Development and Testing of Generative AI-Based Browser Extension for Personalized Independent Learning

Aodi Rizky Saputra[1*], Titania Dwiandini[2]

[1,2] Informatics Engineering, Institute Technology and Business Asia Malang, Jl. Soekarno Hatta – Rembuksari No.1 A, Mojolangu, Kec. Lowokwaru, Kota Malang, Jawa Timur 65113, Indonesia

Keywords

Generative Artificial Intelligence; Personalized Independent Learning; Browser Extension; Open-Source AI Model; Prompt Engineering

****Correspondence Email:***

titania.andini@asia.ac.id

Abstract

The rapid growth of generative artificial intelligence (Generative AI) has opened new possibilities for supporting self-directed learning in higher education. Despite this, many students still struggle to grasp academic terminology when studying independently, especially while using online learning resources. This Research and Development (R&D) study aims to develop and test a browser extension that integrates an open-source generative AI model, specifically a large language model (LLM), to assist students in learning academic terminology. The extension was developed following the Software Development Life Cycle (SDLC) framework. Built with the WXT (Next-Gen Web Extension Framework), the system connects to a LLM model via an API and incorporates prompt engineering as well as option-based personalized learning preferences. To verify its functionality and feasibility, both white-box and black-box testing were performed. The results show that the browser extension is feasible to use and can provide personalized explanations, potentially enhancing students' comprehension of academic terminology during independent study. In conclusion, integrating a free, open-source generative AI model into a browser extension developed with WXT, incorporating prompt engineering and option-based personalized learning preferences, it demonstrates the potential to serve as a tool that supports and engages students in self-directed learning, particularly in understanding academic terminology.

1. Introduction

Digital transformation in education has redefined how students access, process, and construct knowledge. Along with this transformation, the ability to engage in self-directed learning has become an essential skill in higher education, enabling learners to take greater responsibility for their own learning paths. However, despite having abundant online resources, many students still encounter difficulties in understanding complex academic language and specialized terminology.

In practice, several students continue to struggle with comprehending texts written in academic style or containing domain-specific terms, especially in digital-based self-learning contexts. A study conducted by Anwar and Sailuddin (2022) among English as a Foreign Language (EFL) learners at Khairun University, Indonesia revealed that 43% of students experienced difficulties in translating and understanding academic sentence structures as well as complex vocabulary or terminology, which significantly could hindered their reading comprehension. These findings indicate that linguistic complexity remains a major obstacle to independent learning in higher education.

Such challenges stem from multiple factors, including the dominance of low-frequency vocabulary and limited background knowledge of the reading topic. Consequently, these difficulties become more pronounced when students are required to read digital materials such as research journals, international articles, or online learning content without direct instructor guidance. In university settings, students are expected to cultivate a high level of self-awareness and initiative to engage in independent reading, particularly before lectures or while conducting self-directed study.

To bridge the gap between students comprehension and academic terminology encountered during self-directed learning activities, it is necessary to develop a program capable of providing personalized support, particularly in assisting learners to interpret academic terms. In this regard, the integration of AI specifically a free, open-source generative models such as LLM combined with option-based personalization learning preferences and prompt engineering techniques, can be considered a potential solution.

1.1 Literature Review

This section discusses the key concepts from previous studies that form both the theoretical and practical foundation of this research. The reviewed literature covers five interrelated areas, digital literacy of students, personalized independent learning, generative artificial intelligence, prompt engineering, and browser extensions. Each area contributes to understanding and supporting the proposed solution.

Digital literacy has increasingly become a fundamental competency in modern, technology-driven education. It goes beyond basic technical skills and encompasses critical thinking, ethical awareness, and effective communication in digital environments. Existing studies highlight that digital literacy equips students with the ability to navigate information more accurately and responsibly. Dinata (2021) emphasizes that digital literacy strengthens students' capacity to distinguish credible information from misleading content. However, many university students still struggle to comprehend, interpret, and internalize academic texts. This indicates that the core challenge is not merely access to digital tools, but also the cognitive ability to process information deeply and meaningfully.

Building on the need to strengthen digital competence, personalized learning has gained prominence as an approach that adjusts content and pacing to suit individual learner profiles. Yusuf (2024) explains that personalization enables learners to progress according to their preferences, which increases engagement and improves learning outcomes. Furthermore, Hidaya et al. (2024) in Yusuf (2024) observe that self-paced personalization nurtures a sense of ownership, motivation, and autonomy, allowing students to take greater control of their learning process. Supporting this perspective, Novitasari (2023) in Yusuf (2024) reports that personalized learning contributes positively to academic performance. These insights collectively suggest that when learners are provided with flexibility and content tailored to their needs, they are better positioned to engage meaningfully and achieve higher learning effectiveness.

Generative Artificial Intelligence (Generative AI) represents an advanced branch of AI designed to produce new and meaningful content such as text, images, or code based on patterns learned from vast datasets. It extends beyond traditional AI systems that merely classify or predict, offering capabilities that simulate human creativity and reasoning. According to Heryadi (2021) in Arifin et al (2025), AI serves as a foundation for enhancing digital competence through automation and intelligent assistance. Masrichah (2023) further emphasizes that Generative AI has been widely adopted across various sectors to improve efficiency and generate deeper insights. In the educational context, Arifin et al (2025) demonstrate that incorporating

Generative AI into learning environments significantly improves comprehension and engagement by providing contextually adaptive feedback. Similarly, Sharma et al (2025) highlight that Large Language Models (LLMs), as a form of Generative AI, personalize learning by adjusting explanations according to each learner's understanding and needs.

As AI systems rely heavily on user interaction, communication between learners and models becomes essential. This is where prompt engineering serves a pivotal role. According to Heston and Khun (2023), prompt engineering is the process of crafting clear and contextually specific instructions to guide AI outputs effectively. Tassoti (2024)introduced the Five S Prompting Framework—Set the Scene, Be Specific, Simplify Language, Structure Output, and Share Feedback—which helps users produce systematic prompts that lead to accurate and relevant responses.

To operationalize these technologies, browser extensions provide an accessible and flexible platform for implementing intelligent learning tools. Ramadhan and Fauzan (2023) define browser extensions as lightweight programs that expand browser functionality and improve user interaction. Afandi et al (2025) further explain that extensions rely on components such as permissions, content scripts, and background services to interact dynamically with web environments. One of the modern frameworks supporting this development is WXT (Next-Gen Web Extension Framework)—a TypeScript-based, next-generation web extension framework that simplifies extension creation through a modular and maintainable architecture.

Overall, these studies show that digital literacy helps students find and understand information, personalization increases motivation and independence, generative artificial intelligence and prompt engineering provide intelligent and adaptive support, and browser extensions make these tools easily accessible. However, previous research has discussed these aspects separately. This study aims to combine them by developing a Generative AI-based browser extension that personalizes independent learning through preference-based options and prompt engineering. This integration is expected to bridge the gap between students' digital literacy in processing academic material.

2. Research Methods

This study employs a Research and Development (R&D) approach utilizing the Software Development Life Cycle (SDLC) as it's methodological framework, as illustrated in "Fig. 1". The SDLC model was chosen because it offers a structured process for developing software-based solutions. In this study, five stages planning, analysis, design, implementation, and testing were adopted, excluding the maintenance phase typically included in the full six-stage SDLC. These five stages were considered sufficient to achieve the research objectives, ensuring that the development process remained systematic and aligned with the study's scope.

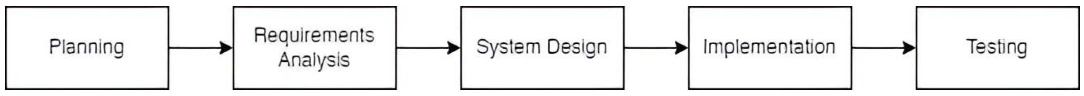


Fig 1. SDLC Methodological Structure

1. Planning

The planning stage focused on addressing students' difficulties in understanding academic terminology, particularly during online self-learning. This issue has been widely recognized as a barrier to comprehending academic texts and domain-specific vocabulary. To respond to this challenge, the project aimed to develop a browser extension that supports personalized and adaptive learning through the integration of a LLM. To achieve this goal, the extension was designed using WXT, a framework suitable for scalable browser extension development. The selected model, OpenAI's gpt-oss-20b, was accessed via the OpenRouter API to enable seamless interaction between the system and the LLM. Rather than applying fine-tuning, the implementation focused on option-based personalization and prompt engineering, guided by the 5S Framework from previous studies to enhance contextual adaptability and user alignment. To maintain focus and feasibility, the project scope was restricted to the Google Chrome browser. Additionally, white-box and black-box testing approaches were planned to comprehensively assess both the internal logic and external functionality of the program.

2. Requirements Analysis

The analysis stage identified the core user and system requirements based on students' needs in understanding academic terminology during independent learning. The system was designed to deliver interactive and context-aware academic assistance through AI-driven responses and a personalization mechanism tailored to individual learning preferences. Functional requirements included the capability to process user queries, generate adaptive explanations, and manage user-selected options that guide the AI's response. Furthermore, this stage examined how prompt engineering would serve as the bridge between user preferences and the language model's output, ensuring that each generated response could aligned with the learner's comprehension level and academic context. Through this design, the system aimed to provide not only accurate information but also an adaptive learning experience that supports self-directed study.

3. System Design

The system design stage translated the conceptual framework into a structured model that defines both the interaction flow and the user interface layout. The design emphasized a seamless connection between user inputs, personalization preferences, and AI-generated academic explanations. As illustrated in "Fig. 2", the system flow begins with retrieving the previously saved / default personalization preferences from the Extension Storage to ensure that the user's customized settings remain consistent across domain. Once this initialization process is complete, the system proceeds to receive the primary Input Data from the user. At this point, the flow enters the Adjust Option decision phase, where the system determines whether the user wishes to modify their current personalization settings. If adjustments are required, the user navigates to the Selecting Option stage within the extension interface to choose new learning preferences—such as explanation style, or difficulty level—which are then saved and stored. When the personalization settings are finalized, the process advances to the Start Generating stage, where the system integrates the Input Data with the confirmed options to construct an engineered prompt that is subsequently sent to the LLM for processing. The generated output is then evaluated. If an error is detected, the system returns an Error Response, otherwise it produces a Success Response, signifying that the explanation or learning support has been successfully generated based on the user's personalized preferences.

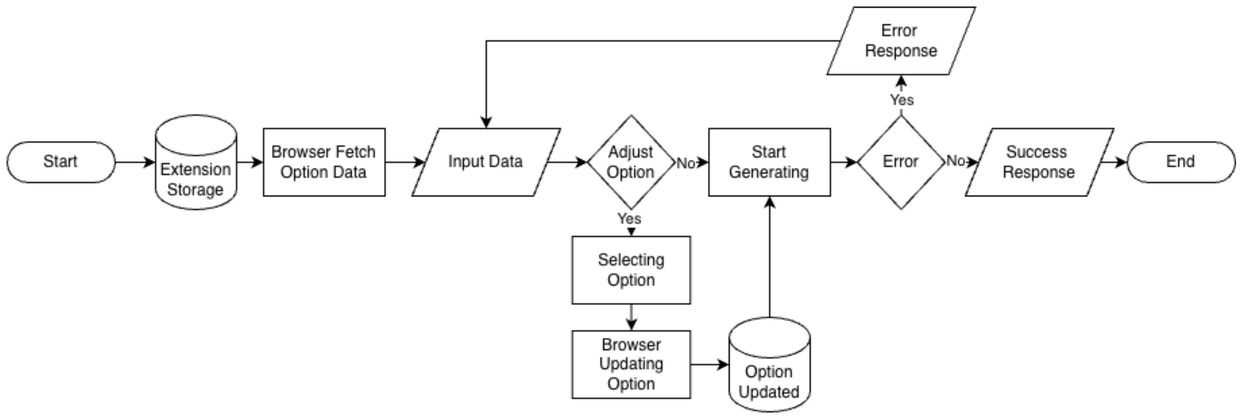


Fig 2. Flow Representation of User and System Processes

These preferences communicates with the gpt-oss-20b model through the OpenRouter API. The language model then could produces contextually relevant explanations, which are displayed directly on the browser interface to support students' comprehension while they read academic materials online.

In parallel, the user interface was designed with a focus on accessibility and ease of interaction, as illustrated in "Fig. 3". The wireframe layout enables learners to set or modify their preferences conveniently without interrupting their browsing activities. The key UI components consist of the user input section, the model response interface, and the learning preferences settings, which facilitate quick and seamless customization of personalized learning support.

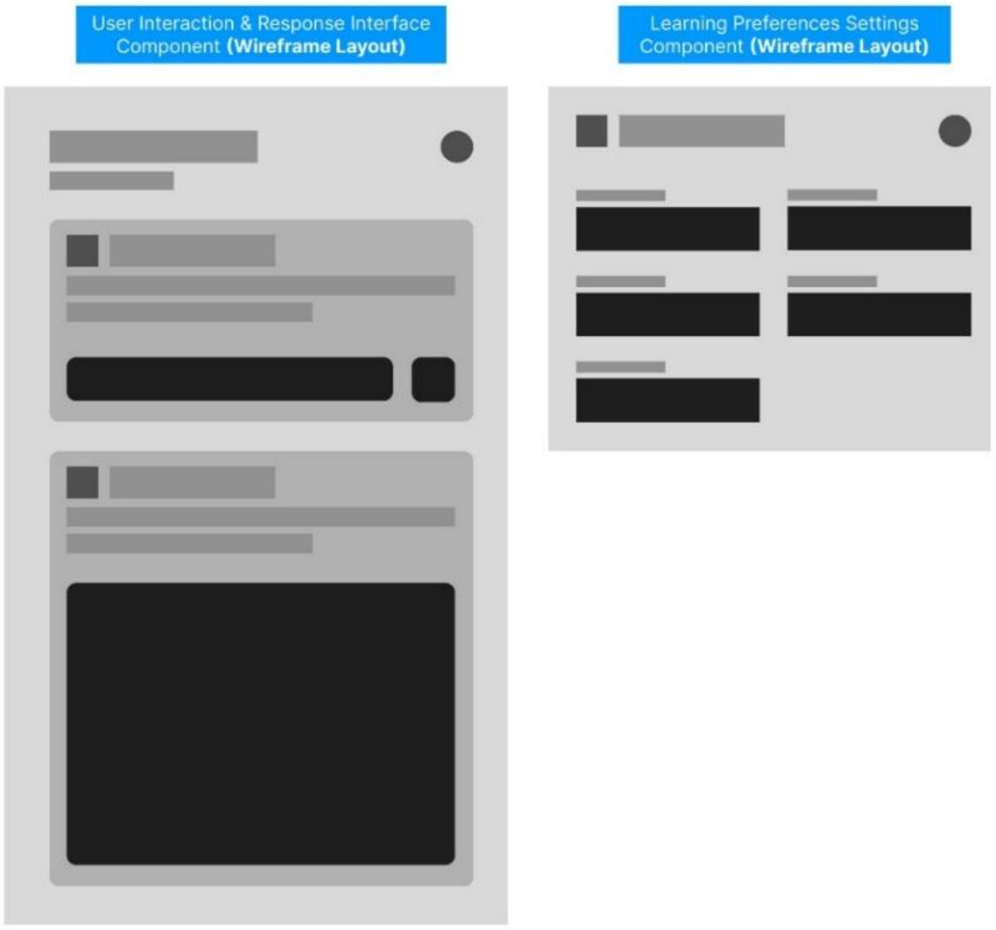

User Interaction & Response Interface Component **(Wireframe Layout)**
Learning Preferences Settings Component **(Wireframe Layout)**

Fig 3. Wireframe Layout of the Browser Extension Interface

4. Implementation

The implementation stage focused on translating the system design into a functional prototype. The browser extension was developed using HTML, TailwindCSS, and TypeScript within the WXT framework, which enables modular and component-based development. The integration with the LLM was established through an API connection to OpenAI's gpt-oss-20b model which from OpenRouter platform, allowing real-time interaction between the user interface and the generative AI engine.

To support personalized learning, option-based personalization and prompt engineering were applied to dynamically adapt AI responses according to user preferences. The personalization mechanism allowed users to select contextual options such as explanation depth, learning difficulty or language, which were then embedded into the system prompts before being sent to the LLM. The backend logic handled these selections to generate coherent and context-aware academic explanations aligned with user-defined parameters.

The final implementation produced a fully operational browser extension capable of retrieving, processing, and displaying personalized AI-generated responses directly in the browser. This prototype served as the foundation for subsequent testing and evaluation phases.

5. Testing

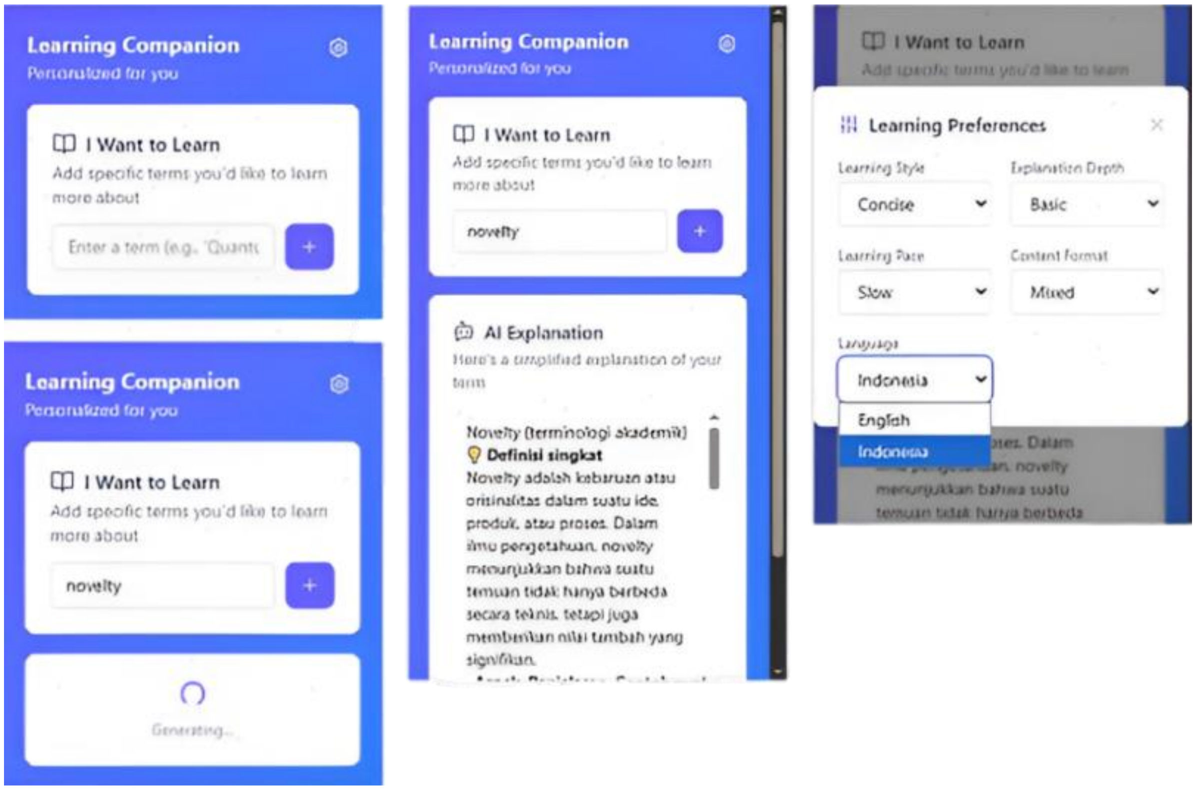
The testing stage aimed to validate the reliability and feasibility of the developed browser extension. Two types of testing were conducted. White-box testing verified the internal logic, data flow, and API integration, confirming the system's technical feasibility and ensuring that each module functioned correctly without execution errors. Meanwhile, black-box testing evaluated the system's adaptive responses based on predefined inputs. This test focused on assessing functional and adaptive feasibility—specifically, whether the browser extension produced contextually accurate explanations aligned with user-selected preferences.

3. Result and Discussion

This section presents the results of the developed browser extension. It consists of two subsections, final prototype and final testing. These evaluations collectively confirm the system's feasibility and identify potential areas for future improvement.

Final Prototype

The final prototype of the browser extension is shown in "Fig. 4". The user interface was developed based on the previously designed wireframe. Where users can input queries or academic terminology they do not understand, configure their learning preferences, and view the generated responses from the model.



Novelty (terminologi akademik)
💡 **Definisi singkat**
Novelty adalah kebaruan atau orisinalitas dalam suatu ide, produk, atau proses. Dalam ilmu pengetahuan, novelty menunjukkan bahwa suatu temuan tidak hanya berbeda secara teknis, tetapi juga memberikan nilai tambah yang signifikan.

Learning Style: Concise
Explanation Depth: Basic
Learning Pace: Slow
Content Format: Mixed
Language: Indonesia

Fig 4. Final Prototype of the Browser Extension Interface

Final Testing

This stage aimed to verify whether the developed program functioned as intended in both its internal logic and external behavior. To ensure comprehensive validation, two complementary methods were applied: White-Box and Black-Box Testing. The White-Box approach examined the internal logic to confirm that all control paths and conditions executed correctly according to the intended design, while the Black-Box approach assessed the system's functionality from the user's perspective by verifying input–output consistency.

To maintain transparency and scientific rigor, this study used a quantitative metric known as the Test Pass Rate (TPR) to evaluate system performance. The TPR represents the proportion of passed test cases relative to the total executed cases and is widely used in software testing to assess reliability and test case effectiveness. As noted by Barraood et al. (2021), testing quality depends on how effectively test cases detect and manage potential errors. The TPR is calculated using the following formula:

$$TPR = \frac{Number\ of\ Passed\ Test\ Cases}{Total\ Number\ of\ Test\ Cases\ Executed} \times 100\%$$

The results of the White-Box Testing are presented in "Table 1", which provides a summary of each test case and its corresponding outcome.

Table 1. Results of White-Box Testing

Test ID	**Scenario**				
	Condition	***Logic Reference***	***Expected Output***	***Actual Result***	***Status***
WB-01	Validate empty input field	Input validation block	System displays warning when input is empty	Warning displayed correctly	Passed
WB-02	Retrieve saved preferences	Option retrieval logic	System loads the last saved preferences from extension storage	Preferences loaded successfully	Passed
WB-03	Update preferences	Update handler	Updated options are stored correctly in the storage	Preferences updated	Passed
WB-04	Construct final prompt using input and preferences	Prompt builder function	Combined prompt = [Input + Preference Options]	Prompt constructed correctly	Passed
WB-05	Send prompt to LLM API and handle response	API request handler	System sends valid request and receives JSON response	Response received successfully	Passed
WB-06	Handle API connection failure	Exception handling block	System shows error message	Error handled properly	Passed
WB-07	Display loading state during payload transmission	UI state management	Loading indicator appears while waiting for API response	Loading animation displayed and removed after response received	Passed

Based on the White-Box test case results, a calculation was performed to determine the TPR:

$$TPR_{white-box} = \frac{7}{7} \times 100\% = 1 \times 100\% = 100\%$$

The calculation yielded a TPR of 100%, indicating that all tested internal logic paths executed successfully without errors, thereby demonstrating high reliability in the system's internal processes. While the subsequent Black-Box Testing results are shown in "Table 2".

Table 2. Results of Black-Box Testing

Test ID	**Scenario**				
	Condition	***Input***	***Expected Output***	***Actual Result***	***Status***
BB-01	User enters valid text and generates explanation	A term or academic terminology	AI generates and displays an explanation based on the default settings	Explanation generated but displayed with	Not Passed

				minor formatting issues	
BB-02	User attempts to generate response with empty input	-	System shows warning "Input cannot be empty"	Warning displayed	Passed
BB-03	User reopens extension after browser restart	-	Previously selected preferences are retained	Preferences persisted correctly	Passed
BB-04	User changes the language preference and generates a response	Language preference set to an available language	AI adapts the explanation according to the selected language setting	Response reflects the selected language setting	Passed
BB-05	User changes content format preferences and generates a response	Content format preference set to an available option	AI adapts to the explanation based on the selected content format setting	Response aligns with the chosen content format	Passed
BB-06	User changes learning style preferences and generates a response	Learning style preference set to an available language	AI adapts to the explanation based on the selected learning style setting	Response adapts correctly to the selected learning style	Passed
BB-07	User changes explanation depth preferences and generates a response	Explanation depth preferences change to "Advanced"	AI provides a detailed explanation based on the chosen depth	Response reflects the selected depth setting	Not Passed
BB-08	User changes learning pace preferences and generates a response	Learning style preferences change to "Fast"	AI responds at a pace matching the selected option	Response reflects the selected learning pace	Passed

Based on the results of the Black-Box testing, a calculation of TPA was also performed to evaluate the overall TPR of the program external functionality. The calculation as follows:

$$TPR_{black-box} = \frac{6}{8} \times 100\% = 0.75 \times 100\% = 75\%$$

The average TPR calculated at 75% indicates that most of the system's functional features operate as expected. Of the eight Black-Box test cases, six passed, while two cases showed minor issues. Specifically, BB-01 experienced a formatting issue that affected the visual display of the generated explanation, and BB-07 did not fully reflect the depth of the selected explanation. A complete summary of all test cases, including White-Box and Black-Box evaluations, is presented in Table 3, which consolidates both.

Table 3. Summary of the Overall Testing Evaluation

Testing Type	**Summary**			
	Total Test Cases	***Passed***	***Failed***	**TPR**
White-Box	7	7	0	100%
Black-Box	8	6	2	75%

The overall TPR of the program was then computed by averaging the success percentages from both testing methods using the following formula:

$$TPR_{average} = \frac{TPR_{white-box} + TPR_{black-box}}{2}$$

Based on the formula above, the calculation result is as follows:

$$TPR_{average} = \frac{100 + 75}{2} = \frac{175}{2} = 87.5\%$$

The 100% TPR from White-Box testing indicates that all control structures and internal processes were executed according to design. The 75% TPR from Black-Box testing reflects that most user-level functions operated as expected, with minor issues identified in some test cases.

4. Conclusions

Based on the implementation and testing results, the developed browser extension was found to be feasible and functional in supporting personalized independent learning. The evaluation recorded a TPR of 100% in White-Box testing and 75% in Black-Box testing, resulting in an overall average of 87.5%. The results show that the system's core components—including option retrieval, preference modification, and LLM-based response generation—executed as implemented, with only minor issues observed during user-level interactions. The program's logic and interface operated as designed, enabling the retrieval and application of user learning preferences.

It should be noted that the current evaluation primarily focused on technical and functional aspects. For future research, incorporating direct user evaluations, particularly with EFL students, is recommended to assess how the system supports comprehension of academic concepts and complex materials. Such user-centered evaluation would provide additional information on the educational impact of the system, showing how personalization influences learners' understanding and overall learning outcomes.

5. References

Anwar, I. W., & Sailuddin, S. P. (2022). Academic Reading Difficulties in Higher Education. Journal of Languages and Language Teaching, 10(2), 309. https://doi.org/10.33394/jollt.v10i2.4849

Arifin, R. W., Alfian, A. N., Sumardiono, S., & Awiliyanto, R. R. (2025). Pengembangan Skill Digital Siswa SMK melalui Pemanfaatan Generatif AI. Jurnal Bhakti Karya Dan Inovatif, 5(1), 63–72. https://doi.org/10.37278/bhaktikaryadaninovatif.v5i1.1072

Dinata, K. B. (2021). ANALISIS KEMAMPUAN LITERASI DIGITAL MAHASISWA COVID-19 proses Pendidikan Matematika Fakultas Keguruan dan Ilmu Pendidikan . Dampak yang mandiri . Salah satu kemampuan yang berperan cukup penting dalam memfasilitasi. 19, 105–119. https://doi.org/10.31571/edukasi.v19i1.

Hanif Abdul Karim Afandi, M. Lazaro Fa. Al-Dzaki, Nurul Qomariasih, & Reza Aulia Wildana. (2025). GuardSurfing : Ekstensi Browser sebagai Alat Bantu Deteksi Website Phishing dengan Metode Klasifikasi XGBoost untuk Deteksi URL Phishing Berbasis Flask Framework. Info Kripto, 19(2), 73–85. https://doi.org/10.56706/ik.v19i2.124

Heston, T. F., & Khun, C. (2023). Prompt Engineering in Medical Education. 198–205.

Masrichah, S. (2023). Ancaman Dan Peluang Artificial Intelligence (AI). Khatulistiwa: Jurnal Pendidikan Dan Sosial Humaniora, 3(3), 83–101.

Ramadhan, R. F., & Fauzan, A. (2023). Pembatasan Internet Berbasis Ekstensi Web pada Chrome Browser. Proceedings Series on Physical & Formal Sciences, 6, 192–199. https://doi.org/10.30595/pspfs.v6i.869

Sharma, S., Mittal, P., Kumar, M., & Bhardwaj, V. (2025). The role of large language models in personalized learning: a systematic review of educational impact. Discover Sustainability, 6(1). https://doi.org/10.1007/s43621-025-01094-z

Tassoti, S. (2024). Assessment of Students Use of Generative Artificial Intelligence: Prompting Strategies and Prompt Engineering in Chemistry Education. Journal of Chemical Education, 101(6), 2475–2482. https://doi.org/10.1021/acs.jchemed.4c00212

Yusuf, B. (2024). Teknologi dan Personalisasi Pembelajaran Pendidikan Islam untuk Generasi Z. 4(4), 277–285.