A MINI-PROJECT REPORT

ON

"ARA - AI-powered Research Assistant"

Submitted to

Rashtrasant Tukadoji Maharaj Nagpur University, Nagpur

in partial fulfillment of the requirement for the degree of

BACHELOR OF TECHNOLOGY

in

Computer Science & Engineering

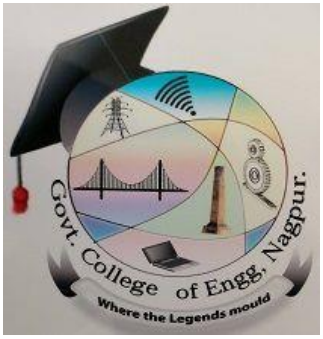
Submitted By

Kaustubh Warade **Aditya Deshmukh**

Devansh Parapalli **Yashasvi Thool**

under the Guidance of

Dr. D. J. Chaudhari



Department of Computer Science & Engineering

Government College of Engineering

Nagpur – 441108 (M. S.)

2023-2024

Certificate

This is to certify that the mini-project entitled

"ARA - AI-powered Research Assistant"

Is a bonafide work and it is submitted to the

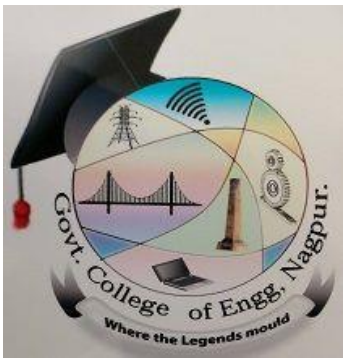
Rashtrasant Tukadoji Maharaj Nagpur University, Nagpur.

By

Kaustubh Warade **Aditya Deshmukh**

Devansh Parapalli **Yashasvi Thool**

For the partial fulfillment of the requirement for the degree of Bachelor of Technology in Computer Science & Engineering, during the academic year 2023-2024.



Dr. Devchand Chaudhari
Project Guide

Dr. Latest Bhagat
Head of CSE Dept.

Department of Computer Science & Engineering.

Government College of Engineering,

Nagpur – 441108 (M. S.)

2023-2024

ACKNOWLEDGMENT

It gives us immense pleasure in submitting the mini-project report on

"ARA - AI-powered Research Assistant"

to our guide **Dr. Devchand Jijiram Chaudhari**, and Head of Department **Dr. Latesh Gagan Malik,** who were a constant source of guidance and inspiration through the seminar work.

We are also very thankful to all staff members of the Computer Science & Engineering department; whose encouragement and suggestions assisted us to complete the Mini-Project work.

We also express our sincere gratitude to our respected Principal **Dr. R. P. Borkar** for providing us necessary facilities.

At last, we are thankful to our friends whose encouragement and constant inspiration helped us to complete this seminar work verbally and theoretically.

Aditya Deshmukh
Devansh Parapalli
Kaustubh Warade
Yashasvi Thool
Third Year B.E.
Computer Science & Engineering
Government College of Engg., Nagpur

ABSTRACT

The exponential growth of research data and information sources has posed significant challenges for researchers, leading to information overload, disconnected insights, missed opportunities, and inefficiencies in organizing and synthesizing knowledge. To address issues with inefficient research, ARA was developed. ARA is an innovative application leveraging advanced AI technologies to revolutionize the research process by enhancing information retrieval, analysis, and synthesis.

ARA represents a cutting-edge solution surpassing traditional note-taking approaches. ARA leverages large language models, semantic web technologies, and knowledge graphs to create a dynamic, interconnected web of research information, such information can be understood and processed by AI models.

ARA has delivered an advanced and efficient tool enhancing researchers' capabilities, streamlines their workflows, and contributes to the overall productivity of research activities. By fostering effective information retrieval and adaptability, ARA has the potential to drive groundbreaking discoveries and accelerate the pace of scientific progress across various disciplines.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

The modern research landscape is characterized by an ever-increasing deluge of information, posing significant challenges for researchers attempting to navigate such a complex terrain. The exponential growth of research data, publications, and online resources has led to a phenomenon known as "information overload," where researchers struggle to manage and process the vast amounts of information available. Consequently, valuable insights and connections often remain hidden, opportunities for collaboration are overlooked, and the overall efficiency of the research process is hindered.

In response to these challenges, ARA has been developed, a revolutionary application harnessing the power of advanced AI technologies to transform the way researchers approach information gathering, organization, and synthesis. ARA represents a paradigm shift from traditional note-taking applications, leveraging cutting-edge techniques such as large language models, knowledge graphs, and semantic web technologies to create a dynamic, interconnected web of research information that can be understood and processed by AI models.

The core objective of ARA is to empower researchers by providing a comprehensive and intelligent research assistant that streamlines their workflow, unveils hidden insights, expands research horizons, and sharpens critical thinking and communication skills. By automating routine tasks, accelerating data analysis, and providing contextual access to relevant information, ARA aims to significantly boost research productivity and efficiency, ultimately contributing to the acceleration of scientific progress across various disciplines.

CHAPTER 2

LITERATURE SURVEY

In the era of rapidly expanding scientific literature, automated methods have emerged as an asset in managing and streamlining the process. AI-based tools offer a myriad of features, including natural language processing, citation analysis, and automated summarization, enabling researchers to discover, evaluate, and organize relevant research papers efficiently. Given below are tools and platforms which were analyzed as part of the requirements of ARA.

2.1. Lex[5]

Lex learns from the text the user adds and uses it to generate new content. Academic and technical writing can be mentally draining, and Lex can take some of the grunt work.

Lex only summarizes the content for the purposes of sharing and does not aid in the process of researching.

2.2. Notion[1]

Notion is a collaboration platform with Markdown[2][40] and inducing kanban boards[3], tasks, wikis[4] and databases. It is a workspace for notetaking, knowledge and data management and project and task management.

Notion only provides rudimentary AI features which enable text prediction, generation and solving of simple arithmetic expressions.

2.3. Elicit[26]

Elicit is an AI assistant that uses language models to answer research questions. It can find relevant papers without needing perfect keyword matches, summarize key takeaways, and extract important information.

However, Elicit cannot search the web for generalized information, is not a good fit for identifying facts and theoretical or non-empirical domains.

2.4. Research Rabbit[37]

Research Rabbit is an AI-powered platform that assists researchers in discovering, visualizing, and analyzing relevant literature.

Research Rabbit cannot search for general information and does not transform text. The authors of Research Rabbit put it well, “Research Rabbit is like Spotify for Research”

2.5. ChatPDF[23]

ChatPDF is an AI-powered tool that functions as an interactive chatbot for PDF documents. It can answer queries, rewrite sections, and provide insights.

ChatPDF cannot utilize information beyond the provided PDF and hallucinates frequently.

2.6. Consensus[24]

Consensus is an AI-powered search engine that answers questions based on peer-reviewed literature, providing evidence-based answers and a “consensus meter” reflecting the state of current research.

Consensus does not possess the ability to generate summarizations or complete any custom task given by the user.

2.7. IBM Watson[46]

IBM Watson is a platform that offers various AI-powered tools for academic research, including data extraction, sentiment analysis, and language processing features to smoothen the research process and discover insights from unstructured data.

However, IBM Watson being a platform provides APIs and SDKs to integrate into your own software, and as such cannot be used by the general populace.

CHAPTER 3

ANALYSIS

In the due course of researching for ARA, the existence of various tools, platforms and components sharing a common goal as ARA were found.

3.1. Existing Systems and Drawbacks.

Lex can generate new content by learning from provided text, but it does not aid in the research process itself by finding or summarizing sources.

Notion provides collaboration and knowledge management features with basic AI capabilities like text prediction but cannot search for or analyze research literature.

Elicit uses language models to find relevant papers and summarize key points but cannot search the general web or handle non-empirical domains.

Research Rabbit visualizes and analyzes literature but does not transform text or search beyond research papers.

ChatPDF allows querying and rewriting PDF documents through an AI chatbot but cannot utilize information beyond the given PDF.

Consensus provides evidence-based answers from peer-reviewed literature but lacks the ability to generate summarizations or handle custom tasks.

IBM Watson offers powerful AI capabilities for academic research like data extraction and language processing, it requires technical integration as a platform rather than an out-of-the-box solution.

3.2. Proposed Application

The proposed, ARA is an innovative application that aims to revolutionize the academic research process through the integration of cutting-edge artificial intelligence technologies. Departing from conventional note-taking

applications, ARA will facilitate the creation of a dynamic, interconnected web of research data that can be comprehended and processed by AI models.

Key features to be incorporated in ARA include intelligent information retrieval mechanisms to gather data from diverse sources, contextual synthesis algorithms to amalgamate gathered information, effective organization frameworks to structure insights, and cross-disciplinary connection identification capabilities.

ARA is designed to address prevalent challenges faced by researchers, such as information overload, fragmented comprehension, overlooked collaboration opportunities, and inefficiencies in academic writing. By automating routine tasks, accelerating data analysis, and providing expedient access to pertinent information, ARA endeavors to substantially enhance research productivity.

Furthermore, it will leverage AI analysis to unveil latent insights, expand research horizons by recommending relevant advancements and potential collaborations, and refine critical thinking and communication proficiencies – ultimately driving innovation and facilitating groundbreaking discoveries within the academic research domain.

3.3. Feasibility Study

A comprehensive feasibility study was conducted to assess the viability and potential impact of the proposed ARA application. The study examined the current challenges faced by researchers, evaluated existing solutions, and identified the critical requirements for an AI-powered research assistant. The study confirmed the pressing need for a tool that can effectively manage information overload, establish connections between disparate data sources, uncover new opportunities for collaboration, and enhance the communication of research findings.

An in-depth analysis of the latest advancements in artificial intelligence, particularly in areas such as natural language processing,

knowledge representation, and machine learning, indicated that the necessary technological foundations are available to develop the envisioned ARA system. Furthermore, the study explored potential data sources, including open-access repositories, academic databases, and online knowledge bases, confirming the availability of sufficient data to train and power the AI models underpinning ARA.

The feasibility study also evaluated the project's resource requirements, including computational power, data storage and management infrastructure, and the integration of relevant AI frameworks and libraries. Based on the findings, the development of ARA was deemed feasible, with a high potential for transforming the research landscape and driving significant productivity gains for academic researchers across various disciplines.

3.4. Technical Design Theory

The technical design of ARA will follow a modular and layered architecture, allowing for scalability, flexibility, and easy integration of various components. The core of the system will be built around state-of-the-art natural language processing (NLP) models and knowledge representation techniques. NLP models will be employed for intelligent information retrieval, content summarization, and text generation tasks. Semantic web technologies, such as Resource Description Framework (RDF)[17][18] and Web Ontology Language (OWL)[18], will be leveraged to represent and reason over the gathered knowledge, facilitating cross-disciplinary connections and insights.

The system will also incorporate machine learning algorithms for continuous learning and adaptation, ensuring that ARA's capabilities evolve with changing research needs and user feedback. Emphasis will be placed on developing robust security measures to protect sensitive research data and ensure compliance with privacy regulations.

CHAPTER 4

DESIGN

4.1. Activity Diagram

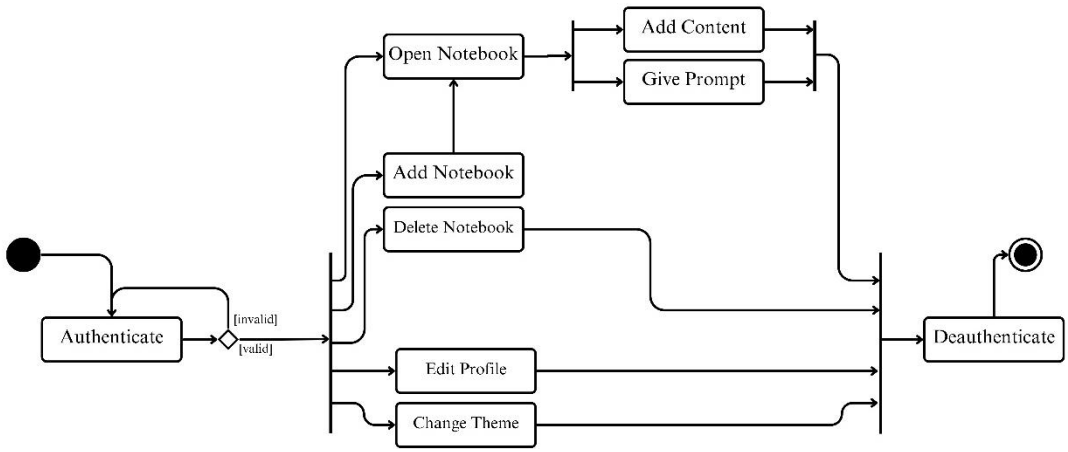


Figure 4.1: Activity Diagram

4.2. Class Diagram

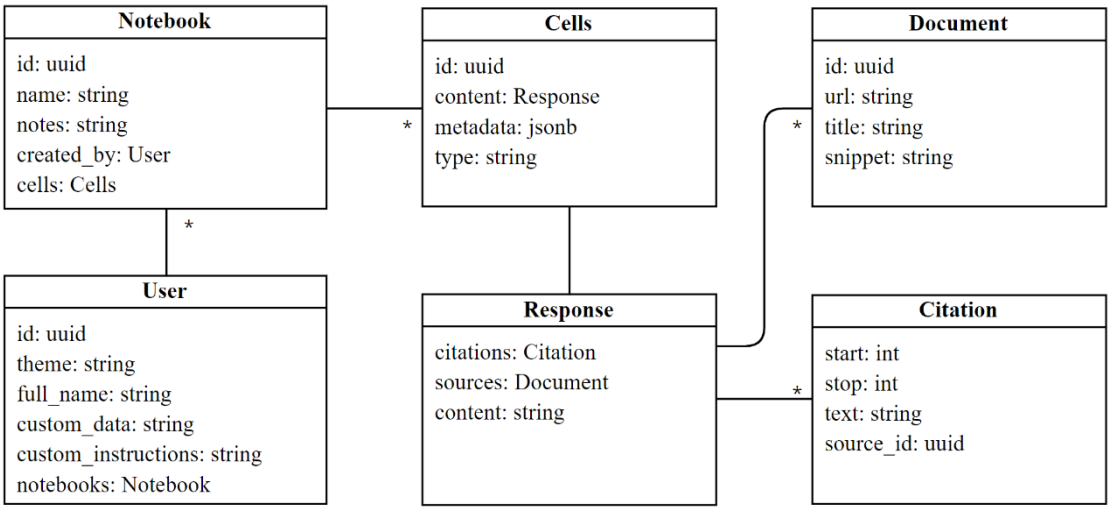


Figure 4.2: Class Diagram

4.3. Collaboration Diagram

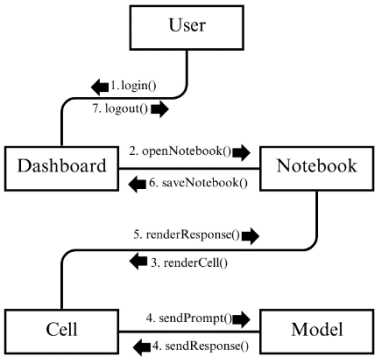


Figure 4.3: Collaboration Diagram

4.4. State Diagram

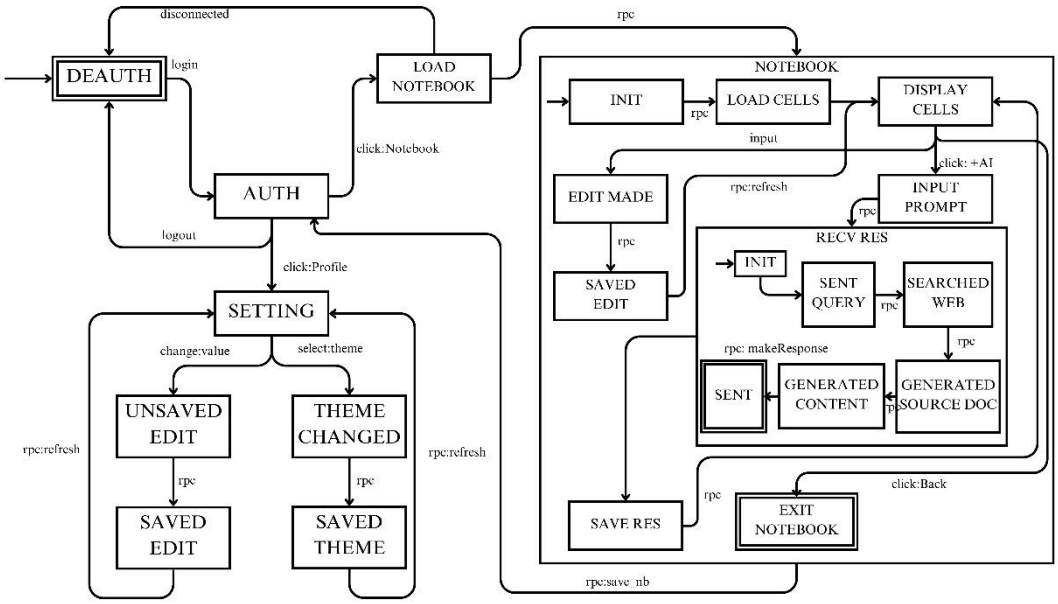


Figure 4.4: State Diagram

4.5. Use Case Diagram

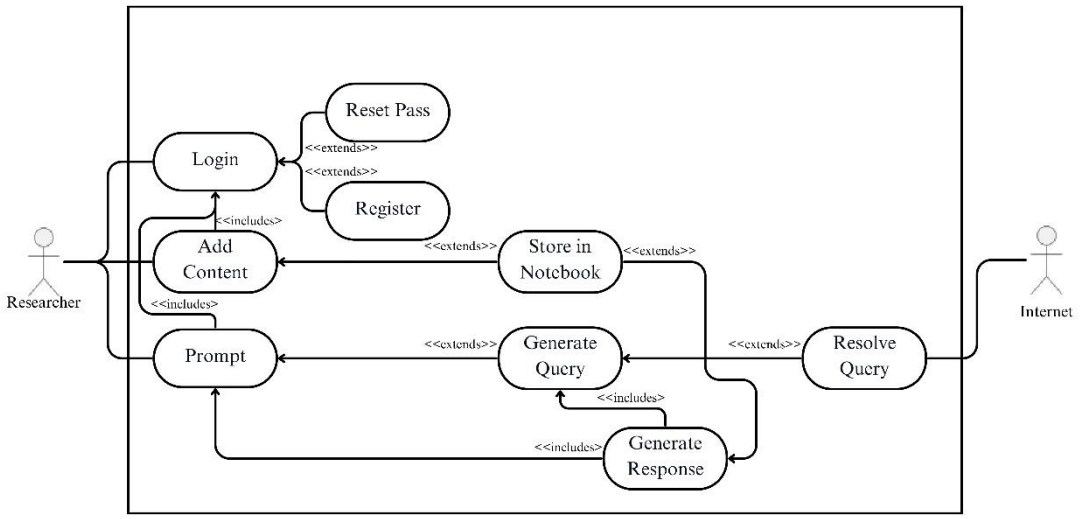


Figure 4.5: Use Case Diagram

CHAPTER 5

SYSTEM REQUIREMENTS

ARA is split into 4 components, each with their own set of requirements.

5.1. Deployment

A centralized server is responsible for serving the assets to the user and hosting the Large Language Models.

5.1.1. Hardware Requirements

ARA requires a moderately powerful computer system, capable of handling compilation and CI/CD tasks in parallel to running multiple threads for servicing requests.

Table 5.1: Hardware Requirements – Web Server

Sr. No.	Item	Specifications	Quantity
1.	Computer System	Intel(R) Core[TM] i7, AMD Ryzen 5000 or better. Nvidia(R)RTX[TM] 4080 or better. 64 GB DDR4 RAM 4 TB SSD Storage 2.5GBPS NIC	1

5.1.2. Software Requirements

The web server requires an advanced tech-stack for deployment due to the inclusion of Large Language Models.

Table 5.2: Software Requirements – Web Server

Sr. No.	Item	Version
1.	Operating System – Debian[TM][39]	12.5
2.	Docker Engine[25]	26.0.0

3.	Docker Desktop[25]	4.28.0
4.	Docker Compose[25]	2.26.0
5.	Kubernetes[42]	1.29

5.2. Development

ARA is built using state-of-the-art libraries and packages, including software still in beta-testing. ARA can be developed sufficiently well on a modern computer, built after 2016. A list of software used during development is given below.

Table 5.3: List of software used during development.

Sr. No.	Item
1.	Node.js[35]
2.	pnpm[28]
3.	Typescript[34]
4.	Svelte[12] [40]+SvelteKit[13][41]
5.	Supabase[47]
6.	PostCSS[41]
7.	TailwindCSS[41]
8.	Autoprefixer[22]
9.	Playwright[27]
10.	Vite[45]
11	Vitest[21]
12	Langchain.js[32]
13	Katex
14	Markdown[2]
15	Python
16	Ruff
17	Langchain.py[32]
18	FastAPI[29]

19	Numpy, Pandas, Scikit-learn
20	Uvicorn[44]
21	Mistletoe
22	Magnum
23	Ujson
24	Gunicorn[31]
25	Nginx
26	DaisyUI
27	Docker Engine
28	Docker Compose
29	Docker Desktop
30	Kubernetes

5.3. User Requirements

Since ARA is built using a Client-Server[36] model, the end user does not require very powerful computers. An end user must simply have a computer built after 2010, containing up to date browser software.

ARA is built using transpilation[30] to ensure that it is functional on older systems. Since ARA is built using SvelteKit, the application can be server-rendered in a technique called Server Side Rendering[33], eliminating the requirement for JavaScript to access the basic functions of the application.

CHAPTER 6

IMPLEMENTATION

The implementation of ARA adhered to the traditional Waterfall software development life cycle, proceeding through sequential phases of feasibility study, requirement analysis, system design, coding, testing, and deployment. A comprehensive feasibility study evaluated the project's viability, followed by rigorous requirement gathering from subject matter experts. The system architecture and detailed designs were meticulously planned based on the requirements. The implementation phase involved developing the data storage, machine learning models, semantic technologies, and user interface modules as per the approved designs. Rigorous testing, including unit, integration, system, and user acceptance tests, was conducted using automated frameworks to ensure quality and reliability. Upon successful testing, ARA was deployed into production, with comprehensive deployment plans and ongoing maintenance to address enhancements and evolving needs, while following the structured Waterfall[15][38] process for effective project management and risk mitigation.

6.1. User Interface

The ARA user interface was carefully designed with a strong emphasis on intuitive navigation, visual clarity, and a seamless user experience. The following screenshots showcase some of the key features and views within the application, along with explanations where required.

The root page of the ARA application features two separate views: one for authenticated users, providing a comprehensive workspace for research management and collaboration, and another for unauthenticated users, serving as an informative landing page and guiding the authentication process.

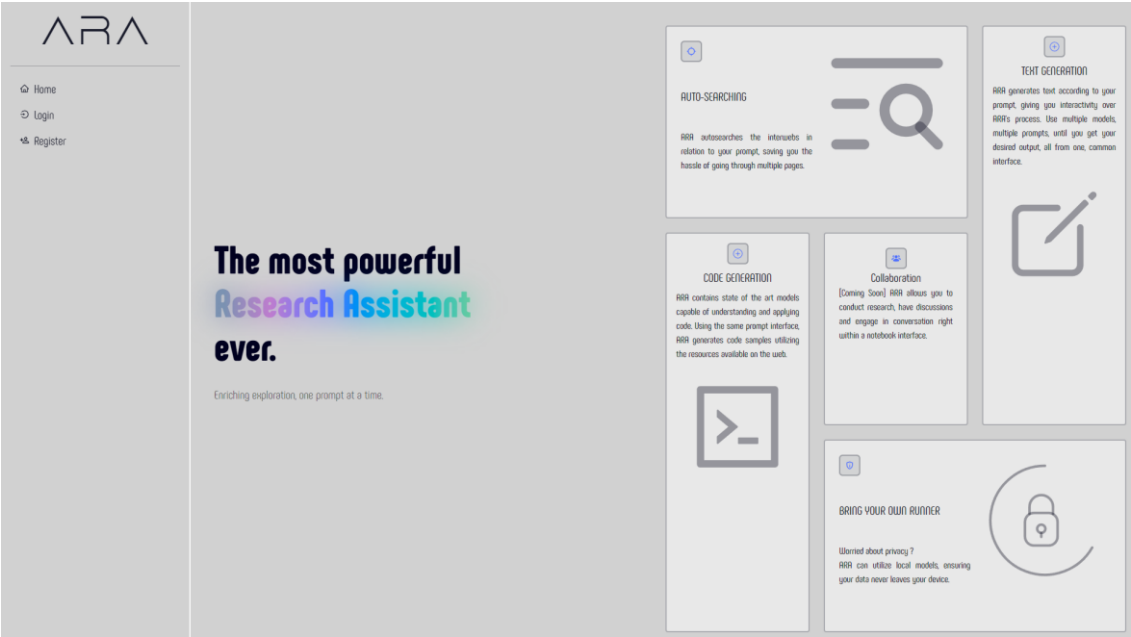


Figure 6.1: Unauthenticated Home Page View

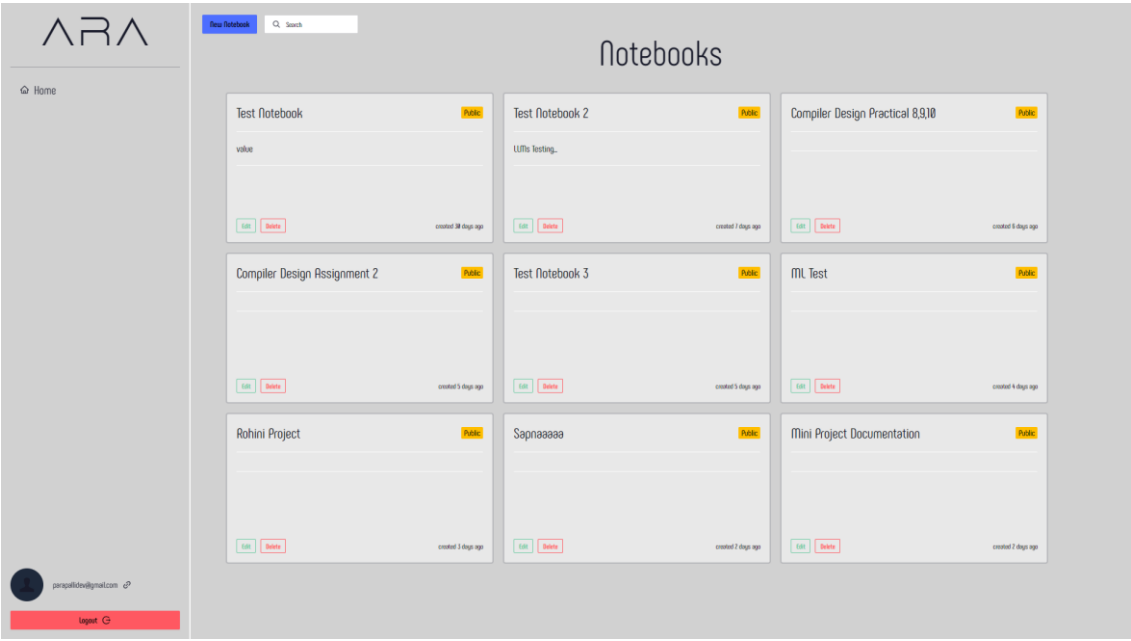


Figure 6.2: Authenticated Home Page View

The login and registration pages provide a convenient way to enter the application. They can be accessed using the left sidebar. Currently implemented are the email and password authentication flows, other flows such as OAuth2[11] are possible to integrate.

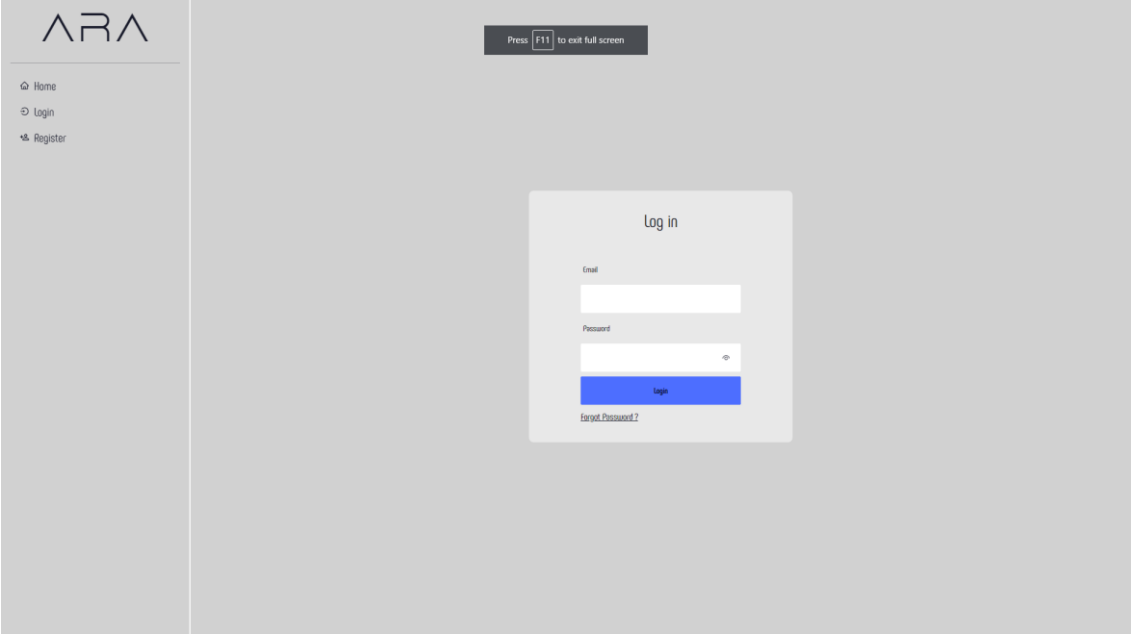


Figure 6.3: Login Page

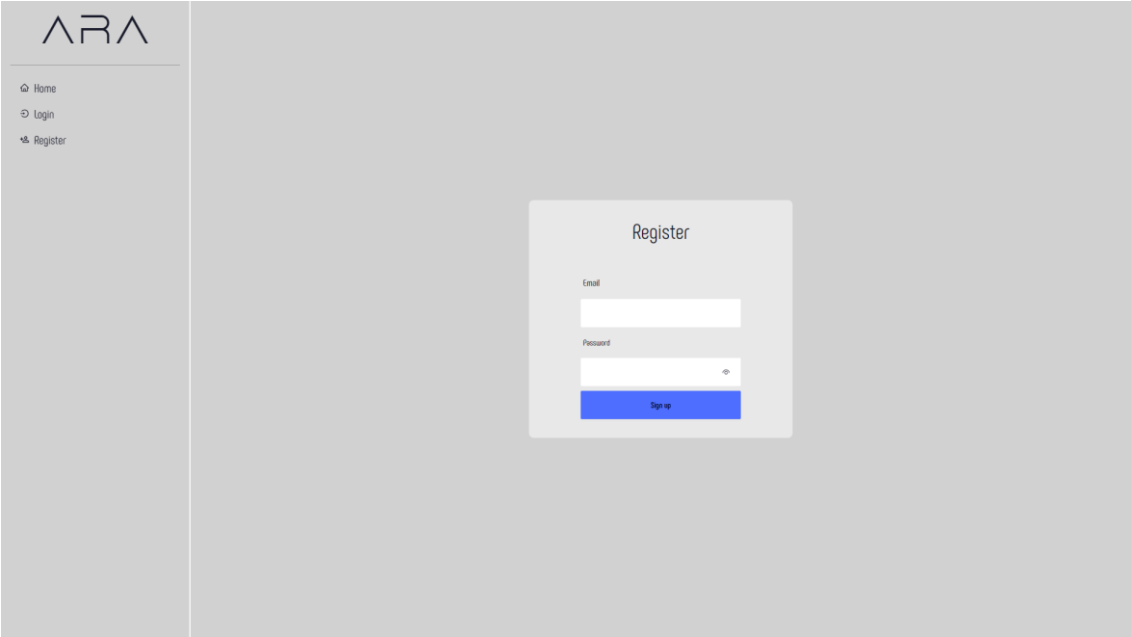


Figure 6.4 Register Page

Once logged in, the user is redirected to the page presented in Figure 6.2

Upon selecting a notebook to open, the user is redirected to the page specific to that notebook, containing its identifier in the URL. The user is then presented a prompt asking to verify the runner's location, self-hosted or hosted by ARA.

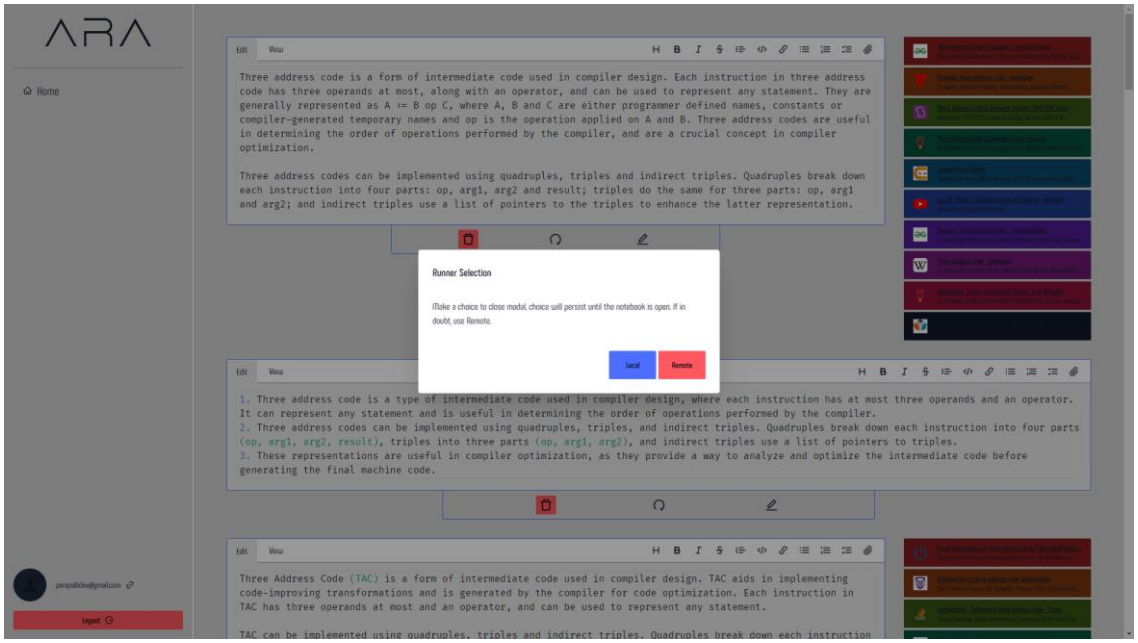


Figure 6.5 Prompt for Runner Selection

Upon selection of the runner, the user is presented the notebook, shown in Figure 6.6.

Figure 6.6: Notebook View - Standard

As presented in Figure 6.6, there are two types of cells, AI-generated ones as well as User-generated ones. The AI-generated contain attached sources, that were used to generate the textual response.

Figure 6.7: Notebook View – Buttons for New Cell

Figure 6.7 shows the buttons that are used to add new cells to a notebook. Provisions are provided for adding a new User Cell, a new AI Cell, and changing of the Runner location.

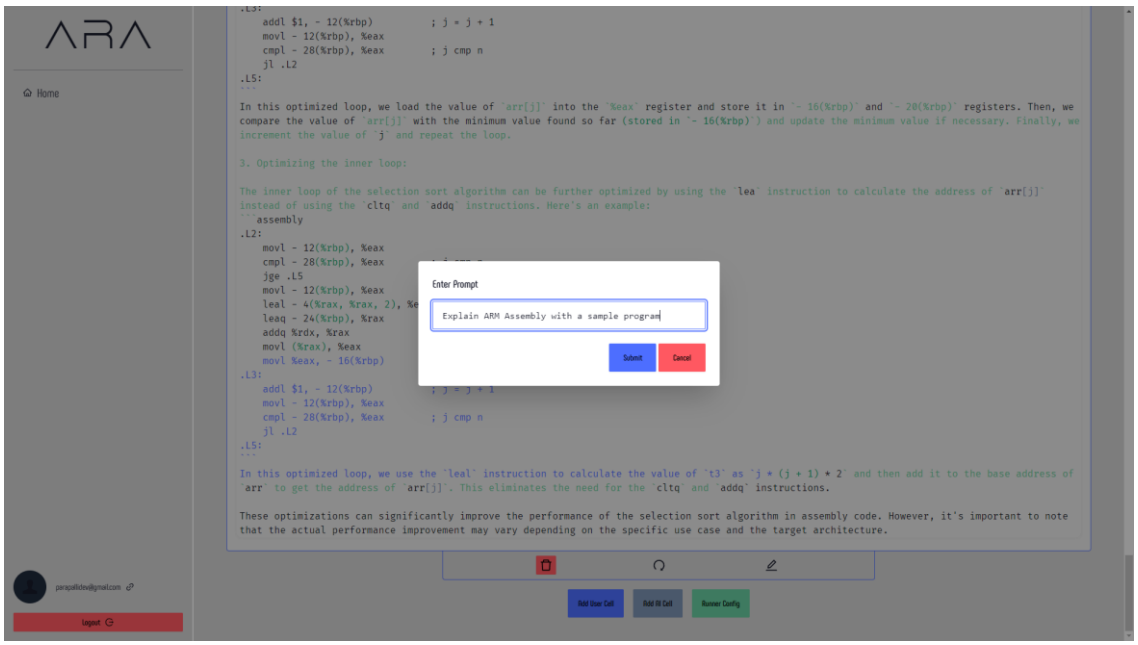


Figure 6.8: Prompt for AI-generated cell.

Clicking on Add AI Cell button asks the user for a prompt to begin the process of searching and generating.

Figure 6.9: Mid-generation cell

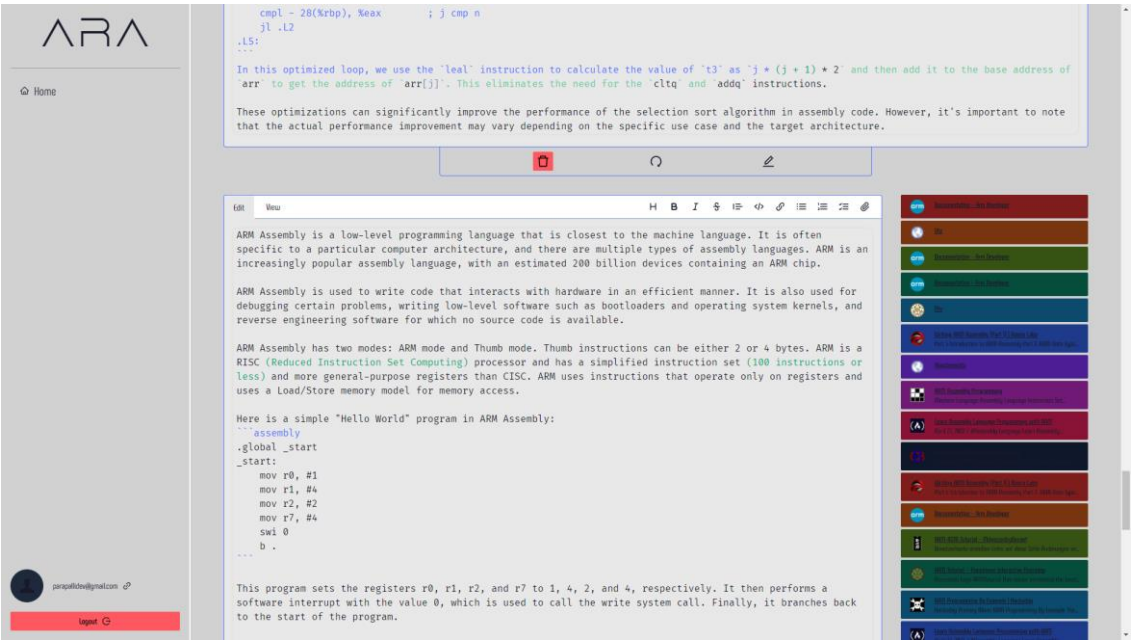


Figure 6.10: Text-generation complete

Figure 6.9 and Figure 6.10 demonstrate the text-generation capablity of ARA. Figure 6.9 demonstrates a cell being generated while in Figure 6.10 the cell has been generated completely.

Clicking on the profile link (present in the bottom left, containing the user's email) navigates the user to the settings page

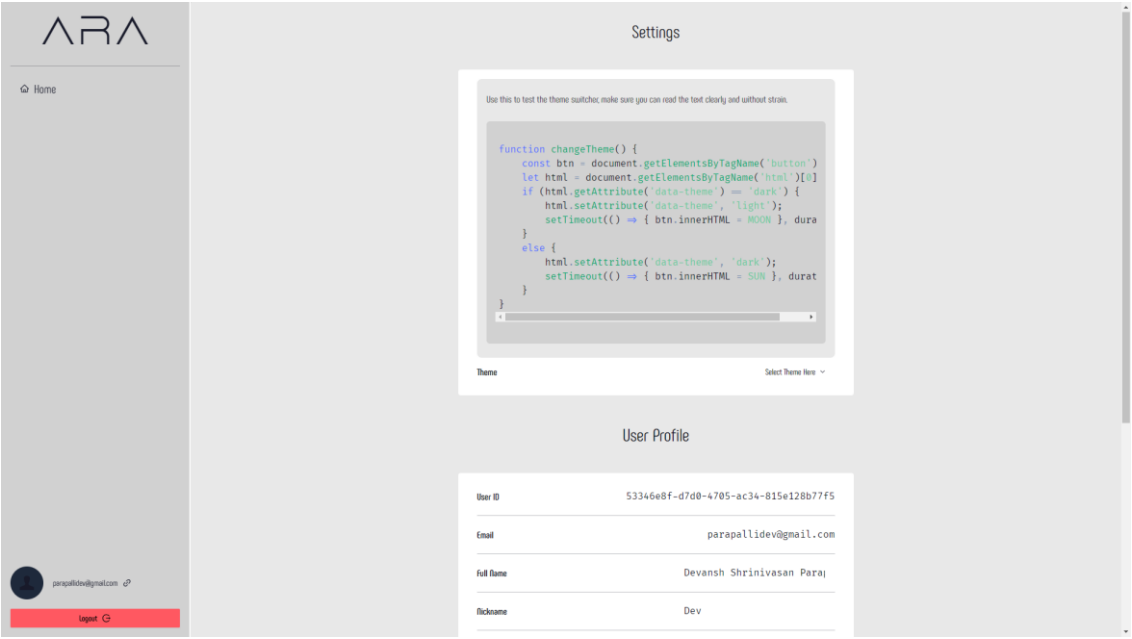


Figure 6.11: ARA Settings (Theme)

Due to the existance of user preferences of theme and the color palette, ARA contains 25 themes, each theme contains distinct values for border-radius (roundness), colors and overall feel and look of ARA. The user is free to choose any theme. The theme choice is saved along with the user's profile.

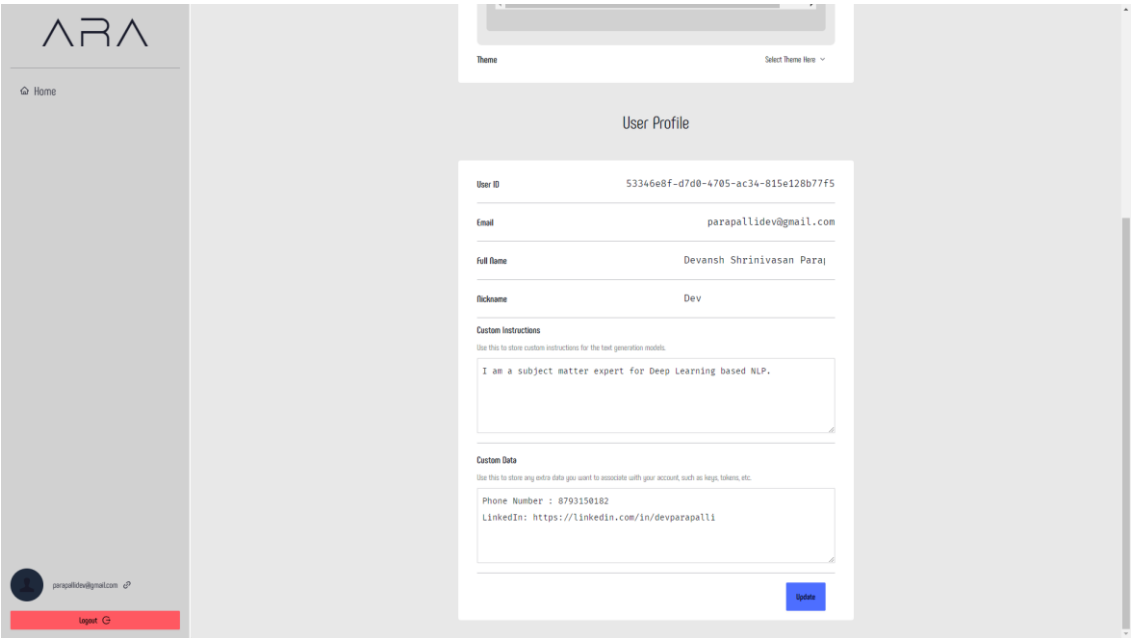


Figure 6.12: User Profile Settings

Figure 6.12 outlines the profile view, containing fields required that can be passed to ARA as additional arguments.

Figure 6.13 shows the "sunset" theme.

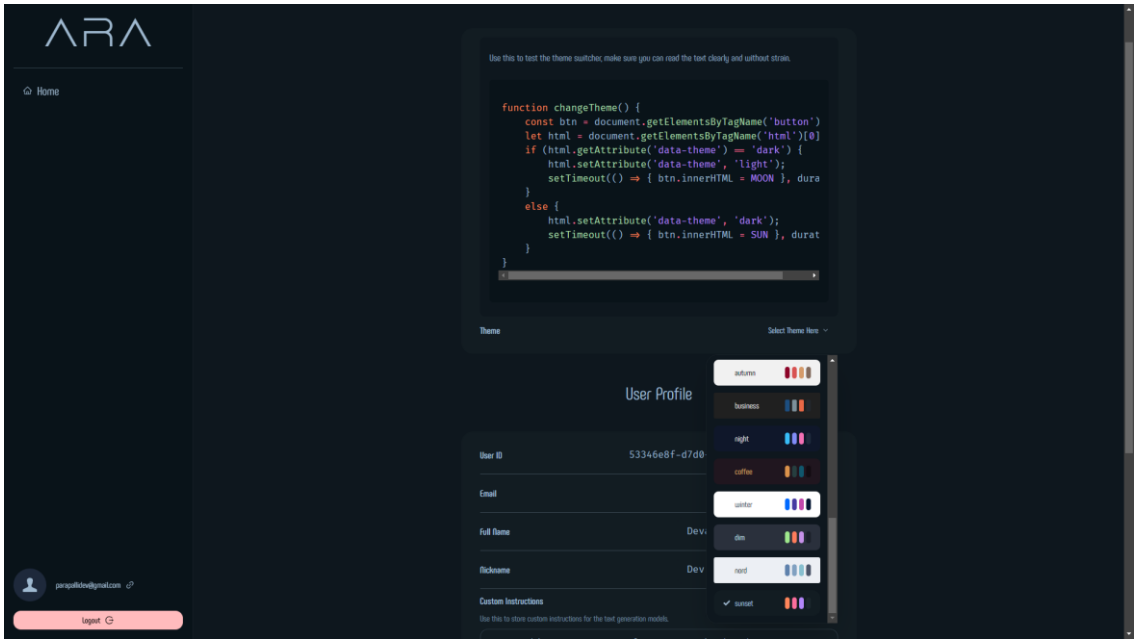


Figure 6.13: Settings Page (Dark Theme)

CHAPTER 7

SOFTWARE TESTING

Rigorous software testing is crucial for ensuring the reliability, functionality, and performance of the ARA system. A comprehensive testing strategy will be implemented throughout the development lifecycle, employing a range of testing techniques and methodologies.

7.1. Components of Testing.

The testing phase will involve the following key components:

7.1.1. Unit Testing

Individual units or components of the ARA system have undergone thorough unit testing to validate their functionality and behavior in isolation. These tests were automated using the Vitest package.

7.1.2. Integration Testing

As individual units are integrated into larger subsystems, integration testing allows for identification and resolution of any interface issues, data inconsistencies, or compatibility problems that arise during the integration phase.

7.1.3. System Testing

The entire system of ARA was subjected to comprehensive system testing to validate its end-to-end functionality and performance under realistic conditions. The test cases simulated a user browsing through the entire system, picking and prodding at it at every step. Such testing was conducted using a framework called Playwright.

7.1.4. Acceptance Testing

Upon completion of the above testing paradigms, ARA was subjected to Acceptance Testing to validate the system's readiness for deployment and conformance to the project's acceptance criteria.

7.2. Automated Test Cases

ARA incorporated a small set of automated, comprehensive test cases to validate the core functionality of the system.

Table 7.1: Automated End-To-End Test Cases

Sr. No.	Test Case	Result
1.	NotebookFunctionality	PASS
2.	AICellPrompt	PASS
3.	Login_Register	PASS
4.	Settings_TestPage	PASS

Unit testing rigorously validated core components like data transformations, state management, AI models, and UI elements to ensure ARA's reliability and maintainability.

Table 7.2: Automated Unit Test Cases

Sr. No	Test Case	Result
1.	Model_response	PASS
2.	Sources_order	PASS
3.	No broken links	PASS
4.	Citations received	PASS
5.	Citation order correct	PASS
6.	Local Models	PASS
7.	Remote Models	PASS
8.	Supabase (connection to supabase)	PASS
9.	Func: convert_cells_to_context	PASS
10.	Func: getRelativeTime	PASS

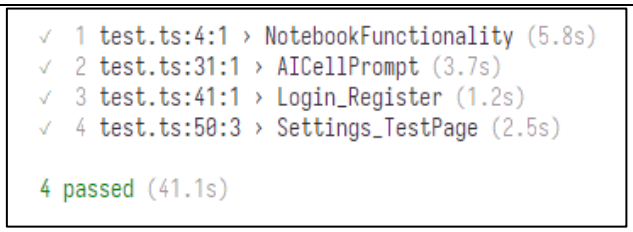


Figure 7.1: Automated End-to-End Test Case Result

Figure 7.2: Automated Unit Test Case Result

7.3. Manual Testing

While automated testing played a crucial role in validating the functionality and performance of ARA, extensive manual testing was also conducted to ensure a comprehensive evaluation of the system's capabilities and user experience.

Manual testing efforts focused on verifying the accuracy and relevance of the information retrieval, synthesis, and organization features. Testers explored a wide range of research topics, assessing the system's ability to

gather data from diverse sources, establish meaningful connections, and present insights in a clear and comprehensible manner.

In addition to functional testing, manual testing encompassed an in-depth evaluation of the system's performance under various conditions, including large data sets. Simulated scenarios pushed the system's limits, ensuring that ARA remained responsive, reliable, and scalable even under significant workloads.

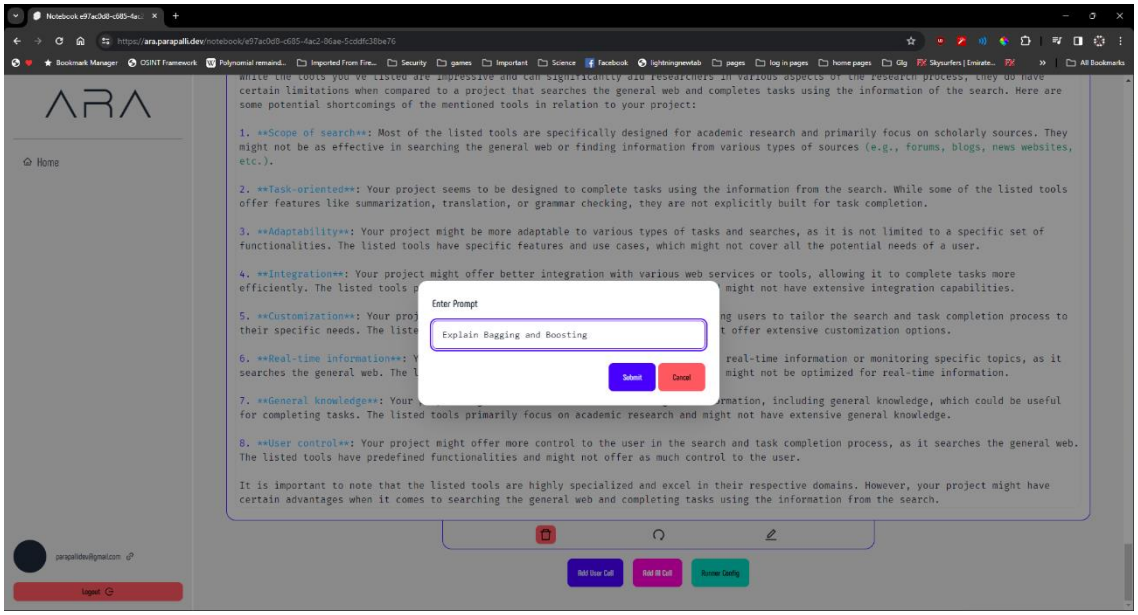


Figure 7.3: Manual Testing Prompt

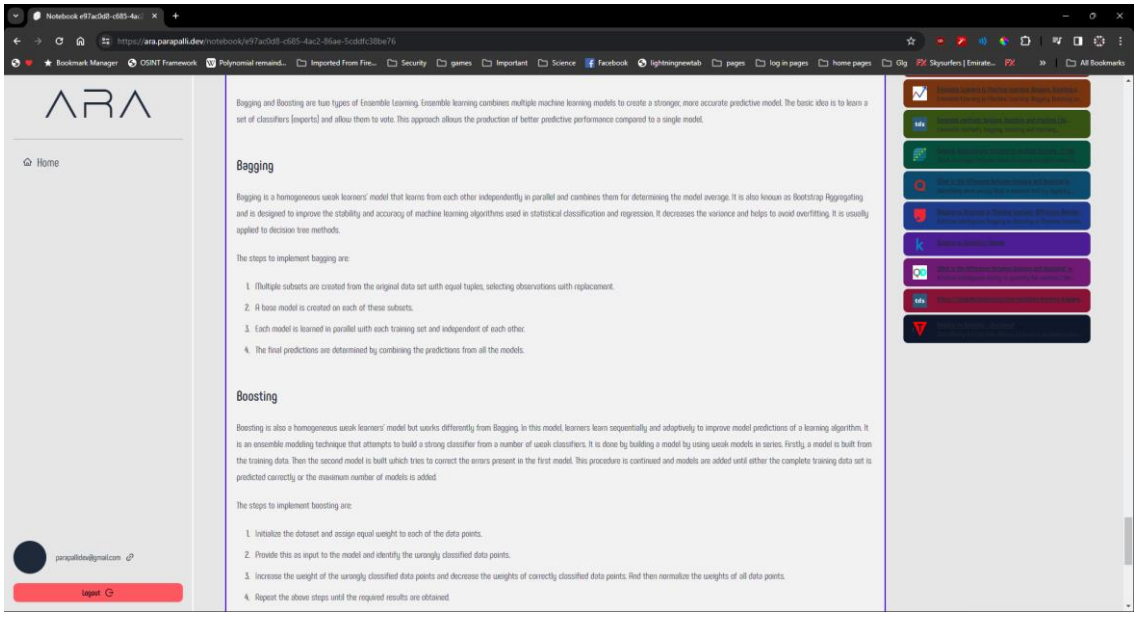


Figure 7.4 Manual Testing Response

CHAPTER 8

RESULT DISCUSSION

The development and deployment of ARA application has yielded promising results. Through extensive testing and evaluation, the system has exhibited significant improvements, as discussed below.

8.1. Information Retrieval and Synthesis

ARA's advanced natural language processing and information retrieval capabilities have proven highly effective in gathering relevant data from diverse sources, including academic databases, open-access repositories, and online knowledge bases. The system's ability to contextualize and synthesize information has resulted in comprehensive and insightful research summaries, facilitating a deeper understanding of complex topics.

8.2. Productivity and Efficiency Gains

Through automation of routine tasks, such as literature reviews, data gathering, and initial analysis, ARA has significantly reduced the time and effort required for researchers to perform these crucial but time-consuming activities. Consequently, researchers have reported substantial productivity gains, allowing them to focus their efforts on higher-level cognitive tasks, such as hypothesis formulation, experimentation, and result interpretation.

8.3. Continuous Learning and Adaptation

One of the key strengths of ARA lies in its ability to continuously learn and adapt based on user feedback and evolving research needs. Through machine learning algorithms and iterative refinement, the system has demonstrated improvements in its performance over time, becoming more accurate, efficient, and tailored to the specific requirements of various research domains.

CHAPTER 9

APPLICATION

The successful development and deployment of ARA has opened numerous potential applications across various domains of academic research. ARA can be leveraged to streamline and enhance research processes, unveil hidden insights, and drive innovation in a wide range of fields.

9.1. Interdisciplinary Research

ARA's ability to establish connections across diverse data sources and knowledge domains makes it an invaluable tool for interdisciplinary research projects. By seamlessly integrating information from multiple disciplines, ARA can provide researchers with a comprehensive understanding of complex problems and fostering novel approaches to address challenges that transcend traditional boundaries.

9.2. Literature Reviews and Meta-Analysis

The efficient information retrieval, synthesis, and organization capabilities of ARA can significantly accelerate the process of conducting comprehensive literature reviews and meta-analyses. Researchers can leverage ARA to quickly identify relevant studies, synthesize findings, and uncover emerging trends or gaps in existing research,.

9.3. Hypothesis Generation and Exploration

By uncovering hidden connections and unveiling previously overlooked insights, ARA can assist researchers in generating novel hypotheses and exploring unexplored research avenues. Through its advanced data analysis and knowledge representation techniques, ARA can identify patterns, anomalies, or unexplained phenomena, fueling curiosity and stimulating new lines of inquiry.

9.4. Research Collaboration and Knowledge Sharing

ARA's collaborative features and knowledge-sharing capabilities can foster increased cooperation among researchers, both within and across institutions. By facilitating the exchange of insights, data, and findings, ARA can promote the formation of interdisciplinary research teams and accelerate the dissemination of knowledge, ultimately driving scientific progress.

CHAPTER 10

CONCLUSION

In conclusion, ARA exemplifies the powerful synergy between cutting-edge technology and academic research, demonstrating the transformative potential of artificial intelligence in accelerating scientific progress and fostering innovation. By addressing longstanding challenges and empowering researchers with advanced tools, ARA is poised to reshape the research landscape and unlocking new frontiers of knowledge.

The development and deployment of ARA represents a significant milestone in the application of cutting-edge artificial intelligence technologies to academic research. Through its innovative approach to information retrieval, synthesis, and knowledge representation, ARA addresses longstanding challenges faced by researchers, including information overload, fragmented understanding, missed collaboration opportunities, and inefficiencies in academic writing and communication.

The extensive testing and evaluation of ARA have demonstrated its ability to streamline research workflows, unveil hidden insights, and foster cross-disciplinary collaboration. By automating routine tasks, accelerating data analysis, and providing quick access to relevant information, ARA has enabled researchers to focus their efforts on higher-level cognitive tasks, significantly enhancing productivity and driving innovation.

Furthermore, ARA's intuitive user interface, personalized research experience, and collaborative features have contributed to a positive user experience, facilitating knowledge sharing and interdisciplinary exploration. The system's ability to continuously learn and adapt based on user feedback and evolving research needs further solidifies its potential for long-term impact and relevance in the rapidly evolving research landscape.

REFERENCES

[1] Notion Labs, Inc. "Your connected workspace for wiki, docs & projects" Retrieved January 10, 2024, from *https://www.notion.so*

[2] The Daring Fireball Company LLC. "Markdown Syntax Documentation". Retrieved January 10, 2024, from *https://daringfireball.net/projects/markdown/syntax#philosophy*

[3] Atlassian "What is a Kanban Board?". Retrieved January 10, 2024, from *https://www.atlassian.com/agile/kanban/boards*

[4] Bo Leuf and Ward Cunningham. 2008. *The wiki way: Quick collaboration on the web*, Boston, Massachusetts: Addison-Wesley.

[5] Lex Inc. Retrieved January 10, 2024, from *https://lex.page/*

[6] The PostgreSQL Global Development Group "PostgreSQL". Retrieved January 10, 2024, from *https://www.postgresql.org/*

[7] Guido van Rossum and Python Development Team. 2018. *The Python Library Reference: Release 3.6.4*.

[8] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. Retrieved January 10, 2024, from *https://www.tensorflow.org/*

[9] François Chollet and others. 2015. Keras. https://keras.io. Retrieved January 10, 2024, from *https://keras.io*

[10] OpenAI "Better language models and their implications". Retrieved January 10, 2024, from *https://openai.com/research/better-language-models*

[11] OAuth "OAuth 2.0". Retrieved January 10, 2024, from *https://oauth.net/2/*

[12] Svelte "Svelte • Cybernetically enhanced web apps". Retrieved January 10, 2024, from *https://svelte.dev/*

[13] Svelte "SvelteKit • Web development, streamlined". Retrieved January 10, 2024, from *https://kit.svelte.dev/*

[14] Mozilla Corporation "Progressive web apps | MDN". Retrieved January 10, 2024, from *https://developer.mozilla.org/en-US/docs/Web/Progressive_web_apps*

[15] Kai Petersen, Claes Wohlin, and Dejan Baca. 2009. *The Waterfall Model in Large-Scale Development*. In Product-Focused Software Process Improvement, Springer Berlin Heidelberg, Berlin, Heidelberg, 386–400.

[16] Amazon Web Services, Inc. "Train Machine Learning Models - Amazon SageMaker Model Training - AWS". Retrieved January 10, 2024, from *https://aws.amazon.com/sagemaker/train/*

[17] W3C "Semantic Web FAQ". Retrieved January 10, 2024, from *https://www.w3.org/RDF/FAQ*

[18] W3C "RDF 1.1 Primer". Retrieved January 10, 2024, from *https://www.w3.org/TR/rdf11-primer/*

[19] The Linux Foundation "PyTorch". Retrieved January 10, 2024, from *https://pytorch.org/*

[20] Fielding, Roy Thomas. 2000. *Architectural Styles and the Design of Network-based Software Architectures*. Doctoral dissertation, University of California, Irvine.

[21] Anthony Fu Ari Perkkiö, Patak, Joaquín Sánchez, Dunqing, Hiroshi Ogawa and Vitest Contributors. (n.d.). *Vitest*. Next Generation Testing Framework. Retrieved April 8, 2024, from https://vitest.dev/

[22] *Autoprefixer CSS*. (n.d.). Retrieved April 8, 2024, from https://autoprefixer.github.io/

[23] ChatPDF GmbH. (n.d.). *ChatPDF - Chat with any PDF!* ChatPDF. Retrieved April 8, 2024, from https://www.chatpdf.com/

[24] Consensus NLP, Inc. (2024, March 3). *Consensus: AI search engine for research*. Consensus: AI Search Engine for Research. https://consensus.app/

[25] Docker Inc. (2024, January 23). *Docker: Accelerated Container Application Development*. Docker. Retrieved April 8, 2024, from https://www.docker.com/

[26] Elicit Research, PBC. (n.d.). *Elicit - Analyze research papers at superhuman speed*. Elicit. Retrieved April 8, 2024, from https://elicit.com/

[27] *Fast and reliable end-to-end testing for modern web apps | Playwright*. (n.d.). https://playwright.dev/

[28] *Fast, disk space efficient package manager | pnpm*. (n.d.). https://pnpm.io/

[29] *FastAPI*. (n.d.). https://fastapi.tiangolo.com/

[30] Fuertes, A. B., Pérez, M. J. V., & Hormaza, J. M. (2023). Transpilers: A systematic mapping review of their usage in research and industry. *Applied Sciences (Basel)*, *13*(6), 3667. https://doi.org/10.3390/app13063667

[31] *Gunicorn - Python WSGI HTTP Server for UNIX*. (n.d.). https://gunicorn.org/

[32] Harrison, C. (2022). *LangChain* [Software]. https://github.com/langchain-ai/langchain

[33] Iskandar, T. F., Lubis, M., Kusumasari, T. F., & Lubis, A. R. (2020). Comparison between client-side and server-side rendering in the web

development. *IOP Conference Series: Materials Science and Engineering*, *801*(1), 012136. https://doi.org/10.1088/1757-899x/801/1/012136

[34] *JavaScript with syntax for types.* (n.d.). https://www.typescriptlang.org/

[35] *Node.js — Run JavaScript everywhere*. (n.d.). https://nodejs.org/en

[36] Oluwatosin, H. S. (n.d.). Client-Server model. *IOSR Journal of Computer Engineering*, *16*(1), 57–71. https://doi.org/10.9790/0661-16195771

[37] Research Rabbit. (n.d.). *ResearchRabbit*. ResearchRabbit. Retrieved April 8, 2024, from https://www.researchrabbit.ai/

[38] Sherman, R. (2015). Project management. In *Elsevier eBooks* (pp. 449–492). https://doi.org/10.1016/b978-0-12-411461-6.00018-6

[39] Software in the Public Interest, Inc. (2024, February 10). *Debian -- the universal Operating system*. Retrieved April 8, 2024, from https://www.debian.org/

[40] Swartz, A. (2013). Aaron Swartz's A Programmable Web: An Unfinished Work. In *Synthesis lectures on data, semantics and knowledge (Print)*. https://doi.org/10.1007/978-3-031-79444-5

[41] *Tailwind CSS - Rapidly build modern websites without ever leaving your HTML*. (n.d.). Tailwind CSS. https://tailwindcss.com/

[42] The Linux Foundation. (n.d.). *Production-Grade Container orchestration*. Retrieved April 8, 2024, from https://kubernetes.io/

[43] The Python Software Foundation. (2024, April 4). *Welcome to Python.org*. Python.org. Retrieved April 8, 2024, from https://www.python.org/

[44] *Uvicorn.* (n.d.). https://www.uvicorn.org/

[45] *Vite*. (n.d.). Next Generation Frontend Tooling. https://vitejs.dev/

[46] *IBM Watson*. (n.d.). Retrieved April 8, 2024, from https://www.ibm.com/watson

[47] *Supabase | the open source Firebase alternative*. (n.d.). Supabase. Retrieved April 8, 2024, from https://supabase.com/

APPENDIX – A: LARGE LANGUAGE MODELS

Large language models are a type of artificial intelligence system that is trained on vast amounts of text data to learn patterns and relationships in human language. They are called "large" because they typically have billions or even trillions of parameters, which are the values that the model uses to make predictions about language.

The training process for LLMs involves feeding them massive datasets of text from various sources, such as books, websites, and databases. The model then learns to recognize patterns in the way words are used and combined to form sentences and paragraphs. This process is known as unsupervised learning, as the model is not explicitly taught rules of grammar or language; instead, it learns these patterns implicitly from the data itself.

Once trained, LLMs can be used for a variety of natural language processing tasks, such as:

1. **Text generation**: LLMs can generate coherent and contextually appropriate text based on a given prompt or input. This can be used for applications like writing assistance, content creation, and creative writing.
2. **Language translation**: LLMs can learn to translate between different languages by being trained on parallel corpora of text in multiple languages.
3. **Text summarization**: LLMs can identify the most important information in a given text and generate concise summaries.
4. **Question answering**: LLMs can understand and respond to questions by drawing upon the knowledge they have acquired from their training data.

5. **Sentiment analysis**: LLMs can analyze the sentiment or emotion expressed in a given text, which is useful for applications like customer feedback analysis and social media monitoring.
6. **Text classification**: LLMs can categorize text into predefined categories, such as spam or non-spam, or identify specific entities like names, locations, or organizations.

One of the key advantages of LLMs is their ability to generalize and transfer their knowledge to a wide range of tasks and domains. Unlike traditional machine learning models that are trained for specific tasks, LLMs can be fine-tuned or adapted to new tasks with relatively little additional training data, making them highly versatile and efficient.

However, LLMs also have some limitations and challenges. They can sometimes generate nonsensical or biased output, especially when prompted with topics or contexts that are underrepresented in their training data. Additionally, their outputs can be inconsistent or contradictory, and they may struggle with tasks that require logical reasoning or common-sense understanding.

Despite these limitations, LLMs have made significant advances in natural language processing and have enabled a wide range of applications across various industries. As the field of AI continues to evolve, researchers are working on improving the robustness, reliability, and interpretability of these models, as well as addressing ethical concerns related to their use.

APPENDIX – B: DEPLOYMENT PROCEDURE

The front end of the web application is built using SvelteKit. SvelteKit includes “adapters” that can modify the built server-side and client-side code to work with a particular platform.

GitHub is a platform used to host source-code. It integrates a Version Control System called Git, the following is adapted from GitHub’s documentation.

“GitHub is a cloud-based platform where you can store, share, and work together with others to write code.

Storing your code in a "repository" on GitHub allows you to:

- **Showcase or share** your work.
- **Track and manage** changes to your code over time.
- Let others **review** your code, and make suggestions to improve it.
- **Collaborate** on a shared project, without worrying that your changes will impact the work of your collaborators before you're ready to integrate them.

Collaborative working, one of GitHub’s fundamental features, is made possible by the open-source software, Git, upon which GitHub is built.

Git is a version control system that intelligently tracks changes in files. Git is particularly useful when you and a group of people are all making changes to the same files at the same time.”

GitHub has integrations with Vercel. Vercel is a platform for deploying web applications that function in a server-less environment, the following is adapted from Vercel’s documentation:

“Vercel is a platform for developers that provides the tools, workflows, and infrastructure you need to build and deploy your web apps faster, without the need for additional configuration.

Vercel supports popular frontend frameworks out-of-the-box, and its scalable, secure infrastructure is globally distributed to serve content from data centers near your users for optimal speeds.

During development, Vercel provides tools for real-time collaboration on your projects such as automatic preview and production environments, and comments on preview deployments."

Vercel requires a compatible front-end framework to properly setup the environment, although it can host simple filesystem based websites without any additional configuration.

Data Storage was carried out using Supabase, an open-source, free (as in freedom, not cost) alternative to Firebase. It uses a PostgreSQL database as its backend. The following is adapted from Supabase Docs.

"Supabase is an open-source Firebase alternative. Start your project with a Postgres database, Authentication, instant APIs, Edge Functions, Realtime subscriptions, Storage, and Vector embeddings."

The connection between the components was carried out using HTTP and HTTPS, utilizing APIs provided by various components of ARA.

The Python code for the runners was hosted on Railway. The following is adapted from Railway Documentation.

Railway is a deployment platform designed to streamline the software development life-cycle, starting with instant deployments and effortless scale, extending to CI/CD integrations and built-in observability.

The GitHub repository was linked to both Vercel and Railway, enabling both to build, compile, publish and deploy all changes as soon as they are committed.