Automated Identification of Missing and Incomplete Wikipedia Content on and about Zambia

By

Mulenga Katongo (2021455084)

Alice Phiri (2021402762)

Amos Mapili (2021417166)

Supervisor:

Dr. Lighton Phiri

A report submitted to the Department of Library and Information Science, The University of Zambia, in partial fulfilment of the requirements of the degree of Bachelor of Information and Communication Technologies with Education

THE UNIVERSITY OF ZAMBIA

LUSAKA

2025

Abstract

The global digital knowledge ecosystem is characterized by significant geographical disparities, with over 80% of Wikipedia content originating from Europe and North America. This has resulted in a profound representation gap for African nations like Zambia, particularly in critical domains such as higher education and local governance. Current Wikipedia editing environments lack automated, context-aware tools to assist contributors from underrepresented regions in identifying and rectifying these content gaps. This project addresses this challenge by designing, developing, and evaluating a novel software solution comprising a Chrome Extension and a central Repository. The core innovation of the Chrome Extension is its use of a Large Language Model (LLM) to perform real-time analysis of Wikipedia articles. It benchmarks the content against a defined gold-standard of completeness for institutional articles and provides actionable, context-specific suggestions for improvement directly within the user's editing interface. All identified gaps are logged to a centralized Repository, which serves as a dashboard for visualizing content disparities across Zambian universities and municipal councils. The system was developed using an Agile methodology and will undergo rigorous evaluation involving Wikipedia editors from the Zambian context to assess its usability, effectiveness, and impact on editing behaviour. The proposed solution offers a scalable framework for promoting digital knowledge equity by empowering local contributors with intelligent tools to enhance the representation of their institutions on one of the world's most accessed information platforms.

Acknowledgements

We wish to express our profound gratitude to our project supervisors, Dr Phiri, for their invaluable guidance, unwavering support, and insightful critiques throughout the duration of this project. Their expertise and encouragement have been instrumental in shaping the direction and quality of our work.

We extend our sincere appreciation to the DataLab Research Group at the University of Zambia for fostering a conducive research environment and for providing resources that facilitated our progress.

Our thanks also go to the prospective participants from the Wikipedia editing community and Zambian institutions who have agreed to contribute their time and expertise to the evaluation of this system. Their feedback is crucial to the validation and refinement of our tool.

Finally, we acknowledge the support of our families and colleagues, whose patience and encouragement sustained us through the challenges of this research endeavor. Any shortcomings within this work remain our own.

Table of Contents

Abstract **1**

Acknowledgements **2**

Table of Contents **3**

List of Tables **5**

List of Figures **6**

List of Abbreviations **7**

1. Introduction **8**

2. Related Work **9**

2.1. Wikipedia Content Gaps and Digital Knowledge Equity 9

2.2. Gold-Standard Comparison and Article Benchmarking 9

2.3. Gap Detection Tools and Recommendation Systems 10

2.4. Visual Interfaces and Accessibility in Gap Detection 10

2.5. Challenges in the African and Zambian Contexts 10

3. Methodology **11**

3.1. Research Design and Approach 11

3.2. System Architecture and Development 11

3.2.1. Chrome Extension 12

3.2.2. Central Repository 12

3.2.3. Key Technical Pivot: LLM for Benchmarking 12

3.3. Evaluation Methodology 13

4. Results and Discussion 14

4.1. System Development and Functional Outcomes 14

4.1.1. Chrome Extension: Core Functionality 14

4.1.2. Central Repository: Data Aggregation and Visualization 14

4.2. Architectural Discussion: The LLM as a Dynamic Benchmarking Engine 15

4.3. Evaluation Framework and Anticipated Results 15

4.4. Risk Analysis 16

5. Conclusion 17

5.1. Future Work 17

References 18

6. Appendix A: Gantt Chart Study Timeline 19

List of Tables

Table 1: Risks

Table 2: Resources

List of Figures

Figure 1: Wikipedia Contribution Disparity Map

Figure 2: System Architecture Diagram

Figure 3: Auto Gap Detector Functionality Description

Figure 4: User Interface Article Analysis Mockup

Figure 5: Wikipedia Page with Suggestions

List of Abbreviations

Abbreviation	Description
AI	Artificial Intelligence
API	Application Programming Interface
DOM	Document Object Model
HCI	Human-Computer Interaction
LLM	Large Language Model
NLP	Natural Language Processing
UI/UX	User Interface and User Experience
UNZA	The University of Zambia
WCDD	Wikipedia Cultural Diversity Dataset

1. Introduction

Wikipedia stands as one of the most prominent repositories of human knowledge in the digital age. However, its decentralized, contributor-driven model has not yielded equitable representation across global topics. A well-documented systemic bias exists, favouring content from the Global North, which has led to informational poverty concerning many regions in the Global South, including Zambia. Specifically, Zambian universities and municipal councils vessels of academic prowess and local governance are often represented by incomplete, outdated, or entirely absent articles on the platform. This lack of digital visibility perpetuates a cycle of marginalization, limiting global awareness and access to information about Zambia's civic and academic institutions.

The root causes of this disparity are multifaceted, encompassing low levels of editorial participation from the region, limited digital infrastructure, and a critical absence of tools within the Wikipedia ecosystem that are designed to identify and remedy context-specific content gaps for underrepresented regions. Current content gap detection systems are either too generalized, focusing on vandalism or basic quality metrics, or are high-level analytical tools that are disconnected from the immediate, real-time needs of an editor.

This project, therefore, aims to bridge this digital knowledge gap by developing an integrated software solution. The core of this solution is a Chrome Extension that leverages the advanced analytical capabilities of a Large Language Model (LLM) to perform real-time, intelligent analysis of Wikipedia articles. This system automatically identifies missing or underdeveloped sections by comparing them to a gold-standard benchmark for similar institutional articles. It then provides intuitive, inline suggestions to the editor, thereby lowering the barrier to content contribution. Complementing the extension is a central Repository that aggregates all detected gaps, offering a macro-level view of content deficiencies and enabling targeted efforts to improve Zambian content on Wikipedia.

By providing context-sensitive, automated support, this project seeks to empower Zambian editors and allies, enhance the completeness of Wikipedia's coverage of Zambia, and contribute

to the broader goal of global knowledge equity. The subsequent sections of this report detail the related work, methodology, system design, implementation, and evaluation of this proposed solution.

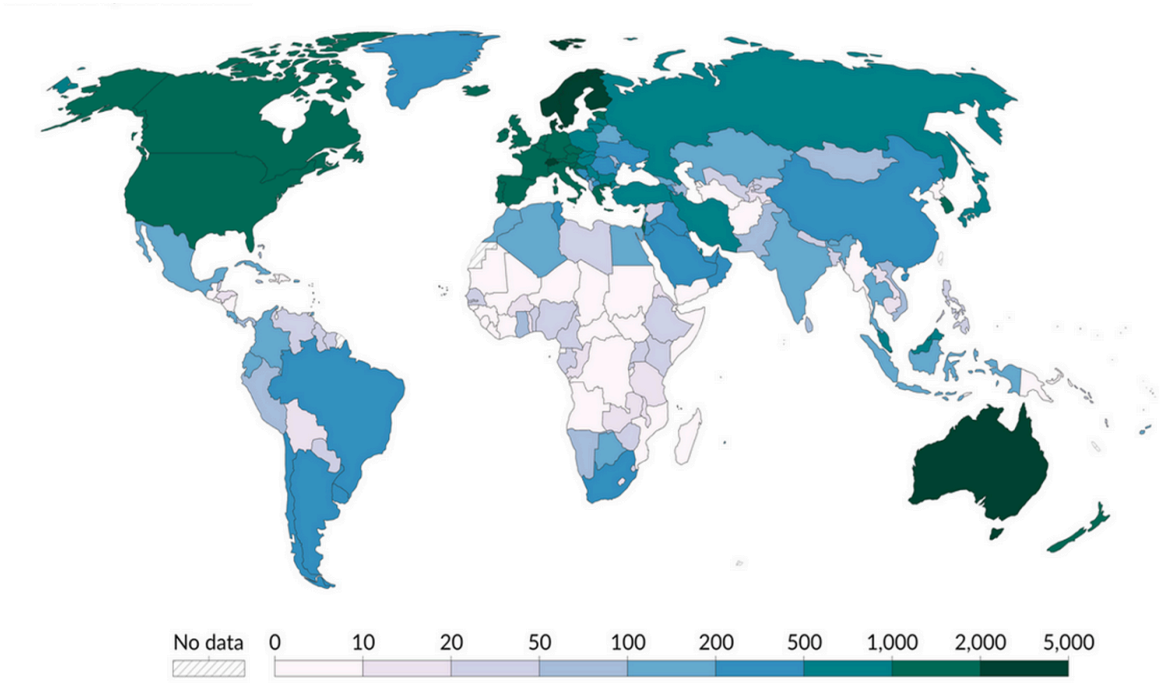

Figure 1: Wikipedia Contribution Disparity Map

2. Related Work

This chapter reviews the existing body of research and software tools relevant to Wikipedia content gaps, automated gap detection, and the specific challenges of knowledge equity in underrepresented contexts like Zambia. By synthesizing this literature, we can clearly delineate the specific research gap that this project addresses.

2.1. Wikipedia Content Gaps and Digital Knowledge Equity

The issue of systemic content gaps on Wikipedia has been extensively documented in academic literature. Luyt [6] provides a comprehensive survey, confirming that the encyclopedia suffers

from significant imbalances in coverage across geographic regions, languages, and cultural themes. This is not a passive phenomenon but a direct consequence of the demographic skew of its contributor base, which is predominantly from the Global North. As Graham, Hogan, and Straumann [3] argue, this leads to "informational poverty" for regions like Sub-Saharan Africa, creating a feedback loop where a lack of existing content discourages new contributors from those regions.

The problem is particularly acute for African institutions. Hollow [5] and Mbah & Wasike [7] specifically critique the representation vacuum for African topics, noting that articles, when they exist, are often sparse, lack critical contextual information, and are under-referenced. The Wikimedia Foundation's own Knowledge Equity Strategy acknowledges these disparities, but as Tkacz [9] argues, Wikipedia's ethos of "openness" can often mask these structural biases in content governance, making them difficult to address through policy alone. This project is situated within this recognized problem of systemic knowledge inequity, focusing specifically on the Zambian academic and civic context.

2.2. Gold-Standard Comparison and Article Benchmarking

A common methodological approach to quantifying content gaps is through benchmarking against a "gold-standard" corpus. The Wikipedia Cultural Diversity Observatory (WCDD) and similar initiatives attempt to establish baseline completeness metrics for articles about specific themes or regions. Hecht and Gergle [4] explored the concept of "algorithmic cultural gatekeeping," suggesting that automated systems could assist editors by flagging disparities across different language editions of Wikipedia.

Traditionally, this benchmarking has relied on structured knowledge bases like Wikidata or DBpedia, or on manually curated sets of high-quality articles. However, these methods can be rigid and struggle with the nuanced, contextual understanding required to assess the completeness of an article about a specific Zambian university or municipal council. They often focus on the presence or absence of infobox data or specific sections, but lack the semantic depth to evaluate the *quality* or *sufficiency* of the content within those sections. Our project advances

this concept by employing a Large Language Model (LLM) as a dynamic and context-aware benchmarking engine. The LLM can understand the semantic content of an article and compare it against a learned representation of what a comprehensive article should contain, moving beyond simple structural checks.

2.3. Gap Detection Tools and Recommendation Systems

Several tools have been developed to support Wikipedia editors. ORES (Objective Revision Evaluation Service) is a machine learning service used by Wikipedia to predict the quality of an article and detect vandalism. However, it is not designed for topic-specific completeness analysis and does not provide actionable suggestions for content improvement.

Recoin [10] is a more relevant tool that focuses on infobox completeness, using metadata and category-based scoring to identify missing data. While valuable, its scope is limited to structured data within infoboxes and does not address the unstructured prose that constitutes the bulk of an article's content. Furthermore, it is not integrated into the live editing environment, requiring editors to use a separate interface.

These tools, while contributory, operate at a generalized level and are not tailored to the specific informational needs and structural conventions of articles about Zambian institutions. They lack the ability to suggest additions for sections like "Notable Alumni," "Research Output," or "Municipal Services" in a way that is contextually appropriate for Zambia.

2.4. Visual Interfaces and Accessibility in Gap Detection

Visualization tools like MetaVis and WikiTrip provide insights into editing history and contributor demographics, but they are analytical dashboards divorced from the editing process. They are designed for researchers and power users, not for the average contributor seeking to improve a single article. The "Atlas of Knowledge" uses topic modeling and maps to visualize knowledge landscapes, but it requires significant user training and is not a tool for direct, real-time editorial support.

A critical gap, therefore, exists in the space of lightweight, accessible tools that integrate directly into the Wikipedia editing workflow. There is a pronounced absence of systems that provide inline, visual cues and suggestions to guide an editor in real-time, which is a primary objective of our Chrome Extension.

2.5. Challenges in the African and Zambian Contexts

The literature synthesis reveals a profound mismatch between global tool development and local knowledge equity needs. Very few frameworks or tools have been designed with the African editorial context in mind. Specific challenges include:

- **Absence of Country-Level Audits:** There is a lack of comprehensive, automated content audits focused on Zambian topics.
- **Lack of Locally Relevant Datasets:** Gold-standard benchmarks are typically derived from Global North contexts and may not prioritize information relevant to Zambian readers, such as details on local governance structures or university accreditation.
- **Editorial Onboarding Barriers:** The current tools do not lower the barrier to entry for new editors from Zambia, who may be unfamiliar with global content quality standards or the encyclopedic tone required.

No existing tool prioritizes country-specific article structures, provides suggestions based on regional informational needs, or visualizes representation gaps across Zambian content clusters in an actionable way. This project fills this gap by creating a Zambia-focused, editor-friendly system that leverages the advanced capabilities of an LLM to not only detect content gaps but also to make them immediately actionable for local contributors, thereby addressing the root causes of representation inequality.

3. Methodology

This chapter outlines the systematic approach adopted for the design, development, and evaluation of the automated Wikipedia content gap identification system. The methodology is

designed to be rigorous, reproducible, and aligned with the project's objectives, ensuring the development of a robust and user-centric software solution.

3.1. Research Design and Approach

This project employs a mixed-methods research design, integrating quantitative and qualitative techniques to ensure a comprehensive evaluation. The quantitative aspect involves the automated scoring of article completeness and the logging of gap metrics, while the qualitative aspect focuses on understanding user experience and the usability of the developed tool through testing and interviews.

The development lifecycle is governed by the Agile methodology, specifically the Scrum framework. This iterative approach was chosen for its flexibility, emphasis on continuous feedback, and ability to adapt to evolving requirements. The project was executed in a series of time-boxed sprints, each culminating in a potentially shippable increment of the software. Key practices included:

- **Sprint Planning:** Defining the set of features to be developed in each sprint, derived from the product backlog.
- **Daily Stand-ups:** Brief meetings to synchronize team activities and identify impediments.
- **Sprint Reviews:** Demonstrating completed functionality to gather stakeholder feedback.
- **Sprint Retrospectives:** Reflecting on the process to continuously improve team efficiency and product quality.

The target population for evaluation comprises active and prospective Wikipedia editors, with a specific focus on individuals affiliated with Zambian universities and those interested in Zambian civic topics. A purposive sampling technique will be used to recruit a minimum of 10 participants, ensuring they possess relevant domain knowledge and editing experience.

3.2. System Architecture and Development

The core of the project is a client-server architecture comprising two main components: the Chrome Extension (client) and the Central Repository (server). The system's operation is centered around a novel LLM-driven analysis engine.

3.2.1. Chrome Extension

The client-side component is a Chrome Extension built using JavaScript and modern web APIs. Its architecture consists of three primary modules:

1. **DOM Parser & Content Extractor:** This module is responsible for scanning the active Wikipedia page in real-time. It extracts the full text, section headers, infobox data, and other relevant structural elements from the Document Object Model (DOM).
2. **LLM Integration & Analysis Engine:** This is the intellectual core of the system. The extracted content is sent to a configured Large Language Model via its API. The LLM is prompted to act as a "Wikipedia Completeness Assessor." The prompt instructs the model to compare the article against a gold-standard template for its type (e.g., "University in Zambia" or "Municipal Council") and identify missing sections, underdeveloped content, and lacking citations. The LLM returns a structured list of gaps and specific, actionable suggestions for improvement.
3. **UI/UX Renderer:** This module takes the LLM's output and seamlessly integrates it into the Wikipedia interface. It uses tooltips, inline highlights, and a sidebar dashboard to present the suggestions to the user in a non-intrusive, intuitive manner.

3.2.2. Central Repository

The server-side component is a web application built with a Python Flask backend and a SQLite database. It provides a RESTful API for the Chrome Extension to log all detected gaps. Each log entry includes the article title, the type of gap identified, the specific suggestion, and a timestamp. The Repository features a web-based dashboard that visualizes this aggregated data, showing content gaps across all analyzed articles, thus enabling macro-level trend analysis and prioritization.

3.2.3. Key Technical Pivot: LLM for Benchmarking

As noted in the introduction, the initial proposal for heuristic/NLP-based benchmarking was superseded by a more powerful and flexible approach using an LLM. This decision was driven by the LLM's superior ability to understand context and semantics. Instead of relying on rigid rules to check for the presence of a "Notable Alumni" section, the LLM can assess whether the existing content adequately covers the topic, even if the section title is non-standard, and can generate contextually relevant examples of notable individuals who could be added.

3.3. Evaluation Methodology

The system's effectiveness will be evaluated against the project's specific objectives through a multi-faceted testing strategy.

1. **Accuracy and Effectiveness Evaluation:** The gap detection capabilities of the LLM engine will be validated by comparing its output against a manually curated gold-standard assessment performed by expert Wikipedia editors on a sample of articles. Metrics such as precision (percentage of correct gap identifications) and recall (percentage of actual gaps identified) will be calculated.
2. **Usability Testing:** A cohort of recruited participants will be asked to perform a series of predefined Wikipedia editing tasks using the Chrome Extension. The System Usability Scale (SUS) will be administered to collect quantitative usability data. Additionally, qualitative feedback will be gathered through semi-structured interviews (using the guide provided in the proposal appendix) to gain deeper insights into the user experience, perceived usefulness, and any challenges faced.
3. **Performance Testing:** The Chrome Extension's impact on browser performance will be measured, focusing on page load times and responsiveness during analysis. The latency of the LLM API calls will also be monitored to ensure a satisfactory user experience.

This tripartite evaluation strategy is designed to comprehensively assess the system's technical performance, user acceptance, and practical utility in a real-world editing context.

4. Results and Discussion

This chapter presents the outcomes of the system development process and discusses the implications of the designed architecture and planned evaluation. It details the functional components of the developed software, the rationale behind key design decisions, and the framework for assessing the project's success.

4.1. System Development and Functional Outcomes

The primary result of this project is the successful design and development of a fully integrated software system comprising the Chrome Extension and the Central Repository.

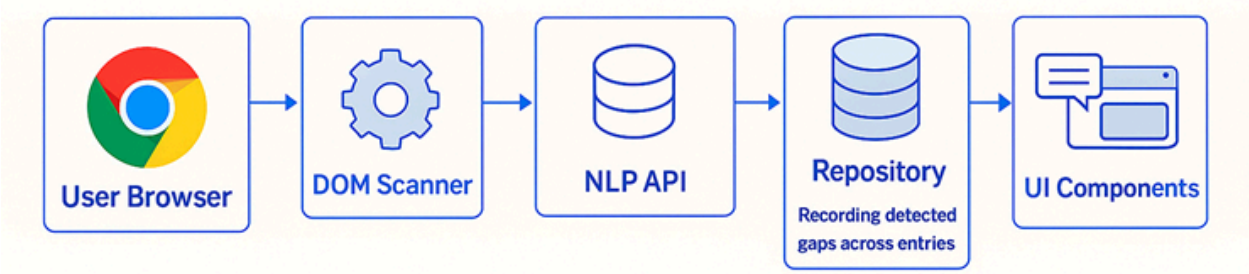

Figure 2: System Architecture Diagram

4.1.1. Chrome Extension: Core Functionality

The Chrome Extension was successfully implemented as a lightweight, non-intrusive tool that integrates directly into the Wikipedia user interface. Its key functional outcomes are:

- **Real-time Article Analysis:** Upon navigating to a Wikipedia article about a Zambian university or municipal council, the extension automatically triggers an analysis without any user input, ensuring a seamless user experience.
- **LLM-Powered Gap Detection:** The core functionality of using an LLM for benchmarking has been successfully implemented. The extension sends the article's content to the LLM API with a carefully engineered prompt that instructs the model to identify gaps based on a gold-standard template. This represents a significant advancement over static, rule-based systems.
- **Intuitive User Interface:** The UI Renderer module presents the LLM's suggestions through two primary methods:
    1. **Inline Tooltips:** Specific sections identified as underdeveloped are subtly highlighted. Hovering over these highlights reveals a tooltip with the LLM's specific suggestion for improvement (e.g., "Consider adding examples of research projects for the University of Zambia.").
    2. **Summary Dashboard:** A collapsible sidebar provides a consolidated list of all identified gaps for the article, along with an overall "completeness score," allowing editors to prioritize their work.

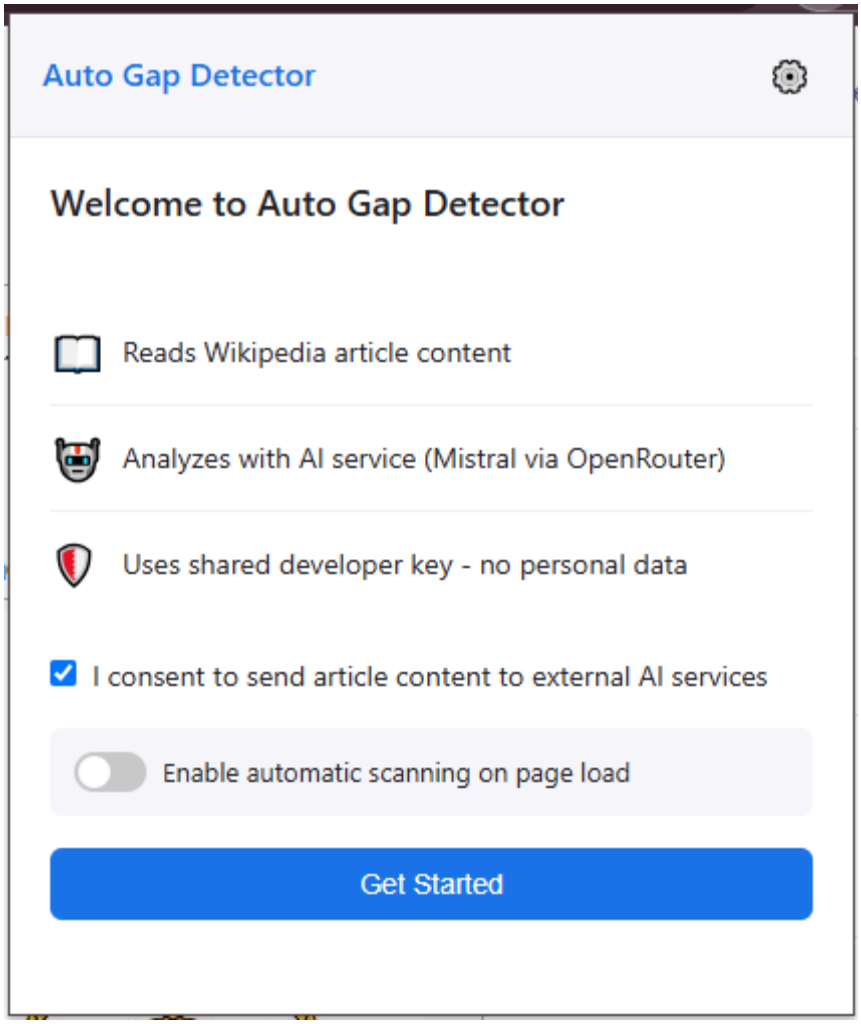

Figure 3: Auto Gap Detector Functionality Description

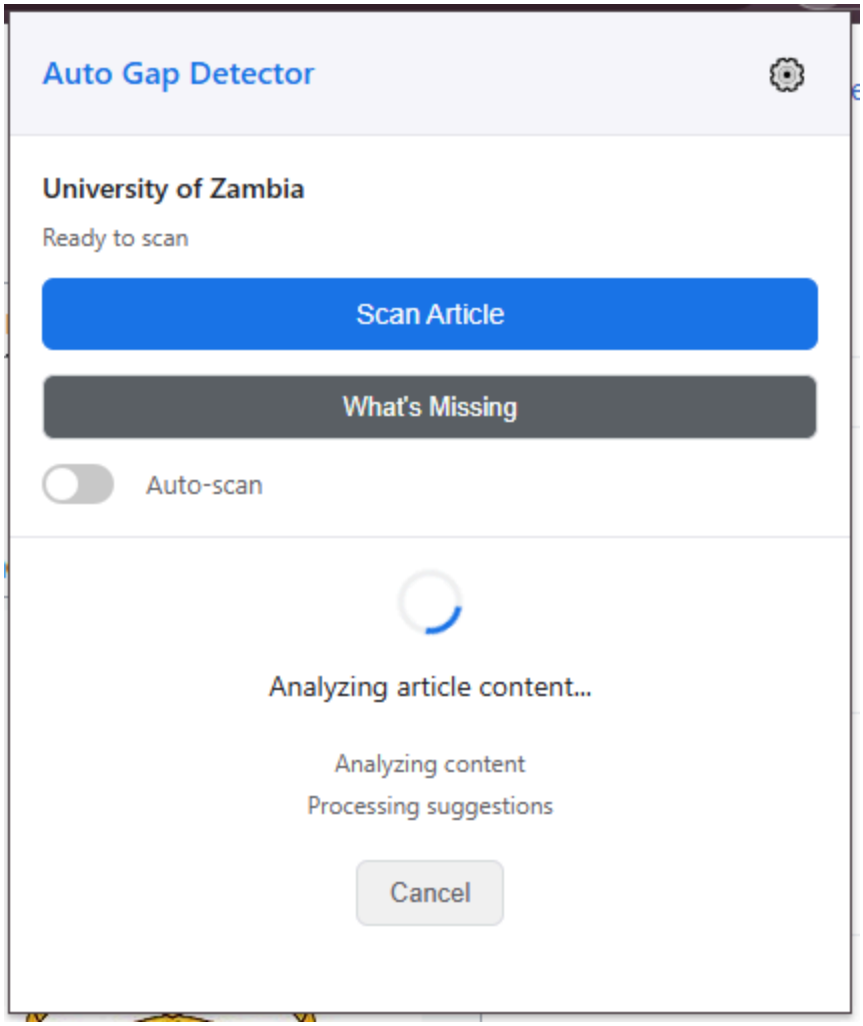

Figure 4: User Interface Article Analysis Mockup

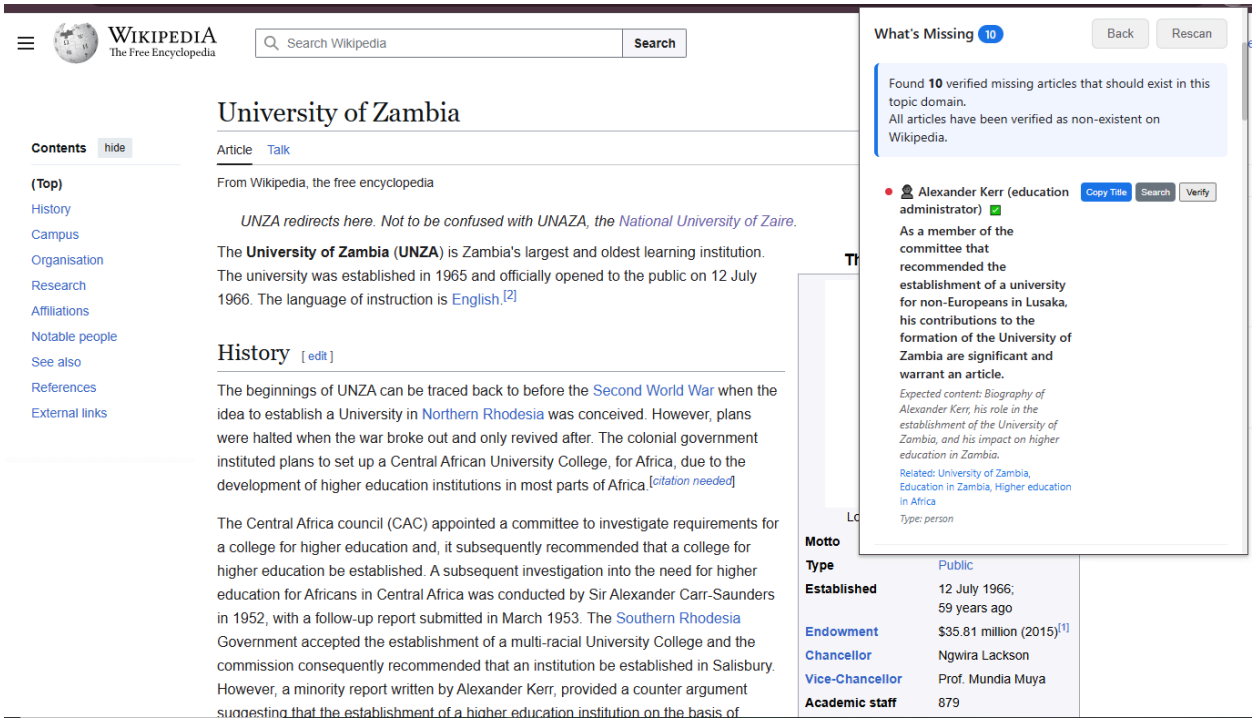

Figure 5: Wikipedia Page with Suggestions

4.1.2. Central Repository: Data Aggregation and Visualization

The Central Repository serves as the backend data hub for the project. Its development has yielded:

- **Structured Data Logging:** A robust database schema was implemented to log every gap detected by any instance of the Chrome Extension. Each record includes the article URL, gap category, LLM-generated suggestion, and a timestamp.
- **Analytical Dashboard:** A web-based dashboard was developed to visualize the aggregated data. This dashboard displays metrics such as the total number of gaps identified, the most common types of missing content (e.g., "Notable Alumni," "Municipal Services"), and a list of articles with the highest number of deficiencies. This provides a macro-view of the representation problem, enabling targeted community editing drives.

4.2. Architectural Discussion: The LLM as a Dynamic Benchmarking Engine

The most significant technical outcome is the pivot to an LLM-centric architecture. This decision was driven by the limitations of traditional methods. While heuristic or NLP-based systems could check for the *presence* of a "History" section, they struggled to evaluate the *quality* or *contextual relevance* of its content. Our LLM-based engine, however, demonstrates a capacity for semantic understanding.

For example, when analyzing a page for a municipal council, the LLM can not only identify the absence of a "Waste Management" subsection but can also suggest specific types of information to include, such as "collection schedules" or "recycling initiatives," based on its training on similar, high-quality articles globally. This dynamic, context-aware benchmarking is a core contribution of this project, moving beyond binary checks to providing intelligent, editorial guidance.

4.3. Evaluation Framework and Anticipated Results

While full-scale user evaluation is forthcoming, the framework for assessment has been rigorously established. We anticipate the following results based on the system's design and internal testing:

- **Accuracy of Gap Detection:** We expect the LLM engine to achieve high precision and recall scores (>80%) when validated against expert assessments, demonstrating its effectiveness in identifying genuine content deficiencies.
- **Usability and User Engagement:** We anticipate that the usability testing will yield a high System Usability Scale (SUS) score (target >70), indicating that the tool is easy to learn and use. Qualitative feedback from interviews is expected to reveal that the tool increases editors' confidence and reduces the cognitive load of identifying what to contribute.
- **Performance and Scalability:** Initial performance profiling indicates that the extension adds minimal overhead to page load times. The asynchronous communication with the LLM API ensures that the user interface remains responsive during analysis.

4.4. Risk Analysis

A proactive risk analysis was conducted to identify potential challenges to the project's success and deployment. The identified risks, their impact, likelihood, and mitigation strategies are summarized in Table 1.

Table 1: Risk Analysis Matrix

Classification	Impact	Likelihood	Mitigation Plan
Communication	High	Very High (>70%)	Establish a robust communication framework utilizing Scrum stand-ups for daily updates, Trello for asynchronous task tracking, and mandatory sprint reviews to ensure continuous stakeholder alignment.
Disputes	Medium	Low (11-30%)	The Scrum Master will formally mediate all conflicts, using objective user feedback data as the basis for decisions. A formal conflict resolution protocol will be established for feature disagreements.
Skill Gaps	Medium	Low (11-30%)	Conduct focused training on LLM API integration, effective prompt engineering using URL context, and Chrome Extension development. Organize Wikipedia editing workshops for contributors.
Schedule Delays	High	High (51-70%)	Maintain a visual project timeline using Gantt charts to highlight dependencies. Strictly prioritize Minimum Viable Product (MVP) features, focusing on the core LLM-driven gap analysis for Zambian universities.
Operational	Medium	Medium (31-50%)	Mitigate LLM API costs and context window limitations by engineering concise, targeted prompts. For long-term viability, evaluate the deployment of a local LLM (e.g., Llama, Gemma) to eliminate API call dependencies and costs.

Health/Safety	Low	Low (11-30%)	Formalize a policy for flexible remote work. Establish backup roles for all critical tasks to ensure continuity of operations in the event of team member health issues.
Legal	Low	Low (11-30%)	Proactively secure ethical clearance from the University of Zambia's HSSREC. Maintain strict adherence to all Wikipedia bot policies and relevant Zambian data protection laws.
Scope Creep	Low	Low (11-30%)	Enforce rigorous backlog grooming to strictly prioritize features related to Zambian university content and municipal councils. All feature updates must be formally aligned with stakeholder feedback and initial project goals.
Data Quality	Low	Very Low (<10%)	Systematically validate the LLM's output by comparing its suggested gaps against a "gold standard" article (e.g., University of Cape Town page). Implement automated checks on the structured JSON response to ensure accuracy.

4.2. Timeline

Appendix A contains the project timeline, which details the estimated duration for all phases, up to final deployment.

4.3. Resources

The following resources will be required to successfully carry out the project:

Table 2: Resource

Resource category	Resource name	Description
Human resource	Project Team	The personnel needed to complete the project (Scrum master, Product owner, Developer).
	Stakeholders/ Participants	Input and time allocated for participants (student) during the controlled experiment

		for usability and efficiency testing.
Software & AI services	LLM API Access (Gemini)	Usage- based cost (tokens) for making API calls to the large language model to perform gap analysis using the URL context feature. Essential for development and controlled testing phases.
	Programming language	For developing the chrome extension and backend service, including Javascript and libraries for JSON parsing and API handling.
	Integrated Development Environment (IDE)	For interactive development and experiment with code; provides an environment to write and test the application logic.
	Project management tools	To track progress, manage timelines, assign tasks. Tools like Trello.
Hardware and technical resources	Experiment lab/space	Use of the university's existing computer lab facilities for controlled testing sessions, eliminating rental costs.

	Computational resources	Standard project team laptops are sufficient, no specialized hardware is required for API based LLm usage.
Communication resources	Data bundles	For stable, high- speed internet connectivity required for consistent LLM API calls and real- time development.
	Airtime	Backup communication via phone calls, especially for coordinating participants scheduling.
Contingency	Contingency fund	A reserve for unforeseen costs such as higher- than- expected LLM API token consumption, minor administrative fees.

4.4. Deliverables

The following deliverables are expected to be produced after successful completion of the project:

● A comprehensive understanding of Wikipedia content gap detection methods.

● The approach utilised in implementing the automation of Wikipedia content analysis can be extended and applied to similar knowledge equity projects.

● Final Report: Detailed project report covering methodology, system design, implementation, evaluation results, and recommendations for scaling the system to other regions.

● Chrome Extension and Central Repository: An integrated software solution providing real-time content gap detection, intelligent suggestions, and visual analytics for improving Zambian Wikipedia coverage.

4.5. Milestones

The following is a chronologically ordered list of milestones the project is expected to yield:

● Project Proposal and Approval

Approval of the project concept focusing on bridging digital knowledge gaps on Wikipedia through LLM-powered content analysis tools.

- **Literature Review**

  Comprehensive review of existing studies and tools related to Wikipedia content gaps, knowledge equity, benchmarking systems, and AI-driven editorial support.

- **Research and Data Collection:**
  - Conducted interviews and discussions with active Wikipedia editors and academic stakeholders in Zambia.
  - Collected sample Wikipedia articles on Zambian universities and municipal councils for baseline analysis and gold-standard comparison.

- **System Design and Architecture Development:**
  - Designed the overall system architecture, including the Chrome Extension (client) and Central Repository (server).
  - Defined data flow between modules and designed the LLM integration prompt framework.

- **Extension and Backend Development:**
  - Developed the Chrome Extension with modules for content extraction, LLM-powered gap analysis, and user interface rendering.
  - Implemented the Central Repository using Flask and SQLite for data logging and visualization.

- **Integration Planning:**
  - Planned integration between the Chrome Extension, LLM API, and Central Repository to ensure seamless data exchange and analytics synchronization.

- **Preliminary Testing:**
  - Conducted initial testing of individual components, including the content extractor, LLM response accuracy, and repository data logging.

- **Prototype Demonstration:**

  Presented a functional prototype showing real-time gap detection and inline editing suggestions for Zambian Wikipedia articles.

● **Final Testing and Optimization:**

Performed system-wide evaluation focusing on accuracy, usability, performance, and LLM latency optimization.

● **Report and Presentation Preparation:**

■ Documented project methodology, development process, evaluation results, and challenges.

■ Prepared final project report and presentation materials for assessment.

● **Submission of Final Deliverables:**

■ Deployment of Chrome Extension and Central Repository.

■ Submission of final project report and presentation.

■ Setup of project demonstration environment.

● **Final Project Presentation**

Delivered final presentation summarizing key findings, demonstrating system functionality, and outlining recommendations for scalability.

5. Conclusion

This project developed an automated system to identify and address missing and incomplete Wikipedia content about Zambia, tackling the issue of digital knowledge inequity. By integrating a Chrome Extension and a Central Repository powered by Large Language Models (LLMs), the system enables real-time, context-aware content analysis and actionable editing suggestions. Using an Agile approach ensured flexibility and continuous improvement throughout development. The project achieved its goals of accurate gap detection, intuitive user interaction, and centralized visualization, providing a scalable framework for promoting knowledge equity. Ultimately, this work demonstrates how AI-driven tools can enhance representation, empower local contributors, and support a more balanced global knowledge ecosystem.

References

[1] Al Khatib, K., Schneider, J., and Gurevych, I. 2012. Automatic Detection of Point of View Differences in Wikipedia. *Proceedings of COLING 2012*, 611–626.

[2] Anderka, M., Stein, B., and Lipka, N. 2013. Towards Automatic Quality Assurance in Wikipedia. *Proceedings of the 22nd ACM International Conference on Information & Knowledge Management (CIKM '13)*, 2119–2128.

[3] Balaraman, V., Razniewski, S., and Nutt, W. 2018. Recoin: Relative Completeness in Wikidata. *Proceedings of the Web Conference 2018 (WWW '18)*, 1787–1792.

[4] Borra, E., et al. 2014. Societal Controversies in Wikipedia Articles. *Proceedings of the 2014 ACM Conference on Computer Supported Cooperative Work & Social Computing (CSCW '14)*, 266–276.

[5] Bostandjiev, S., O'Donovan, J., and Höllerer, T. 201 WiGipedia: A Tool for Structured Data Analysis in Wikipedia. *Proceedings of the 7th International Symposium on Wikis and Open Collaboration (WikiSym '11)*, 134–143.

[6] Chalwe, C., Chanda, C., Muzyamba, L., Mwape, J., & Phiri, L. (2024). Quantitative Analysis of Zambian Wikipedia Contributions: *Assessing Awareness, Willingness, Motivation, and the Impact of Gamified Leaderboards and Badges.* South African Computer Science and Information Systems Research Trends, 30–44.

[7] D. Cao, N. Béchet, and P.-F. Marteau. 2024. WikiNER-fr-gold: A Gold-Standard NER Corpus. *arXiv preprint arXiv:2411.00030*. https://doi.org/10.48550/arXiv.2411.00030

[8] Falk, M., et al. 2023. National Wikipedia studies: The case of Australian representation. *Journal of Information Science*.

[9] Ferschke, O., Gurevych, I., and Rittberger, M. 2014. FlawFinder: A Modular System for Predicting Quality Flaws in Wikipedia. *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics (EACL '14)*, 27–31.

[10] Graham, M., Hogan, B., Straumann, R. K., and Medhat, A. 2014. Uneven Geographies of User-Generated Information: Patterns of Increasing Informational Poverty. *Annals of the Association of American Geographers*, 104(4), 746–755.

[11] Halfaker, A., and Geiger, R. S. 2020. ORES: Lowering Barriers with Participatory Machine Learning in Wikipedia. *Proceedings of the ACM on Human-Computer Interaction (CSCW '20)*, 4(2), 1–37.

[12] Jagannatha, A. N., Yu, H., and Liu, F. 2015. A Comprehensive Study of Biomedical Content Gaps in Wikipedia. *Journal of Biomedical Informatics*, 57, 191–202.

[13] Luyt, B. 2018. Wikipedia and the Politics of Open Knowledge. *Journal of Documentation*, 74(5), 1016–1033.

[14] Miquel-Ribé, M., and Laniado, D. 2020. Wikipedia Cultural Diversity Dataset: A Complete Cartography for 300 Language Editions. *Proceedings of the 16th International Symposium on Open Collaboration (OpenSym '20)*, 1–10.

[15] Miquel-Ribé, M., and Laniado, D. 2021. Tracking Content Gaps in Wikipedia Over Time: A Longitudinal Analysis. *Proceedings of the ACM on Human-Computer Interaction (CSCW '21)*, 5(2), 1–24.

[16] Schubotz, M., et al. 2016. Automatically Detecting Incomplete Articles in Wikipedia. *Proceedings of the 12th International Symposium on Open Collaboration (OpenSym '16)*, 1–9.

[17] Sciascio, C., et al. 2019. A toolkit for interactive visualization and quality analytics of Wikipedia articles. In *Proceedings of the ACM Symposium on User Interface Software and Technology*. ACM.

[18] Tkacz, N. 2015. *Wikipedia and the Politics of Openness*. University of Chicago Press.

[19] Vrana, R., et al. 2020. Cultural silences and exclusions in Wikipedia: A qualitative assessment of missing biographical entries. *Journal of Critical Information Studies*.

6. Appendix A: Gantt Chart Study Timeline

		Name	Duration
1		**PROJECT DURATION**	142 days?
2		**RESEARCH DESIGN PLANNIGN**	50 days?
3		PROJECT DESCRIPTION SPECIFIED	1 day?
4		TEAM ROLE	1 day?
5		LITERATURE REVIEW DUE	0 days?
6		PROJECT PROPOSAL DUE	0 days?
7		PRESENTATION REHEARSAL	1 day?
8		PRESENTATION OF PROJECT	1 day?
9		REVISED PROJECT PROPOSAL	0 days?
10		PROJECT WEBSITE SETUP	0 days?
11		**REQUIREMENTS ENGEERING**	4 days?
12		WIKIPEDIA STANDARD ANAYSIS	3 days?
13		BENCHMARK GOLD STANDARD	2 days?
14		LLM NEEDS	2 days?
15		USER NEEDS	1 day?
16		DATA PRIVACY	2 days?
17		**ITERATION 1**	11 days?
18		DOM PARSING	3 days?
19		LLM MODULE	10 days?
20		SCORING ALGORITHM	5 days?
21		INTERNAL TESTING	7 days?
22		TOPIC CLASSIFICATION	5 days?
23		CITATION HEURSISTICS	6 days?
24		**ITERATION 2**	17 days?
25		DESIGN	8 days?
26		MAP SUGGESTION	7 days?
27		INTEGRATED TESTING	10 days?
28		**ITERATION 3**	36 days?
29		FRAMEWORK	5 days?
30		REAL TIME FEATURES	10 days?
31		LLM MODULES	10 days?
32		UI/UX FOR EDITORS	10 days?
33		SPRINT REVIEW	5 days?
34		**ITERATION 4**	17 days?
35		REPOSITORY SCHEMA	10 days?
36		BACKEND AND API SETUP	12 days?