International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 10, Issue 3, pp 67-71, March-2026

https://doi.org/10.47001/IRJIET/2026.103011

Zimo AI: A Full-Stack AI - Powered Browser Extension and Web App

[1]Aditya Ranjan, [2]Ajay Khoje, [3]Atharva Dekondwar, [4]Neil Kashyap, [5]Prof. Shahrukh Shaikh, [6]Prof. Mayuri Narudkar

[1,2,3,4]Student, Artificial intelligence and Machine Learning Diploma, Ajeenkya D. Y. Patil School of Engineering, Charholi, Pune, India

[5]Guide, Artificial intelligence and Machine Learning Diploma, Ajeenkya D. Y. Patil School of Engineering, Charholi, Pune, India

[6]HoD, Professor, Artificial Intelligence & Machine Learning Engineering Diploma, Ajeenkya D.Y Patil School of Engineering, Charholi, Pune, India

Abstract **- This paper presents Zimo AI, a full-stack artificial intelligence system designed to assist users while browsing web content. The system consists of a Chrome Browser Extension based on Manifest V3 and a React-based web application powered by Google's Gemini 2.5 Flash large language model. The objective of the system is to improve user understanding of webpage information by providing intelligent assistance directly within the browsing environment.**

The proposed system provides four key functionalities: webpage summarization, question answering based on page content, image analysis, and automatic FAQ generation. The summarization module converts webpage content into structured summaries with key points, while the Q&A module allows users to interact with the page content through natural language queries. The image analysis module generates descriptions and extracts text from images, and the FAQ generator automatically produces relevant question–answer pairs to enhance content comprehension.

The Chrome extension extracts webpage data using a two-stage DOM extraction process and supports a Bring Your Own Key approach for secure API key storage in the browser. The companion web application extends the system with user authentication, persistent conversation history, multi-turn chat, URL-based summarization, and image analysis through file uploads.

The deployed system demonstrates the practical integration of large language model capabilities within a web browsing environment. By embedding AI-powered assistance directly into the browser and web application, the system improves users' ability to quickly understand webpage content, extract important information, and interact with online resources more effectively. The results indicate that such integration can significantly enhance accessibility, information comprehension, and overall user interaction with web-based content.

Keywords: Artificial Intelligence, Large Language Models, Browser Extension, Webpage Content Summarization, Question Answering Systems, Image Analysis, FAQ Generation, Natural Language Processing, Web Applications, Human–AI Interaction.

I. INTRODUCTION

The modern internet presents users with an overwhelming volume of textual and visual content. Despite decades of browser development, no standard browser provides a native AI layer to help users comprehend, summarize, or query page content. Users who want AI assistance must copy content into a separate tool and manage the interaction manually — a fragmented, slow workflow that interrupts the browsing session.

Zimo AI solves this through two complementary products. The Chrome Extension (Manifest V3) injects an AI panel into any tab, reading page content via a smart DOM parser (content.js) and providing instant AI responses without leaving the page. The companion React Web Application provides the same four features in an authenticated environment with persistent history in MongoDB, multi-turn conversational chat, and URL-based summarization.

Both products use Google Gemini 2.5 Flash — a multimodal model processing text and images in one API call. The Extension calls Gemini directly using the user's API key stored in chrome.storage.local (BYOK model, no intermediate server). The Web App routes all AI calls through a secure Node.js + Express.js backend managing JWT and Google OAuth authentication. The live deployment is at zimoai.vercel.app.

II. LITERATURE SURVEY

2.1 Existing AI Tools

Modern AI platforms such as ChatGPT and Google Gemini provide powerful capabilities for text generation,

International Research Journal of Innovations in Engineering and Technology (IRJIET)
ISSN (online): 2581-3048
Volume 10, Issue 3, pp 67-71, March-2026
https://doi.org/10.47001/IRJIET/2026.103011

summarization, and question answering using advanced natural language processing techniques.

2.2 Limitations of Existing Systems

These platforms mainly operate through chat-based interfaces and often produce unstructured responses. They require users to manually copy and paste webpage content for analysis and do not directly process the currently opened webpage within the browser.

2.3 Research Gap

Most existing tools focus on individual tasks and lack integrated features such as structured summarization, FAQ generation, image analysis, and automatic webpage extraction through a browser extension. Therefore, a unified AI platform that combines these functionalities in a single system is needed.

III. SYSTEM ARCHITECTURE

The proposed system follows a client–server architecture consisting of two main components: a web application and a browser extension. The web application provides users with a centralized platform to access AI-powered tools such as text summarization, question answering, FAQ generation, and image analysis. The frontend of the web application is developed using React along with HTML, Tailwind CSS, and JavaScript to create a responsive and interactive user interface.

The backend of the system is implemented using Node.js and Express, which handle API requests, authentication, and communication between the frontend and AI services. When users submit text, URLs, or images through the web application, the request is sent to the backend server where it is processed and forwarded to the AI processing layer. The AI services analyze the input and generate results such as summaries, answers, or insights, which are then returned to the frontend for display.

In addition to the web application, the system includes a browser extension that allows users to analyze the currently opened webpage directly from the browser. The extension is developed using web technologies such as HTML, CSS, and JavaScript. It extracts the content of the active webpage and sends it to the backend API for processing. The backend then performs AI-based analysis and returns the generated summaries or insights to the extension interface.

This architecture enables efficient communication between the web application, browser extension, backend server, and AI services, providing a scalable platform for intelligent content analysis.

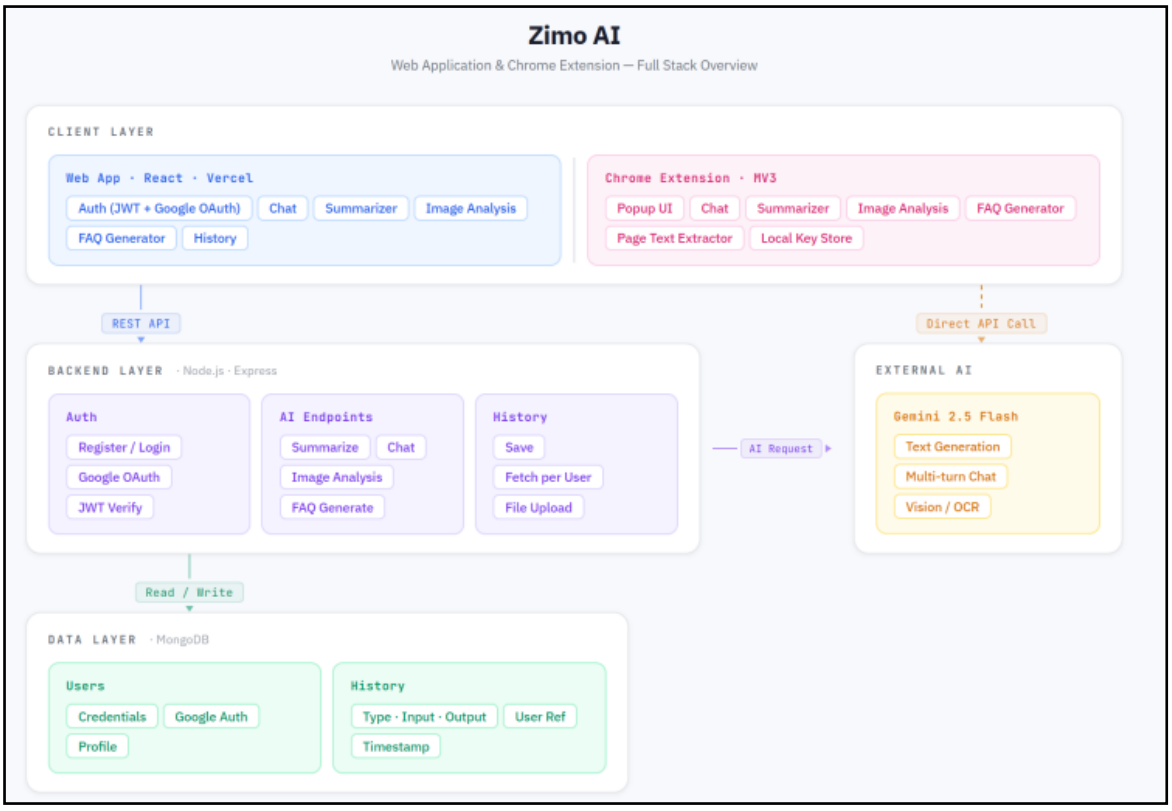

Figure 1: Zimo AI Complete Architecture

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 10, Issue 3, pp 67-71, March-2026

https://doi.org/10.47001/IRJIET/2026.103011

IV. IMPLEMENTATION

4.1 User Interface Design

The system provides an interactive user interface where users can easily access features such as text summarization, question answering, FAQ generation, and image analysis. The interface is designed to allow smooth navigation and efficient interaction with the platform.

4.2 Webpage Content Extraction

The browser extension enables users to extract textual content directly from the currently opened webpage. It automatically detects the main content area, filters out ads and unnecessary elements, and formats the text before sending it to the backend. This allows users to quickly analyze webpages without manually copying text, making the system efficient for summarization, Q&A, and other AI-driven tasks.

4.3 Request Processing and AI Analysis

User input from the web application or browser extension is sent to the server, where it is validated and preprocessed for analysis. The server forwards the input to AI models, which perform tasks such as text summarization, question answering, FAQ generation, and image analysis. The processed results are structured for clarity, optimized for quick delivery, and returned securely to the user interface. The backend ensures efficient handling of multiple requests while maintaining accuracy and reliability.

4.4 Result Generation and Display

After the server completes AI processing, the generated results are sent back to the user interface. The output is organized into clear sections such as summaries, key points, FAQs, or image insights, making it easy for users to interpret. Interactive features allow users to copy, save, or further explore the results. The interface ensures a smooth and responsive experience, displaying information in a visually appealing and structured format. This approach enhances usability and helps users quickly act on the insights provided.

V. TECHNOLOGY STACK

A deliberate architectural choice was made to use Gemini 2.5 Flash as the single AI model across both the Extension and Web App, ensuring consistent quality and a single provider dependency surface. Table 1 provides the complete technology stack used across all system components.

Table 1: Complete Technology Stack

Component	Technology Used
Extension UI	HTML5, CSS3, Vanilla JavaScript ES6+
Extension AI	Google Gemini 2.5 Flash (direct API, BYOK)
Extension Storage	chrome.storage.local (API key + theme)
Web App Frontend	React 18, Vite, React Router v6, Framer Motion
Web App HTTP	Axios with JWT Bearer interceptors
Backend Runtime	Node.js + Express.js
Backend AI SDK	@google/generative-ai (Gemini 2.5 Flash)
File Uploads	Multer memoryStorage, 10 MB max
Authentication	JWT 7-day + bcryptjs salt-12 + Google OAuth
Database	MongoDB + Mongoose, compound index
Deployment	Vercel — zimoai.vercel.app

VI. RESULTS AND DISCUSSION

6.1 Model Selection: Gemini 2.5 Flash

Gemini 2.5 Flash was selected for three reasons: (1) it is natively multimodal, processing text and images in a single API call without a separate vision model; (2) inference latency for typical summarization is 1.5–2.5 seconds, acceptable for an inline browser tool. (3) instruction-following is strong enough to consistently return valid JSON when the output format is explicitly specified — the foundation of the Web App structured output pipeline.

6.2 Comparison with Existing Tools

Table 2 compares Zimo AI against three leading browser AI extensions. Zimo AI is the only system to simultaneously offer FAQ generation, a companion web app, persistent history, Google OAuth, URL summarization, and a BYOK extension model at no cost to the user. All competing tools are subscription-based with no open history or web application.

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 10, Issue 3, pp 67-71, March-2026

https://doi.org/10.47001/IRJIET/2026.103011

Table 2: Feature Comparison with Competing Browser AI Tools

Feature	Zimo AI	Merlin	Sider AI	Monica
Summarizer	Yes	Yes	Yes	Yes
Q&A on Page	Yes	Limited	Yes	Limited
Image Analysis	Yes	No	Limited	Yes
FAQ Generator	Yes	No	No	No
History Storage	Yes	No	No	Limited
Web App	Yes	No	No	No
Google OAuth	Yes	No	No	No
BYOK Extension	Yes	No	No	No
Free to Use	Yes	Freemium	Freemium	Freemium

6.3 Security and Limitations

Security: API keys are securely stored in chrome.storage.local and transmitted only to the Gemini endpoint. User passwords are hashed using bcryptjs with a salt of 12 rounds, and JWT tokens expire after 7 days to enhance session security. CORS is restricted to localhost:5173 and zimoai.vercel.app to prevent unauthorized requests. File uploads via Multer are held in memory and discarded immediately after the AI call to ensure no sensitive data is stored.

Limitations: The DOM extractor may not capture content from JavaScript-rendered Single Page Applications (SPAs). Chat history is limited to the first turn per session, and the browser extension currently supports only Chromium-based browsers.

Future Work: Plans include adding support for Firefox and Edge, enabling full chat persistence across sessions, implementing a Puppeteer-based web scraper for dynamic content, and exploring on-device AI inference for improved privacy and performance.

VII. CONCLUSION

This paper presented Zimo AI, a full-stack AI system deployed in production, comprising a Chrome Browser Extension (Manifest V3) and a React Web Application, unified via a Node.js + Express.js backend and powered by Google Gemini 2.5 Flash. The platform provides Summarization, Q&A Chat, Image Analysis, and FAQ Generation directly within the browser and through an authenticated web interface at zimoai.vercel.app.

Key contributions include two-tier semantic DOM extraction, a BYOK extension model, structured JSON prompting for deterministic AI outputs, dual authentication (email/password and Google OAuth), multi-turn chat with correct Gemini role mapping, Multer-based multimodal image processing, an FAQ accordion with client-side download, and compound-indexed MongoDB history.

Overall, Zimo AI demonstrates that integrating modern multimodal LLM APIs into web and browser-based architectures can significantly reduce the cognitive effort required for users to understand and interact with online content.

ACKNOWLEDGEMENT

The authors express sincere gratitude to Sharukh Shaikh, Guide, Department of Artificial Intelligence and Machine Learning, Ajeenkya D Y Patil School of Engineering, for their invaluable mentorship and guidance throughout this project. Special thanks are also extended to Mayuri Narudkar, Head of the Department of Artificial Intelligence and Machine Learning, for their encouragement and support. The author would like to acknowledge Google for providing access to the Gemini API, and the open-source communities behind React, Vite, Express.js, Mongoose, and Framer Motion for their contributions that made this project possible.

REFERENCES

[1] M. Lewis et al., "BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation," Proc. ACL, 2020, pp. 7871–7880. [Online]. Available: https://aclanthology.org/2020.acl-main.703

[2] P. Lewis et al., "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," Proc. NeurIPS, 2020, pp. 9459–9474. [Online]. Available: https://arxiv.org/abs/2005.11401

[3] A.Radford et al., "Learning Transferable Visual Models from Natural Language Supervision," Proc. ICML, 2021, pp. 8748–8763. [Online]. Available: https://arxiv.org/abs/2103.00020

[4] Google DeepMind, "Gemini: A Family of Highly Capable Multimodal Models," arXiv:2312.11805, 2023. [Online]. Available: https://arxiv.org/abs/2312.11805

ISSN (online): 2581-3048
Volume 10, Issue 3, pp 67-71, March-2026
https://doi.org/10.47001/IRJIET/2026.103011

[5] Google Chrome Developers, "Chrome Extensions Manifest V3 Overview," 2024. [Online]. Available: https://developer.chrome.com/docs/extensions/mv3/

[6] A.Ranjan, "Zimo AI Chrome Extension Installation Guide," Medium, 2026. [Online]. Available: https://medium.com/@adityavranjan/zimo-ai-installation-guide-c9e6c64fffd8

AUTHORS BIOGRAPHY

Aditya Ranjan is a final-year student pursuing a Diploma in Artificial Intelligence and Machine Learning. He served as the Team Lead and Backend Developer for the project. He was responsible for designing the backend using Node.js and Express.js, implementing the AI endpoints, and integrating Google Gemini AI services for functionalities such as summarization, Q&A, FAQ generation, and image analysis. He also supervised the overall system development and deployment of the platform.

Ajay Khoje is a final-year student pursuing a Diploma in Artificial Intelligence and Machine Learning. He served as the UI/UX Designer for the project. He was responsible for designing the client layer and developing responsive and interactive user interfaces using React and Tailwind CSS, ensuring a smooth and intuitive user experience across the web application.

Atharva Dekondwar is a final-year student pursuing a Diploma in Artificial Intelligence and Machine Learning. He served as the Database Developer for the project. He was responsible for implementing database connectivity, managing user data, and developing the user history storage system to track and maintain previous AI interactions.

Neil Kashyap is a final-year student pursuing a Diploma in Artificial Intelligence and Machine Learning. He served as the Authentication and Security Developer for the project. He implemented Google Authorization and user login systems, enabling secure authentication and user account management within the platform.

Citation of this Article:

Aditya Ranjan, Ajay Khoje, Atharva Dekondwar, Neil Kashyap, Prof. Shahrukh Shaikh, & Prof. Mayuri Narudkar. (2026). Zimo AI: A Full-Stack AI - Powered Browser Extension and Web App. *International Research Journal of Innovations in Engineering and Technology - IRJIET*, 10(3), 67-71. Article DOI https://doi.org/10.47001/IRJIET/2026.103011

****

