided by scholarly research on autism spectrum disorders. This ensures that our evaluation criteria are not only relevant but also rooted in a deeper understanding of the challenges faced by individuals on the autism spectrum.

By providing developers with actionable insights and tangible tools, our project goes beyond promoting equal access to information and services. It becomes a catalyst for enhancing comprehension and acceptance of autism spectrum disorders within the digital realm, aligning with established accessibility guidelines for people with autism. Armed with the knowledge derived from empirical studies, developers can proactively modify their websites to be more accommodating, thus bridging the digital gap and fostering a profound sense of belonging for individuals with autism in the expansive online landscape.

In summary, WALA's purpose extends beyond conventional web accessibility tools by specifically addressing the needs of individuals with autism, adhering to recognized accessibility guidelines. Its unique approach, grounded in empirical studies, not only sets it apart but also contributes to a more inclusive and understanding online environment, where every user can navigate the digital landscape with ease and a sense of belonging.

1.2 Scope

- **Benefits:**

  1. **Accessibility Improvement:** The extension empowers web developers to create more inclusive and autism-friendly web content.
  2. **Efficient Testing:** Developers can efficiently analyse and test their webpages for specific attributes that impact autism-friendliness, saving time and resources.
  3. **Guidance:** The extension offers guidance and insights to help developers understand how to enhance their web content to make it more accessible to individuals with autism.

- **Goals:**

  1. **Enhance Web Accessibility:** The primary goal is to facilitate the creation of autism-friendly web content by providing web developers with a tool to assess and improve their websites.

2. **Raise Awareness:** Increase awareness among web developers about the specific needs of individuals with autism in online environments and promote inclusive web design practices.
3. **Efficiency and Effectiveness:** Enable web developers to efficiently test and optimize their web content for autism-friendliness.

- **Objectives:**

1. **Develop the Extension:** Create a user-friendly browser extension that can analyse webpages for visual complexity, distinguishability, image complexity, image-to-text ratio, and text complexity.
2. **Scoring System:** Implement a scoring system for each of the five attributes, providing web developers with clear feedback on areas that require improvement.
3. **User Interface:** Design an intuitive user interface tailored for web developers, allowing them to easily understand and interpret the results.
4. **Educational Resources:** Provide educational materials within the extension to inform web developers about the significance of each attribute and how to make improvements.
5. **Documentation:** Create comprehensive documentation and tutorials for web developers on how to use the extension effectively.

By targeting web developers as the primary users, the scope of WALA addresses their specific needs in making web content more accessible to individuals with autism. It not only empowers developers to create inclusive websites but also fosters a community of practice around autism-friendly web design.

1.3 Related Work

Autism Spectrum Disorder (ASD) poses unique challenges, particularly in the context of web accessibility. The growing prevalence of ASD has accentuated the need for websites to be designed with careful consideration for individuals with autism, considering their specific cognitive and sensory requirements [10].

Existing international accessibility guidelines, though well-intentioned, lack empirical validation within the context of ASD. According to [1] autism spectrum disorder (ASD) is one of the cognitive disabilities that have been least researched in web accessibility. These guidelines, namely, WCAG 2.1, Guideline 1.3 and Guideline 1.4 often rely on diagnostic criteria rather than concrete behavioural studies, leading to potential discrepancies between the guidelines' anticipations and the real-world experiences of individuals with autism.

In response to this gap, recent research [1] conducted a groundbreaking eye-tracking study. This study delved into the intricacies of web accessibility for individuals with autism, focusing on the visual complexity of web pages and the distinguishability of page elements. By utilizing the VIPS algorithm and the VICRAM tool, the research empirically evaluated the browsing patterns of individuals with autism. The study's findings underscored the nuanced ways in which individuals with autism interact with visually complex or poorly distinguishable web elements. Specifically, the study revealed that high visual complexity and low distinguishability led to non-equivalent user experiences for individuals with autism, shedding light on the critical impact of web design on their online interactions.

Furthermore, we can talk about the ontology-based heuristic approach for the automatic identification of visual elements and their roles on web pages, with a primary focus on enhancing web accessibility. The architecture comprises three major components: the automatic identification of visual elements using the Vision Based Page Segmentation Algorithm (VIPS), the generation of heuristic rules from an ontology called "eMine," and the application of these rules to web pages using the Jess rule engine for automatic annotation of visual elements and their roles. The heuristic rules are designed to capture detailed knowledge about the visual elements, considering factors such as underlying tags, element size, position, and other attributes. To complement this approach, we utilized the VICRAM tool, referenced as [12], to compute visual complexity, and VIPS for calculating distinguishability. VICRAM assists in assessing the overall visual complexity of web pages, providing valuable insights into the intricacies of the visual design. By integrating these tools, the combined methodology aims to offer a comprehensive analysis of web page visual elements, addressing both their roles and visual characteristics.

Visual complexity and Distinguishability alone are not sufficient metrics for people with autism to use websites in a more accessible way. Readability is also important metric for web accessibility. It is influenced by collection of textual elements that impact a reader's comprehension. It is shaped by how many years of schooling it would take someone to understand the content. In this context, recent study [6] states that it is highly desirable that the language should be as much straightforward, plain, and simple. The usage of language with characteristics makes the design of the interface quick to understand and navigate easily on the web page [6]. These empirical insights underscore the significance of developing tools that address the specific needs highlighted in [1]. Our project aims to leverage these findings by creating a user-friendly browser extension. This extension will serve as a practical implementation of empirically validated guidelines, identifying and addressing visual complexities and distinguishability issues on web pages. By integrating these behavioural insights into our tool, we strive to bridge the existing gap between theoretical guidelines and the lived experiences of individuals with autism, thereby fostering a more inclusive digital environment.

Talking about the use of images in text documents to enhance autism spectrum disorders with Autism Spectrum Disorder (ASD) [13]. It employs eye-tracking technology in the first study of its kind with ASD participants to evaluate text documents. The study aims to understand differences in attention patterns between autistic and non-autistic individuals, particularly in the context of image and text combinations. The research compares two types of images, photographs, and symbols, to determine their effectiveness in simple documents. Additionally, the study evaluates human-produced easy-read documents as a gold standard for accessibility, providing insights into perceived difficulty levels and preferences of autistic individuals in text presentation. The results contribute to the development of guidelines for creating accessible text for autism. The paper [13] is instrumental in shaping our understanding of the necessity to compute text complexity for individuals with autism, serving as a foundational reference in our research.

1.4 Product Overview

1.4.1 Product perspective

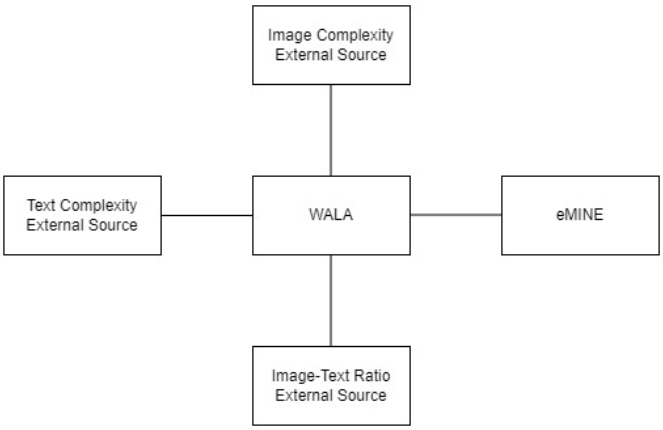

Figure 1: Context-Block Diagram of WALA

Figure 1 shows the Context-Block diagram of WALA. We are currently assuming that all of 5 metrics would be calculated from external sources, but they might become part of the system later if we decide to incorporate it in the future. "eMINE" (https://bitbucket.org/metu-emine/role-detection-api/) will be responsible for providing us with the visual complexity and distinguishability of the requested webpage.

1.4.1.1 System Interfaces

1. **Interface for Visual Complexity and Distinguishability:** Requests visual complexity and distinguishability data from an external service.
2. **Interface for Image to Text Ratio:** Requests image to text ratio data from a specific external service.
3. **Interface for Text Complexity:** Requests text complexity analysis from a designated external service.
4. **Interface for Image Complexity:** Requests image complexity analysis from an external service.

1.4.1.2 User interfaces

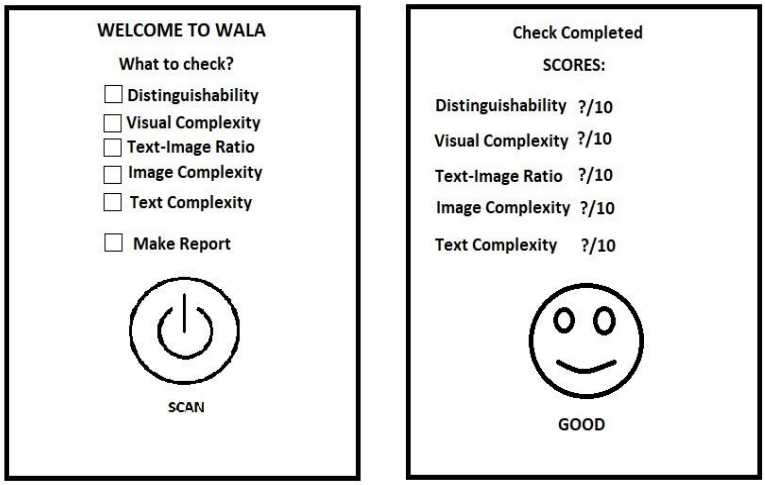

Figure 2: GUI of WALA

- Figure 2 above shows the general GUI of WALA. It shows the options of metrics a user can chose from a list of 5 metric options in terms of checkboxes. Moreover, an additional, Make Report checkbox is present to indicate if a user wants a report generated. All of this will work if the user presses on the big scan button.

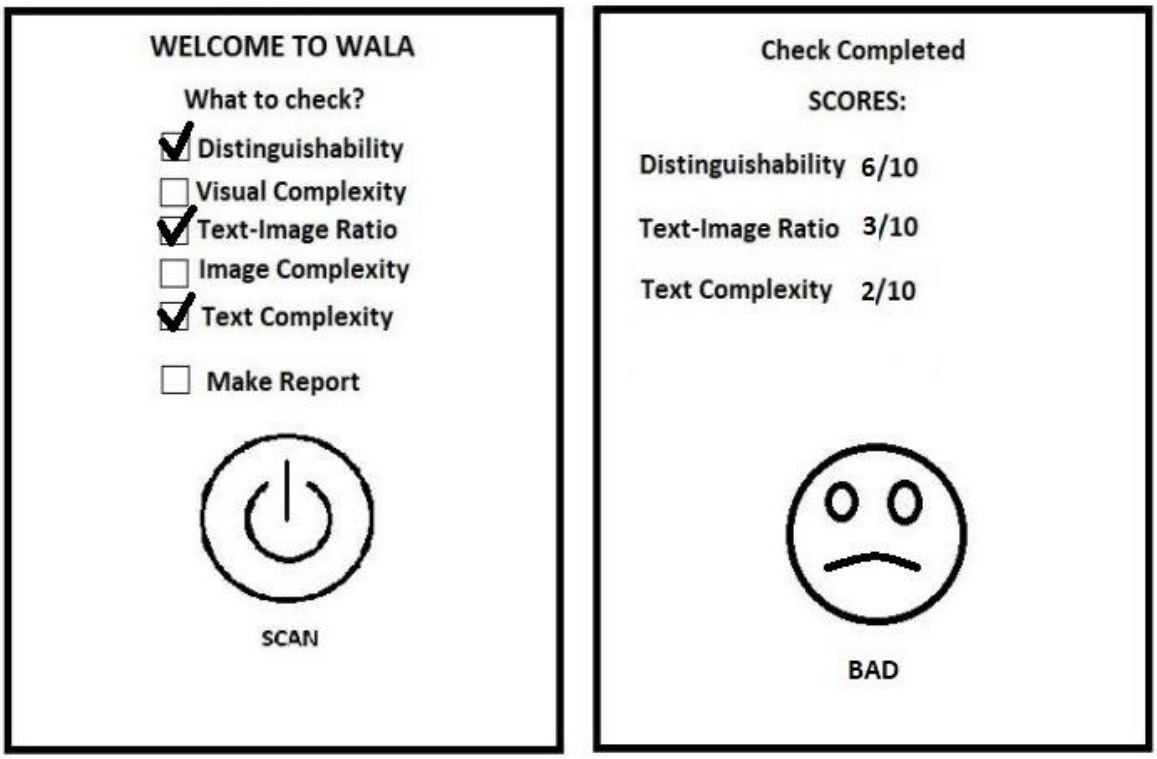

Figure 3: GUI of WALA

- Figure 3 above shows the general GUI of WALA. It shows the selected complexities and their scores. In this figure, Make Report is not selected.

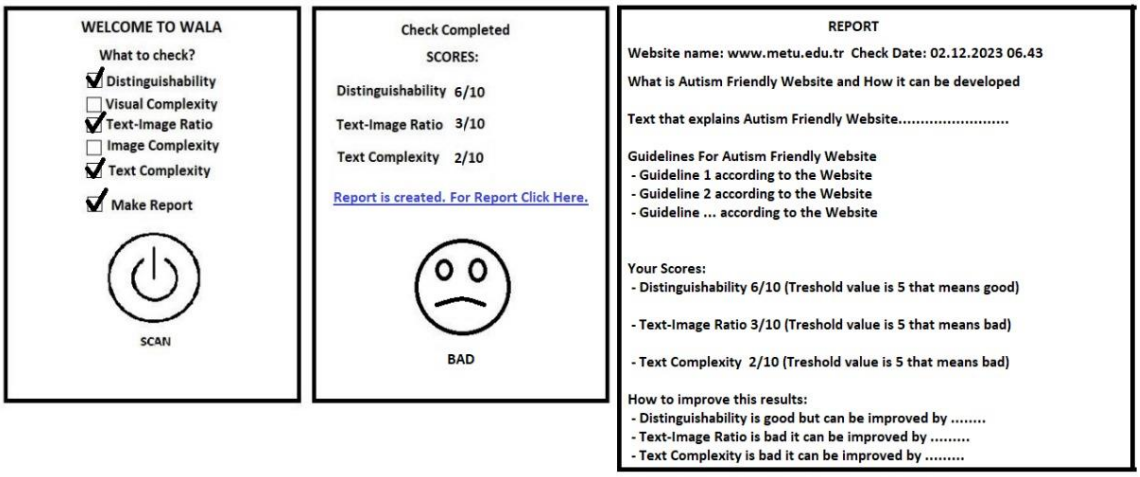

Figure 4: GUI of WALA

- Figure 4 above shows the general GUI of WALA. It shows the selected complexities and their scores. In this figure, Make Report is selected, and general structure of report has been shown.

1.4.2 Product functions

- The system shall allow to take the link of the website and calculate and give score about followings:
  - Visual Complexity
  - Distinguishability
  - Image Complexity
  - Text Complexity
  - Text-Image Ratio

- The system shall allow developers to see possible problematic areas on their websites.
- The system shall allow developers to see possible solutions for problematic areas on their websites.
- The system shall allow users to see if website is autism friendly or not with emoticons and metric scores.

1.4.3 Identified stakeholders and design concerns.

Stakeholders and their concerns in the context of the described browser extension for evaluating web content:

1. **Users:**

Concerns: User experience (user accessibility) in terms of ease of interface, clarity in metric selection, comprehensibility of the generated report, relevance of suggestions, and understanding the emoji-based accessibility indicator.

2. **Developers/Technical Team:**

Concerns: Efficient integration with the eMINE API, internal metric calculations, system performance, scalability, maintenance, and ongoing updates to ensure compatibility with evolving web technologies.

3. **Content Creators/Publishers:**

Concerns: Interpretability of metric scores, clarity in improvement suggestions, alignment with content creation workflows, and understanding the criteria for the "Autism Friendly" emoji.

4. **Accessibility Advocacy Groups:**

Concerns: Accuracy and fairness in evaluating content for autism-friendliness, alignment with established accessibility standards, and the effectiveness of the tool in promoting inclusivity.

Design Concerns:

1. **Scalability:**

Ensuring that the system can handle a growing user base and increasing volumes of content evaluations without compromising performance.

2. **Security:**

Implementing robust security measures to protect user data, especially when interacting with external APIs and storing sensitive information.

3. **Usability:**

Designing an intuitive and user-friendly interface to facilitate a positive user experience and encourage widespread adoption.

4. **Maintainability:**

Developing the system in a way that makes it easy to update, fix issues, and adapt to changes in web technologies over time.

5. **Integration:**

Ensuring smooth integration with the eMINE API and other internal components to provide accurate and timely metric calculations.

6. **Interpretability:**

Designing the report and metric scores in a way that is easily understandable for users, content creators, and other stakeholders.

1.4.4 User characteristics

1. **Web Developers:** The intended user group includes web developers with diverse educational backgrounds, ranging from self-taught enthusiasts to individuals with formal education in computer science or web development. Their experience levels vary widely, from beginners to seasoned professionals, and their technical expertise covers programming languages like HTML, CSS, and JavaScript, along with familiarity with web development frameworks.
2. **Accessibility Experts:** This group brings a distinct set of characteristics to our audience. These professionals boast specialized knowledge in accessibility standards and user experience-related fields. Their expertise lies in evaluating website accessibility, providing insightful recommendations for enhancement, and utilizing specialized tools for comprehensive assessments. Meeting their expectations involves aligning features with established standards and ensuring a seamless integration into their workflow.

3. **Individuals with Autism or Cognitive Disabilities:** The end-users of the assessed websites may have varying educational backgrounds and experiences in using the internet. Their technical expertise is generally limited, focusing more on ease of use and user experience rather than technical intricacies.

1.4.5 Limitations

1. **External Service Dependencies:** The functionality of our web browser extension relies on external services for metrics such as visual complexity, text complexity, and image analysis. Limitations in these external services, such as rate limits, API constraints, or service downtimes, can restrict the frequency and volume of requests your extension can make, affecting real-time assessments and responsiveness.
2. **Data Accuracy and Consistency:** The accuracy of assessments is crucial for our extension's reliability. Limitations in the accuracy of data provided by external services or discrepancies between assessments can pose challenges. Ensuring consistent and precise metrics across various webpages might be challenging due to the inherent complexities of web content.
3. **Browser Compatibility:** As a browser extension designed for Google Chrome, it may not offer seamless functionality on alternative browsers like Firefox. While efforts are made to optimize compatibility, some features or APIs may not perform uniformly across different browsers, affecting the user experience and feature availability on non-Chrome platforms.
4. **Network Connectivity Dependency:** The extension's functionality is contingent upon a stable internet connection. In scenarios where users experience network connectivity issues, the extension's ability to perform real-time assessments and fetch external data may be impeded, affecting the overall user experience.

1.4.6 Assumptions and dependencies

1. **Correctness of eMINE Metrics/Results**:
   Assumption: eMINE consistently sends correct metric scores back to the extension.
   Dependency: The accuracy of the extension's assessments relies on eMINE providing reliable metric data.
2. **eMINE Threshold Assumptions:**
   Distinguishability: A score less than 0.5 is considered low distinguishability and deemed non autism friendly.
   Visual Complexity: A score greater than 5 out of 10 is classified as high visual complexity, indicating non-autism friendliness.
3. **Thresholds for Internal Metrics:**
   Assumption: Internal metrics (text-image ratio, image complexity, text complexity) have predefined thresholds for assessing website elements.
   Dependency: The extension uses specified thresholds to evaluate internal metrics:
   - Text-Image Ratio: A ratio greater than 0.5 is considered bad.
   - Image Complexity: A score greater than 5 is considered bad.
   - Text Complexity: A score greater than 5 is considered bad.

2 Specific requirements

2.1 External interfaces

System Interface	Functionality	Input	Output
External Service for Visual Complexity and Distinguishability (eMINE)	Assess webpage for visual complexity and distinguishability.	Webpage content for analysis.	Metrics indicating visual complexity and distinguishability.
External Service for Image Complexity	Assess the complexity and clarity of images on the webpage.	Images from the webpage for analysis.	Metrics indicating image complexity and clarity.
External Service for Text Complexity	Evaluate textual complexity of the content on a webpage.	Textual content from the webpage for analysis.	Complexity level of the textual content.
External Service for Text-Image Ratio	Analyse the balance between textual and graphical content.	Webpage content for analysis.	Ratio indicating the balance of text to images.

2.2 Functions

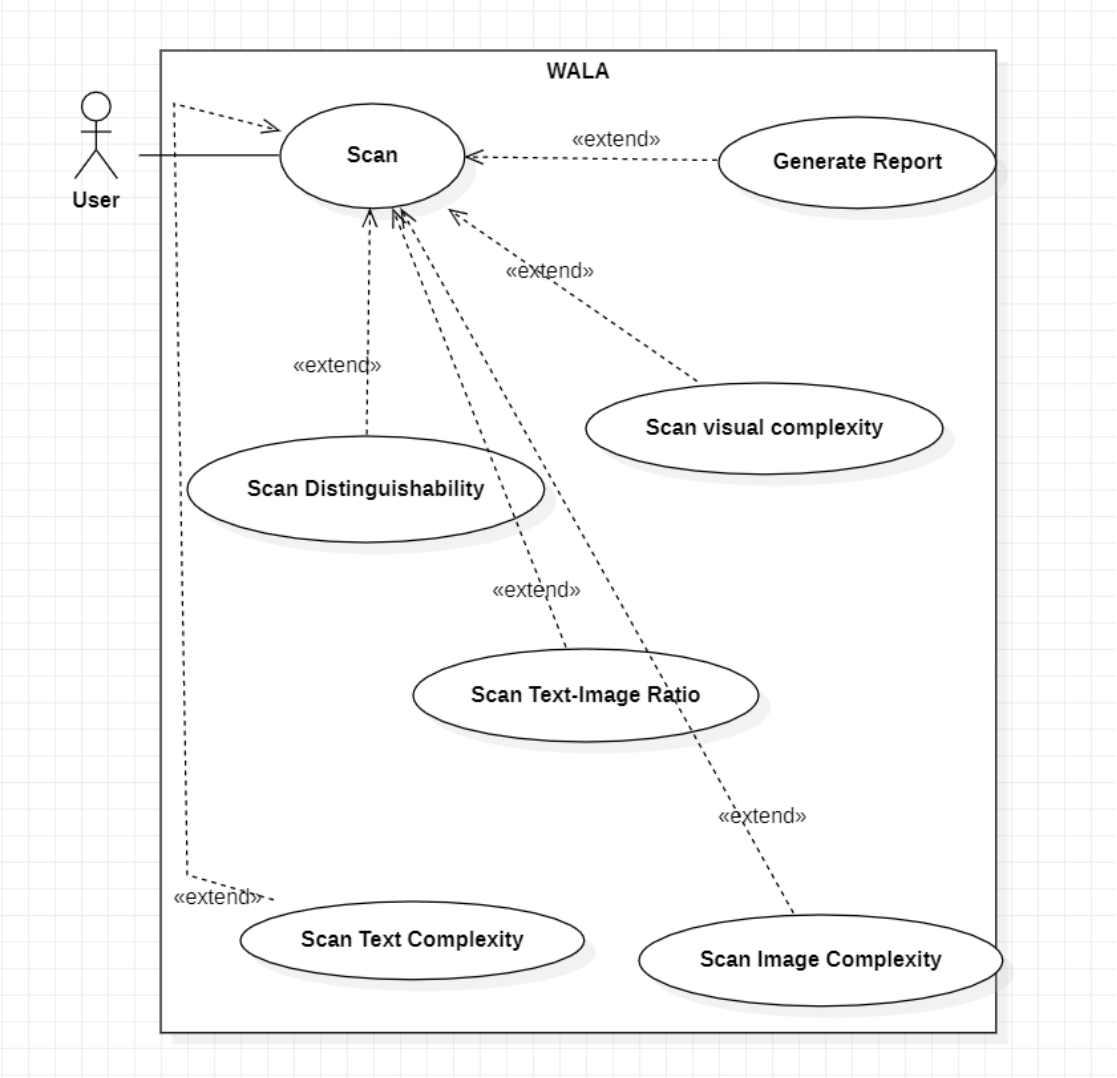

Figure 5: WALA Use-Case Diagram

Figure 5 above shows our Use-Case Diagram for WALA. It has one actor which is User (or Web Developer) and has one main use-case i.e., Scan which means scanning the whole webpage to retrieve information about the five inclusions which are scanning visual complexity, distinguishability, text complexity, image complexity and text-image ratio.

Detailed Use Case Description

Use case	Scan
Actors	User
Cross references	Report
Typical Course of Events	
Actor Intentions	**System Responsibility**
1) User launches the extension	
	2)System loads it up
3)User clicks on Scan	

	4)System displays 5 options (all reference except report) as a check in box
5)User selects the one to compute	
	6)System calculates those results
	7)System displays the result in the form of emoji scale
	8)System loads Report button
9)User closes the extension	
Alternative Courses	
5)User ticks Distinguishability	
	6)System sends the URL to Emine
	7)System receives feedback from Emine
	8)System displays the result in the form of emoji scale
	9)continue to step 8
5) User ticks Visual complexity	
	6)System sends the URL to Emine
	7)System receives feedback from Emine
	8)System displays the result in the form of emoji scale
	9)continue to step 8
5)User ticks Image Complexity	
	6) System calculates the Image Complexity
	7)System displays the result in the form of emoji scale
	8)continue step 8
5) User ticks Text Complexity	
	6) System calculates the Text complexity
	7)System displays the result in the form of emoji scale
	8)continue step 8
5)User ticks Text-Image ratio	
	6) System calculates the Text-Image ratio
	7)System displays the result in the form of emoji scale
	8)continue step 8

9) See Report use case	

Use case	Report
Actors	User
Cross references	Scan
Typical Course of Events	
Actor Intentions	**System Responsibility**
1)See Scan use case	
	2) Display generates report button below the emojis
3) User clicks on the button	
	4)System uses the information from Scan to generate a report
	5)System displays the PDF report
	6)System displays a download report button
7)User clicks on download button	
	8)System downloads the report
9)User closes the extension	

2.3 Usability Requirements

1. **User Satisfaction:** The system should be user-friendly and intuitive, ensuring that users can navigate the interface easily and understand the assessment results. User satisfaction is vital for the tool's acceptance and continued usage.
2. **Accuracy and Precision:** The system should provide precise scores and detailed feedback on each evaluated criterion. Accurate and precise information is essential for web developers to pinpoint specific issues on their webpages. Clear feedback enables developers to make targeted improvements, enhancing the overall autism friendliness of their websites. Precision in feedback is critical for actionable insights and effective website optimization.
3. **Reliability and Consistency:** The system should produce consistent results across different evaluations of the same webpage. This means that if a same webpage is scanned by multiple users, similar scores should be provided to them unless the webpage has been updated. Consistency in assessment results ensures that web developers can trust the tool's feedback. Reliable and consistent evaluations are necessary to establish the tool's credibility and build user confidence, encouraging its adoption and regular use.

2.4 Performance requirements

- **Response Time:** The extension should provide assessment results within a reasonable response time, typically within a few seconds after scanning a webpage. A swift response time ensures that web developers can efficiently analyse multiple webpages one after another, enhancing their productivity and encouraging regular use of the tool.

- **Scalability:** The extension should be able to handle assessments for various types of webpages, including those with complex designs and large amounts of content.
  Scalability is essential to accommodate the diverse nature of websites, ensuring that the tool remains effective and reliable regardless of the complexity or size of the webpage.
- **Accuracy and Consistency:** The extension's assessment results must be accurate and consistent across different webpages and usage scenarios. Ensuring reliability in assessment outcomes is fundamental for web developers to trust the tool's feedback and make informed decisions regarding website adjustments.
- **Resource Utilization:** The extension should use system resources efficiently, such as CPU and memory. By optimizing resource usage, the tool minimizes its impact on the user's device, ensuring that it operates smoothly without causing undue strain on system resources.
- **Portability in Systems:**
  Given the dynamic nature of web technologies, the extension should demonstrate adaptability to evolving standards and frameworks. This ensures that the tool remains relevant and effective as web development practices continue to evolve, providing lasting value to its users.

2.5 Logical database requirements

In our project, since we are working on a Chrome extension, no system data will be stored in a database. Therefore, a database will not be used, and as a result, there is no Entity-Relationship Diagram (ERD) for it.

2.6 Software system attributes

1. **Reliability:**

Factor: Error Handling and Recovery Mechanisms

Justification: The reliability of the web browser extension is paramount to provide accurate and consistent assessments of webpages. Implementing robust error handling and recovery mechanisms ensures that the extension can gracefully handle unexpected errors, such as network issues or invalid webpage structures, ensuring reliable performance and precise evaluations.

2. **Availability:**

Factor: Redundancy and Failover Mechanisms

Justification: Ensuring the availability of the web browser extension is crucial for web developers who rely on timely feedback to optimize their websites. By implementing redundancy and failover mechanisms, such as backup servers and load balancing, the extension can maintain continuous availability.

3. **Maintainability:**

Factor: Modularity and Documentation

Justification: A modular structure allows developers to update or enhance specific components without affecting the entire system. Comprehensive documentation, including

clear explanations of assessment criteria and guidelines for interpretation, enables both developers and users to understand the extension's functionality thoroughly.

4. Extensibility:

Factor: Plugin and Module Integration

Justification: To enhance the extensibility of the web browser extension, it should support seamless integration with plugins and modules. This enables developers to extend the functionality of the extension by adding new features or customizing existing ones without extensively modifying the core system. The extensibility factor promotes versatility, allowing the tool to adapt to diverse user needs and evolving web development requirements.

2.7 Supporting information.

N/A

3 Software Estimation

General System Characteristics (GPC)	Degree of Influence (DI) 0: least, 5: most	
Data communications	2	
Distributed processing	2	
Performance	3	
Heavily used configuration	0	
Transaction rates	2	
Online data entry	2	
End-user efficiency	3	
Online updates	4	
Complex processing	3	
Reusability	4	
Installation ease	0	
Operational ease	0	
Multiple sites	0	
Facilitate change	4	
Total of Degree of Influence	**29**	
Value adjustment factor (VAF)	**0,94**	VAF = TDI*0.01+0.65

Figure 6: Software Estimation (1)

Inputs	Option(L),Website(H)
Outputs	report(H),Visual Comp.(L), Image Comp.(L), Text Comp.(L), Image-Text Ratio(L), Distinguishability(L)
Inquiries	Computing Image Comp.(L), Computing Text Comp.(L), Computing Text-Image Ratio(L)
Logical Internal Files	Guidelines(L)
External Interface Files	EMine(M)

Figure 7: Software Estimation (2)

Program Characteristics	Low Complexity	Medium Complexity	High Complexity	Total	
Inputs	1	0	1	9	
Outputs	5	0	1	27	
Inquiries	3	0	0	9	
Logical Internal Files	1	0	0	7	
External Interface Files	0	2	0	14	
Unadjusted Total of Function Points (UTFP)				66	
Adjusted Total of Function Points (ATFP)				62,04	ATFP = UTFP*VAF

Figure 8: Software Estimation (3)

Programming Language	Percentage in Project	Language Unit Size	Language Unit Size in Project	
HTML	0,1	15	1,50	
JavaScript	0,3	80	24	
Python	0,5	20	10	
CSS	0,1	16	1,60	
			0	
			0	
Language Unit Size			37	
Lines of Code			2295	LOC = ATFP * Language Unit size
Kilo Delivered Source Instruction (KDSI)			2,30	ATFP*Language Unit size/1000

Figure 9: Software Estimation (4)

Development Type	Organic	
Estimated Effort in Man-Months	5,75	MM = a*KDSI^b
Estimated Development Time in Months	4,86	TDEV = 2.5*MM^c
Estimated Team Size	2,00	MM/TDEV

Figure 10: Software Estimation (COCOMO) (5)

- We selected our development type as Organic because our project is not a huge innovation for the technology sector, and we will be working with mostly external services which are already usable. Moreover, our project size is not that high as we calculated our lines of code in estimation.

Adjusted Total of Function Points (ATFP)	62,04	
Kind of Software	Business	
Software Organization's Skills/Abilities	Average	
Estimated Effort in Man-Months	7,60	(ATFP^(3*0.43))/27
Estimated Development Time in Months	5,90	ATFP^0.43
Estimated Team Size	2,00	7,60/5,90

Figure 11: Software Estimation (Jones First Order) (6)

- We selected our kind of software as Business because, if we compare other choices, our product will not be published as shrink-wrap product (not sold in stores). Also, our product is not going to work with a system since it is a web-based project. Furthermore, we selected our Software Organization's Skills/Abilities as average because as a team, we are not the best skilled ones in terms of coordination and teamworking as we are still in our undergraduate programs. In other way, we are not also the worst ones as we have started to get along and get each other's style of working and we will soon be getting even better. So, we decided to use average option here.

Schedule Tables Estimate	Nominal Schedules	
Lines of Code	2295	~10000
Estimated Effort in Man-Months	9	
Estimated Development Time in Months	6	

Figure 12: Software Estimation (7)

- We selected schedule table estimate as Nominal Schedule because as undergraduate students, we cannot assume that everything we will work on will be right/correct, and there will be always a case that we will have to work on. So, we selected less optimistic assumption.

- Upon employing Jones First Order for estimation, our calculated effort stood at 7.60 man-months, while the schedule table estimate indicated a requirement of 9 man-months while COCOMO has the lowest value 5.75 man-months. This variance underscores the need for additional man-months with the COCOMO approach, highlighting potential task complexities. Conversely, the development time estimation revealed 5.90 months via COCOMO and 6 months in the schedule table estimate. However, there is also difference between First Order (which is nearly same as Nominal Schedule Table) and COCOMO. These values are indicating a close alignment in the anticipated development duration, emphasizing consistent progress in our development efforts.

4 Architectural Views

The architecture of software systems will be described based on the Logical View, Process View, Development View, and Deployment View, as suggested by Kruchten (1995)[1]

4.1 Logical View

The logical view shows the key abstractions in the system as object classes.

4.1.1 Class Diagram

WALA's system was not designed to be object-oriented and thus shown below are the required functions and required data types of our system.

1. <u>Required Functions</u>

- **Scan Webpage:**

  Description: This function will initiate the process of scanning the current webpage.

  Parameters: None

  Return Type: None

- **Fetch Webpage Data:**

  Description: Retrieves relevant data from the current webpage, including HTML/CSS, text, and images.

  Parameters: None

[1] P. B. Kruchten, "The 4+1 View Model of architecture," in IEEE Software, vol. 12, no. 6, pp. 42-50, Nov. 1995, doi: 10.1109/52.469759.

Return Type: Webpage Data Structure

Assumption: Parameters are none as the browser extension automatically takes the current webpage URL it is opened on.

- **Call external API "eMINE" for Visual Complexity and Distinguishability:**

  Description: Sends the webpage data to the "eMINE" service to calculate visual complexity and distinguishability.

  Parameters: URL

  Return Type: Visual Complexity Score, Distinguishability Score

- **Evaluate Text-Image Ratio:**

  Description: Calculates the ratio of text to images on the webpage.

  Parameters: Text elements, image elements by first parsing the webpage using BeautifulSoup Python Library and getting these elements. (https://www.crummy.com/software/BeautifulSoup/bs4/doc/).

  Return Type: Text-Image ratio score (numeric)

- **Evaluate Text Complexity:**

  Description: Analyses the complexity of the text content on the webpage.

  Parameters: Text content by first parsing the webpage using BeautifulSoup Python Library and getting all the text elements. (https://www.crummy.com/software/BeautifulSoup/bs4/doc/).

  Return Type: Text complexity score (numeric)

- **Evaluate Image Complexity:**

  Description: Assesses the complexity of images on the webpage.

  Parameters: Image elements by first parsing the webpage using BeautifulSoup Python Library and getting all the image elements. (https://www.crummy.com/software/BeautifulSoup/bs4/doc/).

  Return Type: Image complexity score (numeric)

- **Generate Emoji Feedback:**

  Description: Converts the individual scores into a concise and user-friendly emoji-based feedback indicating if the webpage is autism-friendly or not.

  Parameters: Scores for visual complexity, distinguishability, text-image ratio, text complexity, image complexity

  Return Type: Emoji string.

- **Generate Report:**

  Description: This function compiles all metric scores and provides suggestions to improve the webpage based on these scores.

  Parameters: Scores for visual complexity, distinguishability, text-image ratio, text complexity, image complexity

Return Type: Report (string or document)

2. <u>Special Data Structures and Required Data Types</u>

- **Report Data Structure:**

  Description: A structured format to hold information about metric scores and corresponding suggestions.

  Fields:

  Visual Complexity Score

  Distinguishability Score

  Text-Image Ratio Score

  Text Complexity Score

  Image Complexity Score

  Suggestions (for each metric)

- **Webpage Data Structure:**

  Description: A structure or object to hold information about the webpage, including HTML/CSS elements, text content, and image elements.

- **Score Data Type:**

  Description: A data type to represent the scores for visual complexity, distinguishability, text-image ratio, text complexity, and image complexity.

- **Emoji Data Type:**

  Description: A data type to represent the emoji feedback generated based on the scores.

4.2 Process View

The process view shows how the system is composed of interacting processes.

4.2.1 Activity Diagram

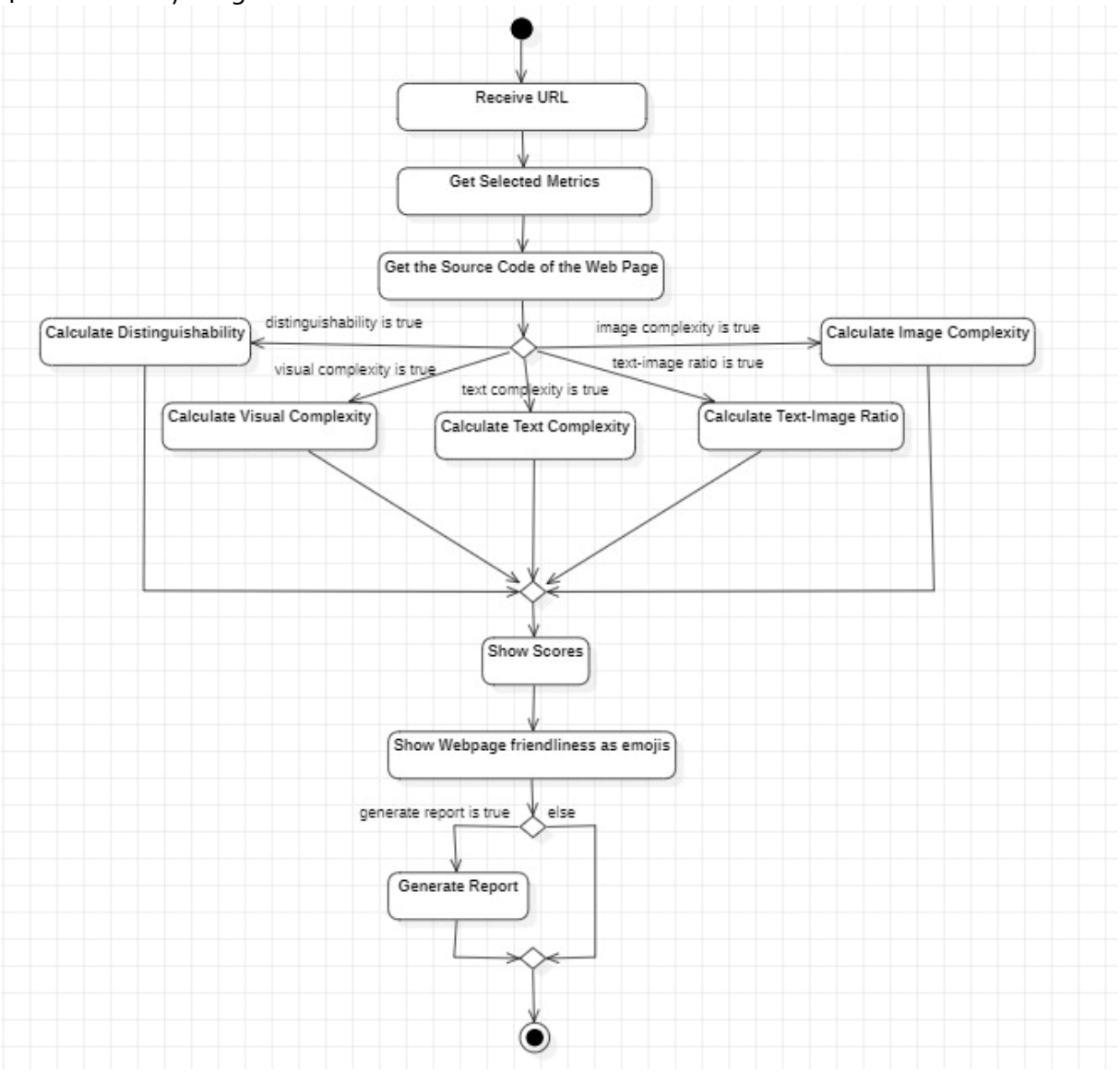

Figure 33: Activity Diagram of "Scan + Report" Use Case

The figure 13 above shows the activity diagram of our "Scan + Report" use case. It starts by receiving the URL of the current webpage (the browser extension is opened on), and the metric choices selected by the user (for example visual complexity only). According to the metric choices, the specific decision branches are taken, even though the decision node is there, multiple lines may be active depending on users' choice. After that, the relevant scores are displayed. Following that, the webpage friendliness is shown through a relevant emoji. Lastly, if the generate report option is selected, a decision branch is taken, and the report is generated.

4.2.2 Sequence Diagrams

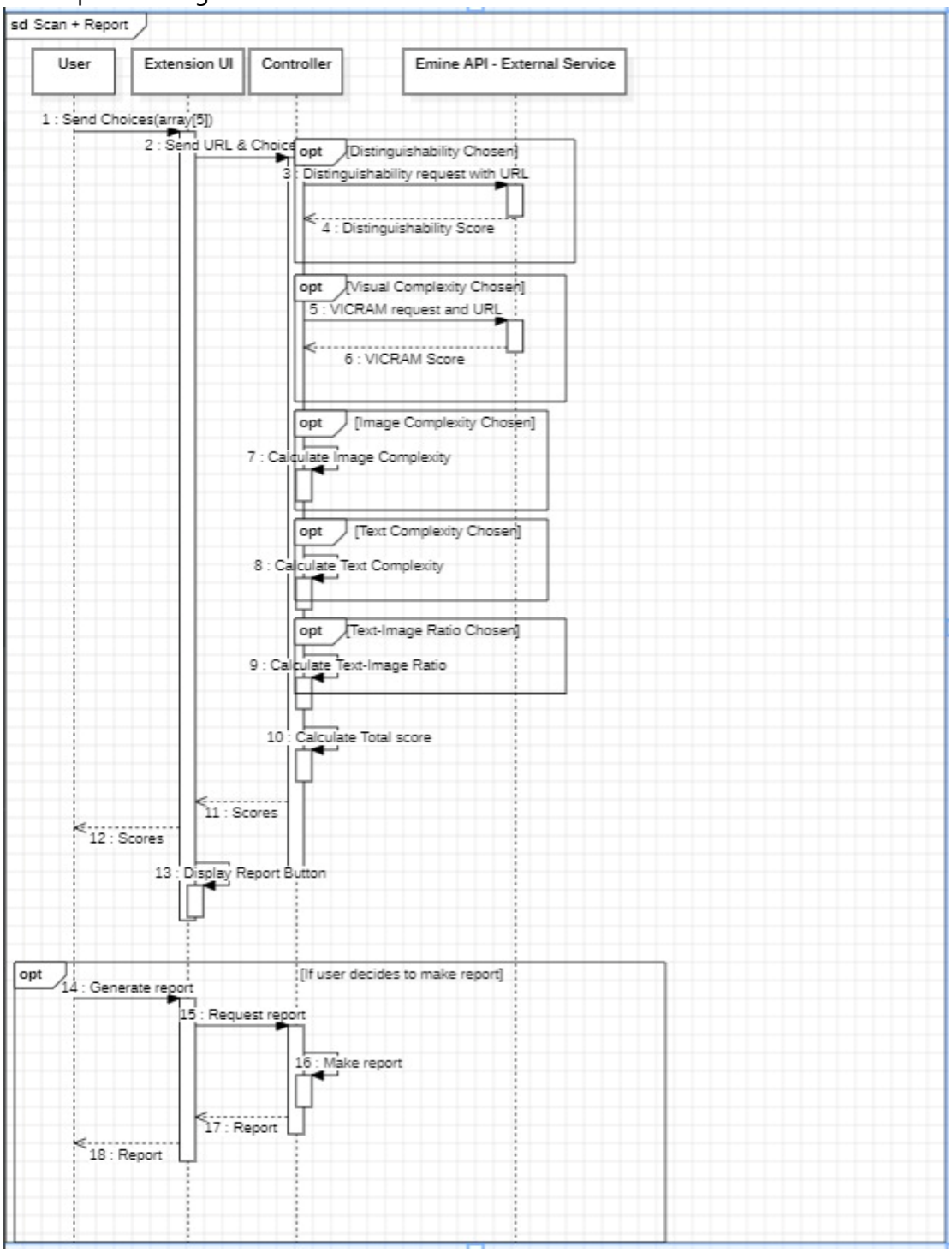

Figure 14: Sequence Diagram of "Scan + Report" Use Case

The figure 14 above shows the Sequence Diagram of our "Scan + Report" use case. There are a total of four lifelines I.e., User, Extension UI (WALA), Controller and eMINE (https://iam.ncc.metu.edu.tr/emine/emine-software/), which is an external source that will compute visual complexity and distinguishability and send it back to our system controller. Firstly, the User sends a message in the form of which complexity metrics they have chosen. The extension together with the choices, passes the URL (automatically fetched) to our controller. The Controller now calculates the other three metrics internally (which are text-image ratio, text complexity and image

complexity) in scores and requests and fetches the VICRAM Score (or in general the visual complexity and distinguishability scores) externally according to the user choices of metric calculations (shown by the opt boxes). All scores are then passed back to our browser extension. Moreover, if the user opts for the generate report choice, the request is sent to our controller which makes the report according to the scores and sends it back to the User.

4.2.3 Data Flow Diagrams

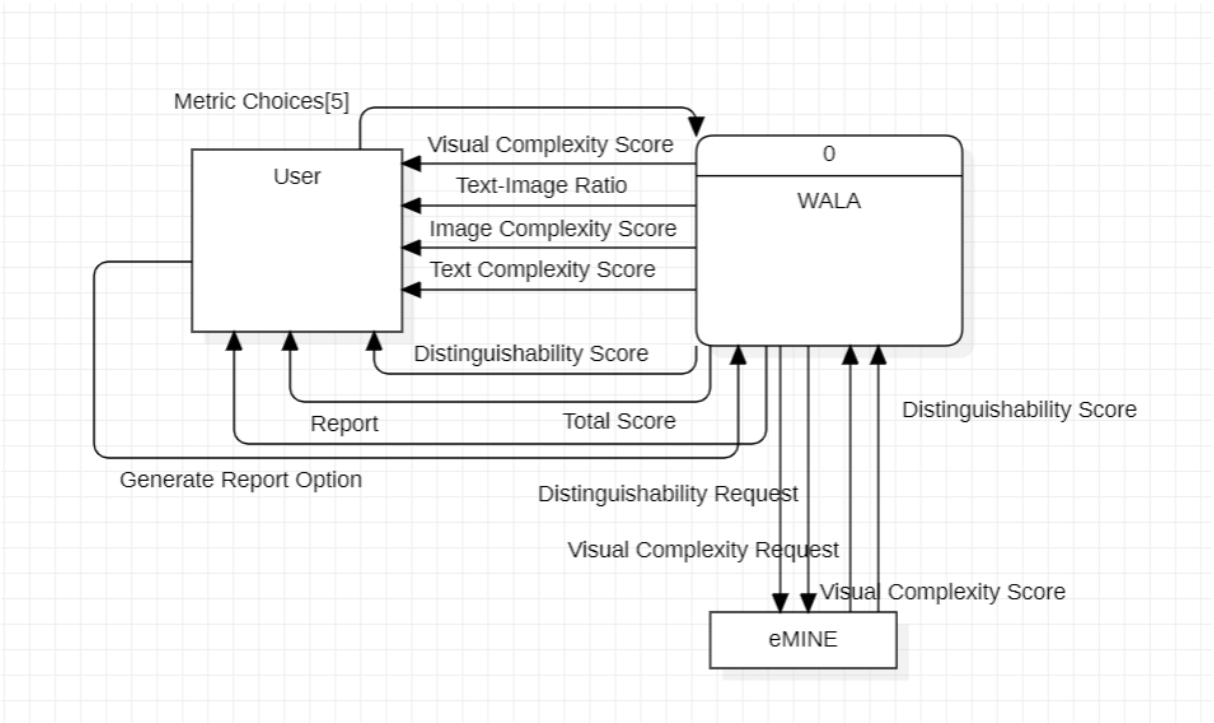

Figure 15: Context-Level Data Flow Diagram

The figure 15 above shows the Context=Level data flow diagram of WALA. Metric Choices is basically the choices the user will select out of 5 metrics. Other data flows are straightforward. eMINE will be our external entity which we will interact with to fetch the visual complexity and distinguishability of the desired webpage. All other scores will be calculated internally as part of the system.

Assumption: The webpage URL is not passed by the user but directly fetched by the browser extension (system) itself that is why it is not passed here as data by the User.

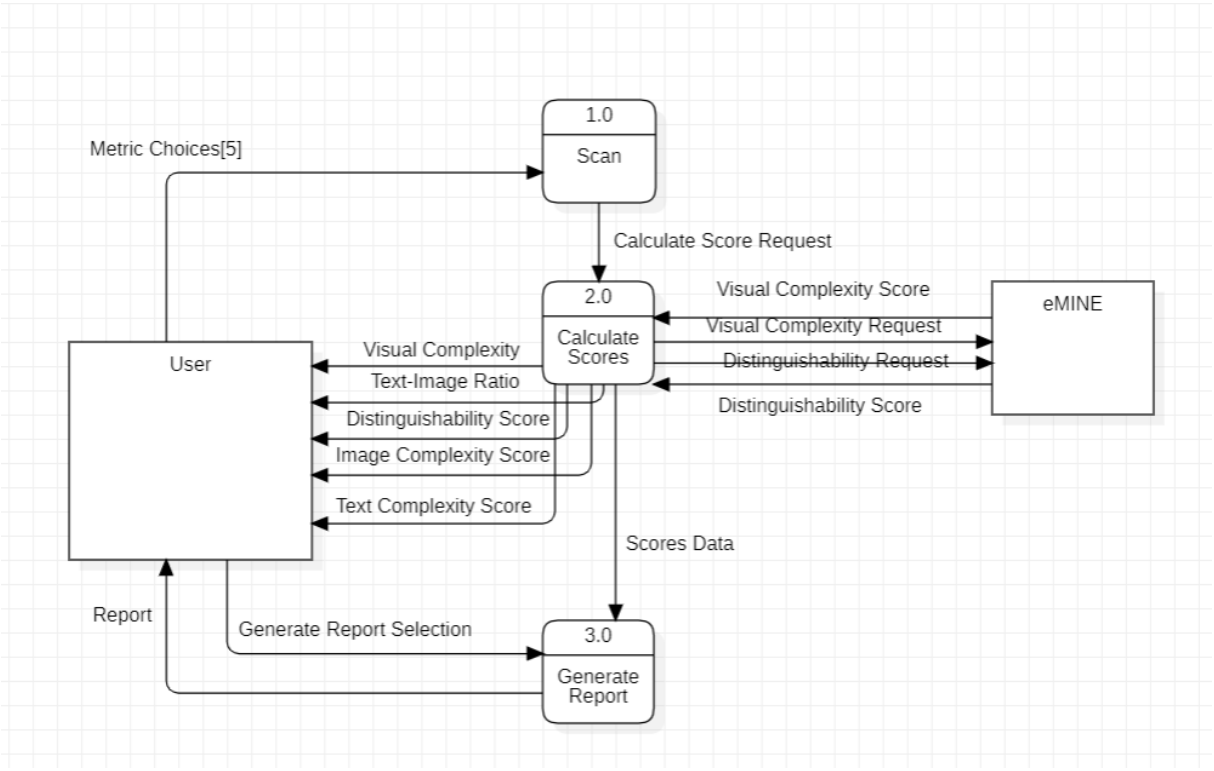

Figure 16: Level-0 Data Flow Diagram

The figure 16 above shows the Level-0 data flow diagram of WALA. In contrast to context-level, the initial WALA process has been broken down to three sub-processes. "Scan" process just takes the user metric choices and then sends the score request to "Calculate Scores" sub-process which both calculates the scores internally (for example image complexity) and externally i.e., visual complexity and distinguishability. If user wishes to generate a report, the scores are sent to the last sub-process "Generate Report" which generates and sends a report back to the User.

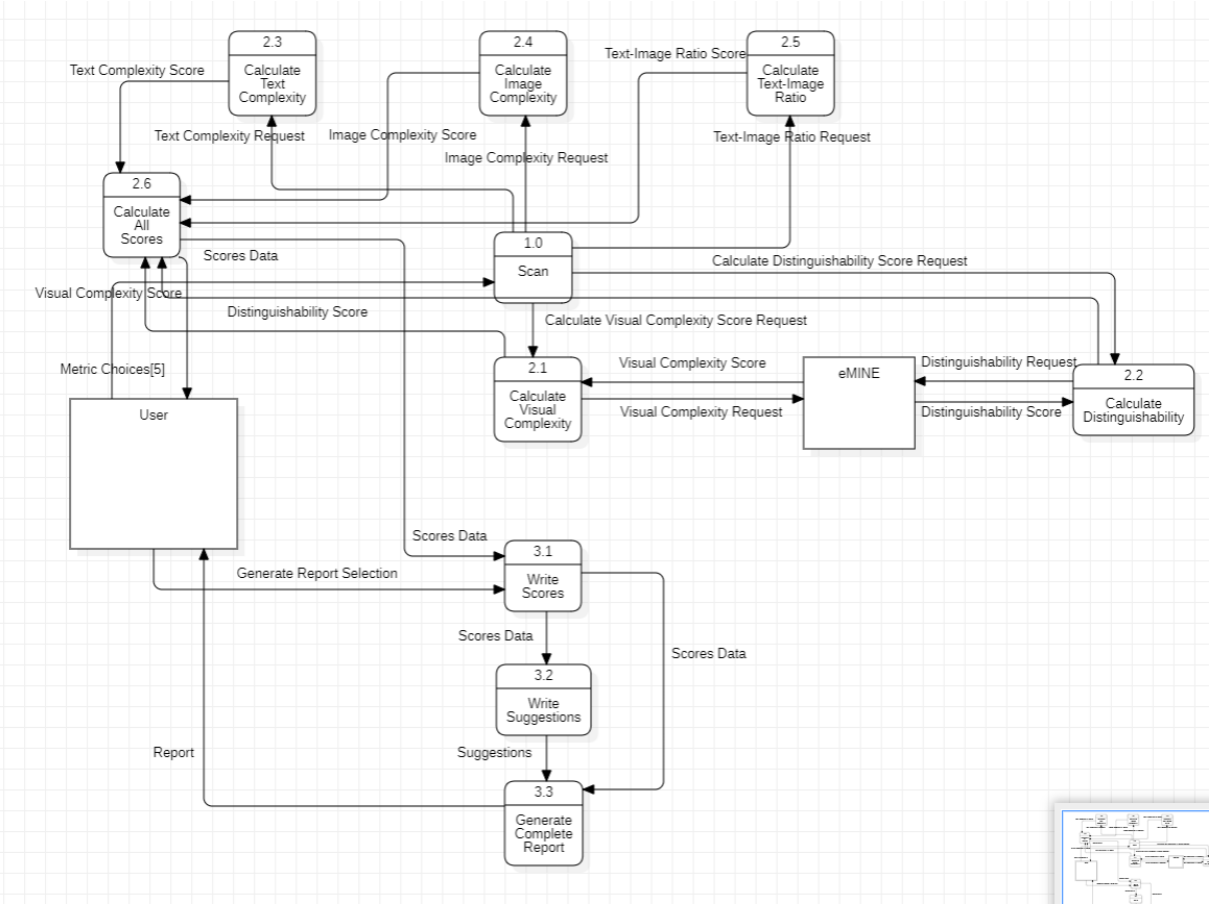

Figure 17: Level-1 Data Flow Diagram

The figure 17 above shows the Level-1 data flow diagram of WALA. In contrast to level 0, the initial Generate Report process has been broken down to three sub-processes. "Write Scores" process just takes the scores calculated and writes them in a report, a "Write Suggestions" process which writes relevant suggestions according to the score data and then the last process "Generate Complete Report" which combines both and makes a complete report and sends it back to the User. Furthermore, Calculate Scores process 2.0 has been broken down into six sub-processes which are Calculate Visual Complexity, Calculate Image Complexity, Calculate Text Complexity, Calculate Text-Image Ratio, Calculate Distinguishability and Calculate All Scores. These sub-processes will be called only if the user selects a metric that is related to the processes, for example user selects only Visual complexity, the scan process will call the Calculate Visual Complexity process and it will calculate it and forward it to Calculate All Scores which will simply display the scores to the User.

4.3 Development & Physical View

The development view shows how the software is decomposed for development. The physical view shows the mapping of software onto hardware.

4.3.1 Component Diagram & Deployment Diagram

***Provide** UML Component Diagram and write its design rationale. **Provide** UML Deployment Diagram by considering both **components and critical artefacts** and write its design rationale.*

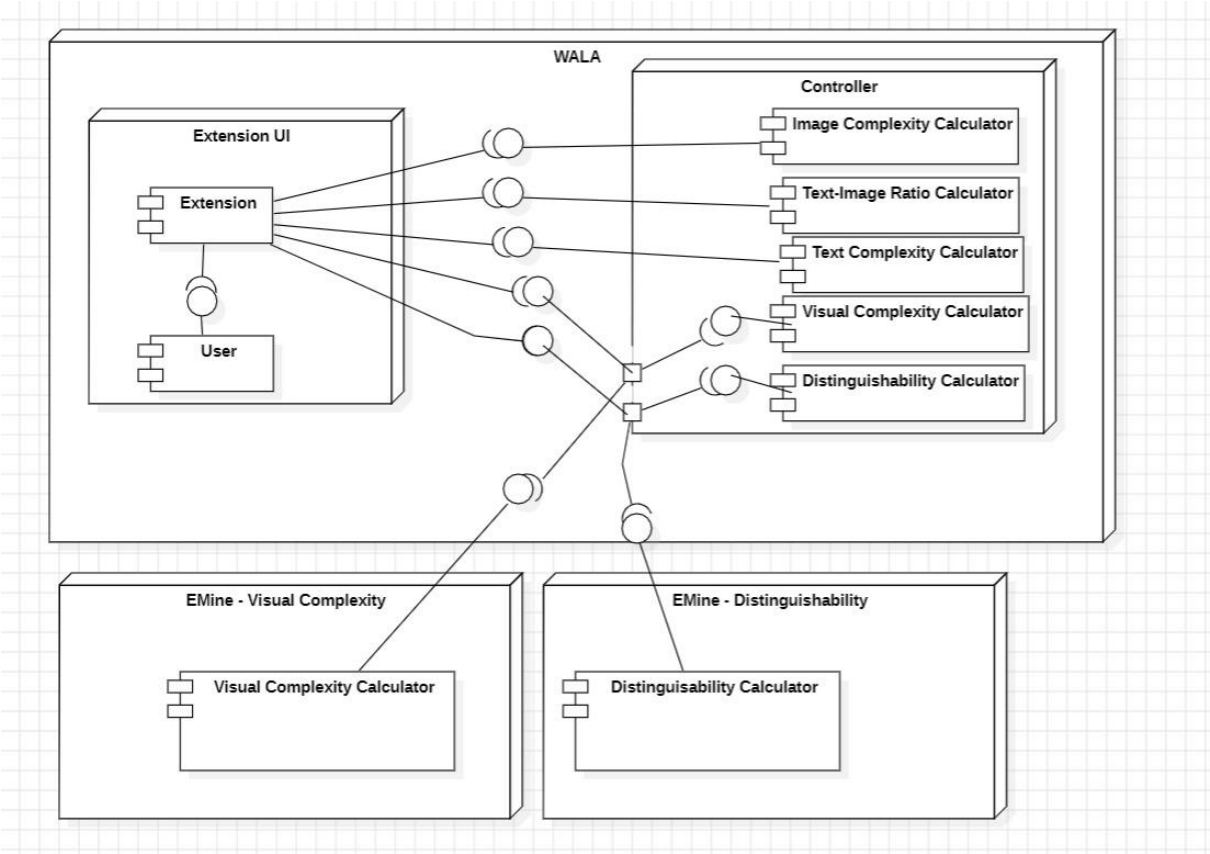

Figure 18: Component & Deployment Diagrams

Figure 18 above shows us both the Component and Deployment Diagrams of WALA. There are four main deployments, WALA, Controller and eMINE (Visual Complexity & Distinguishability) which is an external service. WALA has two inner services Extension UI, which has two components Extension and User. The other service is our controller, which has five components Image Complexity Calculator, Text-Image Ratio Calculator and Text Complexity Calculator, and Visual Complexity and Distinguishability Calculators. The controller provides services to the Extension UI while it takes services from eMINE.

5 Project Scheduling

5.1 Milestones and Tasks

1. Research
2. SRS
3. SDD
4. Base Extension
5. Testing eMINE
6. Developing metric calculations
7. Integrating functionality
8. Report generation
9. Deployment
10. Final Report

Task ID	Task	Duration	Task Dependency	Team members
1.	**Research 5 metric**	**2 weeks**		**All members**
2.	SRS diagrams	1 week	T1	All members
3.	Filling up SRS template	1 week	T1	Shayan and Ali
4.	SDD diagrams	1 week	T1	All members
5.	Filling up SDD template	1 week	T1	Nurlan and Batuhan
6.	Learning how to make extension	1 week		All members
7.	**Creating an extension**	**2 weeks**	**T2 and T6**	**All members**
8.	Learning API	1 week		Ali and Shayan
9.	**Testing EMINE API**	**1 week**	**T8**	**Ali and Shayan**
10.	Developing Text Complexity algorithm	2 weeks	T1	Batuhan
11.	Developing Image complexity algorithm	2 weeks	T1	Shayan
12.	Developing Text-Image ratio algorithm	2 weeks	T1	Nurlan and Ali
13.	**Testing the new metrics calculations**	**1 week**	**T10, T11, and T12**	**All members**
14.	Developing a Controller	1 week	T3 and T5	Nurlan
15.	**Integrating all the functionality in the Controller**	**1 weeks**	**T13 and T19**	**All members**
16.	Developing a report generator	1 week	T1, T9 and T13	Shayan and Batuhan
17.	**Integrating the report generation in the controller**	**1 week**	**T16 and T14**	**Shayan and Nurlan**
18.	Testing	1 week	T17	All members

19.	**Formatting the Final Report**	**1 week**	**T3 and T5**	**Batuhan and Ali**

5.2 Gantt Chart

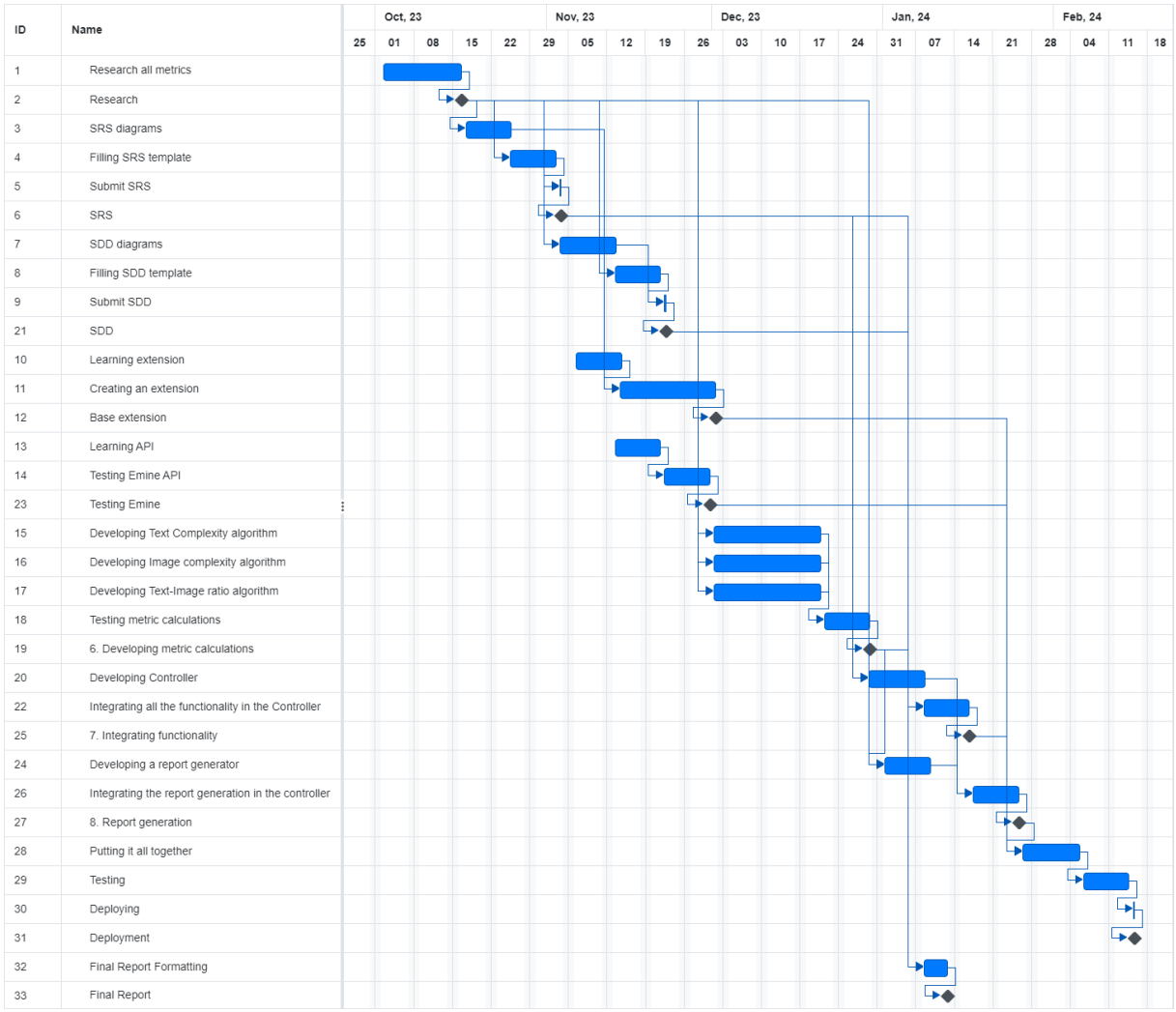

Figure 19: Gantt Chart

Figure 19 shows the Gantt Chart of our project for this semester.

6 Conclusion

Conclusion and Critical Evaluation:

Throughout the semester, our team successfully integrated external APIs, specifically eMINE, to calculate visual complexity and distinguishability. The decision to keep the code modular has proven beneficial, allowing for scalability and ease of incorporating additional metrics in the future. The internal calculations for image complexity, text complexity, and image-text ratio have contributed to a comprehensive evaluation of webpages.

One notable achievement is the creation of a web browser extension, making WALA a user-friendly tool that can be easily accessed and utilized by individuals with autism. By providing a nuanced analysis of webpage elements, WALA has the potential to significantly improve the online experience for its target audience.

Retrospective of the Semester:

Reflecting on the semester, we encountered both challenges and successes. The collaborative efforts of the team resulted in the successful development of a functional tool. Challenges included navigating the integration of external APIs, debugging complex calculations, and ensuring the tool's overall usability. However, these challenges provided valuable learning experiences, enhancing our problem-solving and collaborative skills.

User feedback and iterative testing played a crucial role in refining the tool. Regular feedback sessions and continuous improvement cycles allowed us to address usability issues and enhance the accuracy of accessibility assessments. The iterative development approach ensured that WALA evolved to meet the specific needs of its users.

Future Work:

Looking ahead to the next semester, there are exciting prospects for WALA. The modular structure of the code allows for the seamless addition of new metrics. We envision expanding the tool's capabilities by incorporating additional accessibility metrics and refining existing ones. This continuous improvement will contribute to WALA's effectiveness in providing a comprehensive accessibility assessment.

Furthermore, considering the positive impact WALA could have, we plan to explore avenues for sharing the tool with a broader audience. Publishing WALA in online forums that offer various accessibility tools would increase its visibility and accessibility to individuals and organizations interested in enhancing web accessibility.

In conclusion, WALA represents a significant step toward addressing web accessibility challenges for individuals with autism. The journey of its development has equipped us with valuable insights, and we look forward to further refining and expanding WALA in the upcoming semester. Our commitment to creating a more inclusive digital environment remains steadfast, and we are excited about the possibilities that lie ahead.

7 References

1. Eraslan, S., Yesilada, Y., Yaneva, V., Ha, L. A. "Keep it simple!": an eye-tracking study for exploring complexity and distinguishability of web pages for people with autism. Univ Access Inf Soc 20, 69–84 (2021). https://doi.org/10.1007/s10209-020-00708-9
2. *Uxcel - Where design careers are built*. (n.d.). Uxcel. https://app.uxcel.com/courses/design-accessibility/designing-for-austistic-spectrum-disorder-796
3. Yaneva, V. (2016). Assessing text and web accessibility for people with autism spectrum disorder. https://wlv.openrepository.com/handle/2436/620390

4. *Bitbucket*. (n.d.). Bitbucket. https://bitbucket.org/metu-emine/role-detection-api/src/develop/
5. https://medium.com/@OPTASY.com/what-are-some-of-the-best-web-accessibility-testing-tools-to-evaluate-your-website-with-69def25a386
6. Ojha, P. K., Ismail, A., & Srinivasan, K. K. (2021). Perusal of readability with focus on web content understandability. *Journal of King Saud University - Computer and Information Sciences*, *33*(1), 1–10. https://doi.org/10.1016/j.jksuci.2018.03.007
7. Michailidou, E., Eraslan, Ş., YeşiLada, Y., & Harper, S. (2021). Automated prediction of visual complexity of web pages: Tools and evaluations. *International Journal of Human-Computer Studies*, *145*, 102523. https://doi.org/10.1016/j.ijhcs.2020.102523
8. Akpinar, M., & YeşiLada, Y. (2013). Vision based page Segmentation Algorithm: Extended and perceived success. In *Lecture Notes in Computer Science* (pp. 238–252). https://doi.org/10.1007/978-3-319-04244-2_22
9. Michailidou, E., Eraslan, Ş., YeşiLada, Y., & Harper, S. (2021b). Automated prediction of visual complexity of web pages: Tools and evaluations. *International Journal of Human-Computer Studies*, *145*, 102523. https://doi.org/10.1016/j.ijhcs.2020.102523
10. Robison, J. E., & Gassner, D. (2023, March 23). There is no epidemic of autism. It's an epidemic of need. STAT. https://www.statnews.com/2023/03/23/autism-epidemic-cdc-numbers/#:~:text=In%20a%20pair%20of%20new,This%20increase%20may%20sound%20scary
11. Yaneva, V. (2016, May 19). Accessible texts for autism: an eye-tracking study. https://wlv.openrepository.com/handle/2436/609866
12. Akpinar, M., & YeşiLada, Y. (2013). Heuristic role detection of visual elements of web pages. In Springer eBooks (pp. 123–131). https://doi.org/10.1007/978-3-642-39200-9_12

8 Appendices

8.1 Acronyms and abbreviations

WALA: Web Accessibility Tool for Autism.

8.2 Glossary

N/A