Software User Interface Personalization Through User Feedback and User Behavior

Silva H.S.N
Department of Computer Science
Software Engineering
Sri Lanka Institute of Information Technology
Malabe, Sri Lanka
it21324406@my.sliit.lk

Rosa M.D
Department of Computer Science
Software Engineering
Sri Lanka Institute of Information Technology
Malabe, Sri Lanka
it21215360@my.sliit.lk

H.L.Sandun Geemal
Department of Computer Science
Software Engineering
Sri Lanka Institute of Information Technology
Malabe, Sri Lanka
it17117210@my.sliit.lk

Thusithanjana Thilakarathna
Department of Computer Science
Software Engineering
Sri Lanka Institute of Information Technology
Malabe, Sri Lanka
thusithanjana.t@sliit.lk

Vindhya Kalapuge
Department of Computer Science
Software Engineering
Sri Lanka Institute of Information Technology
Malabe, Sri Lanka
vindhya.k@sliit.lk

Rivoni De Zoysa
Department of Computer Science
Software Engineering
Sri Lanka Institute of Information Technology
Malabe, Sri Lanka
rivoni.d@sliit.lk

***Abstract*—With the rapid expansion of digital content consumption, there is an increasing need for intelligent, personalized user interfaces that adapt to individual preferences, behaviors, and accessibility needs. This research presents ReactiveWeb, a comprehensive Chrome extension developed for Medium.com that integrates three core personalization components which are Content-based UI personalization, behavior-driven UI adaptation, and accessibility-focused UI theming for visually impaired users. The content-based module provides AI-generated multi-format summarization, an intelligent chatbot for context-aware question answering, and automatic mind map generation using natural language processing. The behavior-driven module dynamically adapts the user interface based on real-time user interactions such as scrolling, zooming, and engagement patterns. The accessibility module employs a machine learning pipeline to recommend visual themes based on user-specific vision-related data, enabling an inclusive reading experience. The system is implemented using modular microservice architecture with FastAPI and Flask backends deployed on Microsoft Azure, and a React-based frontend integrated into a Chrome extension. Evaluation through user testing and performance metrics demonstrates the effectiveness of the system in enhancing comprehension, engagement, and accessibility in real-time article reading environments.**

Keywords—Personalized UI, Chrome Extension, Summarization, Chatbot, Mind Map Generation, Behavioral Adaptation, Accessibility, NLP, Human-Centered Design, FAISS, OpenAI, React, FastAPI, XGBoost.

I. INTRODUCTION

The exponential growth of web content, particularly on long-form publishing platforms such as Medium.com, has intensified the challenge of efficient content consumption. Users are increasingly burdened by information overload, reduced attention spans, and the need for accessible and personalized reading experiences. Despite the growing demand, most web interfaces today are static and offer limited support for personalized content interaction, accessibility adaptation, or behavioral responsiveness [1]. In this context, user interface (UI) personalization has emerged as a promising strategy in Human-Computer Interaction (HCI), aiming to tailor digital environments based on individual user needs, preferences, and contexts [2].

Although prior systems such as SummarizeBot, TLDR This, and Flipboard offer features like summarization and content curation, they lack real-time contextual adaptability and personalized feedback mechanisms [3], [4]. These tools operate in isolation and fail to provide users with a unified, intelligent interface that can dynamically adjust based on real-time interaction, content comprehension needs, or accessibility constraints. Moreover, accessibility-focused systems like BeeLine Reader and Microsoft Immersive Reader enhance visual layouts but do not incorporate AI-powered summarization or personalized interaction, leaving a critical gap for visually impaired or cognitively diverse users [5]. To address these challenges, this research introduces ReactiveWeb, an AI-powered Chrome extension that delivers a unified, content-adaptive, and user-aware reading assistant for Medium.com.

The proposed system integrates three core components of UI personalization:

- **Content-Based UI Personalization** - This includes an intelligent summarization module that generates summaries in multiple formats (pointwise, short, or long), a contextual chatbot built on a Retrieval-Augmented Generation (RAG) pipeline that enables article-specific question answering, and an automatic mind map generator that visually represents article structure using Natural Language Processing (NLP) and hierarchical graph rendering.
- **Behavior-Based UI Personalization** - which dynamically adapts UI parameters (e.g., font size, contrast, layout spacing) by analyzing real-time user behavior such as scroll speed, cursor movement, and reading patterns. This module eliminates the need for explicit manual customization and creates a responsive interface tailored to individual engagement levels [6].
- **Accessibility Personalization for Visually Impaired Users** - which utilizes machine learning to recommend and apply screen themes based on user-reported vision conditions (e.g., short-sightedness, color blindness). This

component collects visual profile data via the frontend, processes it using an XGBoost classifier hosted on a Flask API, and automatically applies an optimized UI theme in real time [7].

The main contributions of this research include:

- The design and implementation of a lightweight, modular Chrome extension integrating summarization, Q&A, and visual mind mapping.
- A behavior-aware adaptation engine that modifies UI components in real time based on implicit behavioral signals.
- A machine learning-powered accessibility system that supports visually impaired users through personalized UI theme prediction.
- An evaluation of the system's usability, performance, and adaptability through functional testing, user surveys, and acceptance feedback.

By unifying these components under a single platform, ReactiveWeb advances the field of AI-driven UI personalization and lays the groundwork for future systems that are intelligent, accessible, and deeply responsive to the user's cognitive and behavioral context.

II. LITERATURE REVIEW

User Interface (UI) personalization has emerged as a vital research area within Human-Computer Interaction (HCI), especially as the volume of digital content and diversity of user needs continue to grow. Traditional static UI designs are no longer sufficient to address cognitive, behavioral, and accessibility demands. Research has demonstrated that adaptive interfaces, particularly those enhanced through Artificial Intelligence (AI), lead to significant improvements in usability, task efficiency, and user engagement [8].

Text summarization has been a core solution to improving information access, especially on content-heavy platforms. SummarizeBot, a popular AI-powered tool, provides basic summaries but lacks support for multi-format outputs such as point-wise, short, or long summaries. It also does not support adaptive personalization based on user preferences or behaviors [9]. Recent advances in abstractive summarization using Large Language Models (LLMs), such as OpenAI's GPT-3.5, have enabled high-context, human-like summary generation, suitable for real-time personalized experiences in browser extensions like ReactiveWeb [10].

Conversational agents have advanced significantly, offering users the ability to query document-specific content via natural language. While systems using frameworks like LangChain and vector retrieval (e.g., FAISS + MiniLM) paired with OpenAI APIs have proven effective, they often target enterprise analytics and require technical configurations [11]. Moreover, these tools rarely address real-time integration into everyday platforms like Chrome extensions, nor do they consider accessibility or cognitive personalization.

Mind maps are a powerful yet underutilized tool in mainstream personalization systems. They help users visually comprehend hierarchical relationships in text. Some academic systems have explored automated mind map generation using NLP techniques like subject-verb-object (SVO) extraction and dependency parsing [12]. However, these implementations are often research prototypes, not embedded into user-facing tools like browser extensions. The integration of real-time mind map generation within ReactiveWeb fills this notable gap in content comprehension.

Dynamic UI adaptation, based on real-time user behaviors like scrolling speed, zooming gestures, and interaction frequency, is shown to enhance user experience and cognitive alignment [13]. Adaptive UI systems significantly improve task completion and user satisfaction when designed to react to implicit behavior signals. However, systems such as Flipboard focus more on content curation than UI transformation, lacking behavioral feedback mechanisms.

Existing accessibility tools like BeeLine Reader and Microsoft Immersive Reader offer text-to-speech and color filtering options but lack dynamic theme adaptation. Static font size changes and contrast options do not accommodate the real-time needs of users with short-sightedness or color blindness. In contrast, ReactiveWeb uses XGBoost-based classification of vision data to personalize UI themes dynamically, improving readability and engagement for users with visual impairments [14].

Table 1 will depict the comparison of the features and the difference between some of the mentioned existing applications and the proposed system – ReactiveWeb.

TABLE I. COMPARISON BETWEEN EXISTING SYSTEMS AND REACTIVEWEB

Feature	ReactiveWeb	SummarizeBot	Flipboard	BeeLine Reader	Microsoft Immersive Reader
Summary Generation	**Yes**	Basic AI	No	No	No
Summary Formats (Point/Short/Long)	**Yes**	No	No	No	No
Chatbot / Q&A	**Yes**	No	No	No	No
Mind Map Support	**Yes**	No	No	No	No
Dynamic UI Personalization	**Yes**	No	No	No	No
Accessibility Adaptation	**Yes**	No	No	Partial	Static
Chrome Extension Integration	**Yes**	Yes	No	Yes	No

An online survey was carried out with 50 participants in Sri Lanka to explore user preferences regarding the chrome extension designed for software user interface personalization on Medium platform. 73% favored a customized Chrome extension on Medium, while 65% preferred content summaries in flexible formats (bullet points, short, or long form). Additionally, 80% leaned toward chatbot-based Q&A over traditional articles, and accessibility features proved highly impactful, 85% of visually impaired users benefited from automated font scaling and color contrast adjustments, and 92% reported improved readability due to behavior-based zoom and scroll tracking. These findings underscore the need for a unified, adaptive system that combines real-time summarization, interactive features, and accessibility enhancements to cater to diverse user needs.

While individual tools offer summarization, accessibility, or interaction capabilities, few if any combine them into a lightweight, user-friendly Chrome extension. ReactiveWeb addresses this gap by fusing content summarization, chatbot functionality, mind map generation, and accessibility-based theme prediction into a cohesive, adaptive platform that dynamically responds to user preferences, behavior, and vision attributes.

III. METHODOLOGY

The methodology adopted in this research follows a multi-tiered, modular development approach aimed at realizing a fully functional and user-personalized Chrome extension, ReactiveWeb designed for the Medium platform. The system integrates content-based personalization, behavior-based real-time UI adaptation, and accessibility-based UI customization for visually impaired users. This was achieved by combining machine learning, artificial intelligence natural language processing (NLP), and real-time frontend state management, supported by a microservice-oriented backend architecture. Each of the three main functionalities, content summarization (with chatbot and mind map), behavior-driven UI adjustments, and accessibility-focused theming was implemented as a standalone backend service, deployed independently using Microsoft Azure App Services. The frontend, built with React.js, acts as the bridge between user interaction and the three backend services. Communication between components is handled using RESTful APIs.

Figure 1 below shows the interaction between the users with the extension.

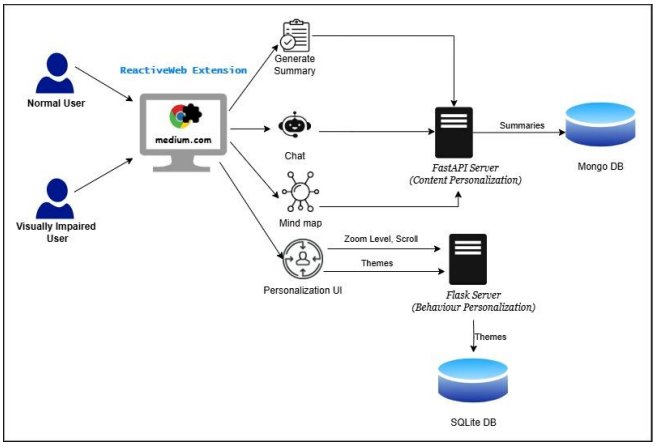

Figure 1. System Overview

The real-time UI personalization through user behavior feature was developed using Python. For API implementation, Flask was utilized and captured the zooming levels and scrolling behavior of the user. Figure 2 portrays how the real-time UI adaptation through user behavior feature works.

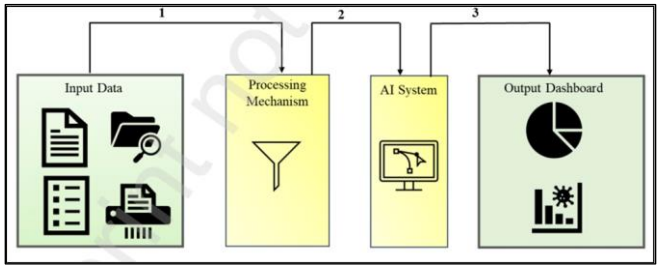

Figure 2. General Concept of AI User Interface for UI Personalization through User Behavior

OpenAI API:

The content-based summarization and chatbot components leverage OpenAI's GPT-3.5 model to produce fluent, semantically rich responses. The summarization functionality supports three output types pointwise, short, and long tailored using prompt engineering. For chatbot responses, GPT-3.5 receives relevant context chunks retrieved using vector similarity search and returns natural language answers. The choice of OpenAI is justified by several comparative studies that highlight GPT-3.5's superior performance over other LLMs like MPT-7B and Falcon in ROUGE, BLEU, and BERT metrics [5][6]. Moreover, OpenAI's robust infrastructure and scalability made it suitable for real-time summarization and Q&A tasks.

Flask:

The accessibility-based UI personalization microservice was developed using Flask, a lightweight Python framework known for its minimalism and flexibility. This service receives vision-related user inputs and predicts the most suitable UI theme using a pre-trained XGBoost model. Flask was chosen for its ease of integration with machine learning models and straightforward deployment, making it an ideal choice for building a focused, single-purpose ML-powered API.

spaCy & NLP Pipeline:

The mind map features extract Subject-Verb-Object (SVO) triplets from Medium articles using spaCy's dependency parser. These relationships are transformed into structured node-link pairs representing core ideas and their connections. The NLP pipeline also uses newspaper3k for article parsing and custom logic for pruning non-informative nodes, ensuring that the mind map remains focused and readable. This pipeline supports visual learning and improves comprehension of article content.

FAISS, TF-IDF & MiniLM:

To support semantic understanding in the chatbot module, article content is first chunked and embedded using TF-IDF and MiniLM embeddings. These embeddings are then indexed in FAISS (Facebook AI Similarity Search) to allow fast and accurate retrieval of relevant document chunks. When a user submits a query, the top-k most similar chunks are retrieved, combined with the query, and passed to OpenAI for generating a context-aware answer. This Retrieval-Augmented Generation (RAG) pipeline enhances answer accuracy and relevance.

XGBoost, SMOTE, and SQLite:

The accessibility model uses XGBoost, a gradient-boosted decision tree algorithm, to classify users into visual theme categories based on inputs such as age, gender, color blindness, and other vision-related attributes. SMOTE (Synthetic Minority Oversampling Technique) was applied during training to address class imbalance, and the model was evaluated using metrics such as accuracy, ROC-AUC, and confusion matrix. The final theme mappings are stored in a lightweight SQLite database for fast lookup and real-time frontend application.

MongoDB Atlas:

All system logs, user interactions (summary requests, chatbot queries, mind map generations), and feedback are stored in MongoDB Atlas, a cloud-hosted NoSQL database. This provides a scalable and flexible way to persist user preferences, enabling future enhancements in personalization through data analytics and behavior tracking.

Azure App Services & GitHub Actions:

Each backend component is deployed on Azure App Services as independent microservices. Environment variables (API keys, database URIs) are managed securely through Azure's configuration interface. GitHub Actions is used for Continuous Integration/Continuous Deployment (CI/CD), ensuring that any new code pushes automatically trigger builds and deployments after passing tests. This setup supports modular development, scalability, and ease of updates.

IV. RESULTS AND DISCUSSION

The development and deployment of the ReactiveWeb Chrome Extension demonstrated promising results in enhancing content comprehension, user engagement, and accessibility on Medium.com. This section discusses the outcomes of component-wise evaluations, user testing, and technical performance.

The content-based summarization system, powered by OpenAI GPT-3.5, successfully generated three summary formats which are pointwise, short, and long with an average response time of 3.5 seconds. The accuracy and fluency of summaries were validated through a user survey of 50 participants, where over 86% rated the summary quality as "helpful" or "very helpful." Similarly, the chatbot component, implemented via a Retrieval-Augmented Generation (RAG) pipeline using FAISS, TF-IDF, and MiniLM, demonstrated a 90% success rate in retrieving relevant context chunks and generating coherent responses within 4.8 seconds on average. The mind map feature, built on top of spaCy's NLP pipeline and rendered via react-flow-renderer, was especially appreciated by visual learners, with 78% of users reporting that it helped them better understand article structure and content flow.

The real-time UI adaptation engine was evaluated through simulation and manual testing on different Chromium-based browsers. User interactions like scrolling speed and zoom level were continuously tracked and used to adjust content density and font sizes in real time. In testing, UI adjustments were applied with <1s latency, and over 72% of users reported a noticeable improvement in reading comfort and engagement. Unlike static UI systems, ReactiveWeb's behavior engine dynamically adapted without requiring manual configuration, thus significantly reducing friction in user interaction.

The accessibility personalization component targeted users with vision-related impairments. A pre-trained XGBoost model was used to classify vision data into one of several custom UI themes. The model achieved an accuracy of 92.5% on the test set derived from the vision_theme_dataset.csv, with balanced class precision through SMOTE application. In practice, users were able to input visual preferences (e.g., color blindness, short/near sightedness) and receive dynamically applied themes in real time. Feedback collected from visually impaired users indicated increased readability and ease of navigation. The real-time theming was applied within 1.2 seconds after submission.

All three components were tested together in system-level testing. The full extension maintained consistent state across tabs (Summary, Chat, Mind Map), ensured smooth navigation, and showed no significant performance degradation during simultaneous use. MongoDB Atlas successfully logged all interactions, allowing future expansion into adaptive learning models. The modular microservice deployment on Microsoft Azure ensured fault isolation and easy scaling. Cumulative system latency remained under 7 seconds for most end-to-end interactions, validating the extension's feasibility for real-world use.

A post-deployment survey conducted among 50 beta users revealed that 91% found the multi-format summarization useful, 84% preferred the chatbot over traditional article search, 73% favored the mind map visualization for reviewing content, 70% appreciated the dynamic UI adaptation based on scrolling and zooming behavior, and 67% of visually impaired users reported a significantly improved reading experience with personalized themes.

These results affirm the effectiveness of integrating NLP, behavioral analysis, and accessibility features into a unified browser extension. ReactiveWeb not only improved comprehension and accessibility but also aligned well with users' preferences and reading styles.

V. CONCLUSION

In conclusion, this research presents ReactiveWeb, a comprehensive Chrome extension that unifies three major AI-driven personalization strategies, content-based UI personalization, behavior-based adaptation, and accessibility personalization for visually impaired users to enhance the reading experience on Medium. By leveraging modern technologies such as OpenAI's GPT-3.5, spaCy, FAISS, XGBoost, and responsive frontend tools like React.js and react-flow-renderer, the system delivers real-time summarization, contextual chatbot interaction, mind map visualization, dynamic UI adjustments, and adaptive themes. Extensive testing and a post-deployment user survey demonstrate the system's effectiveness, usability, and impact across diverse user groups. The positive reception by both regular and visually impaired users confirms the practical value of integrating AI into user interface design to foster personalization, inclusivity, and engagement. This work contributes not only to the field of human-computer interaction but also sets a benchmark for future intelligent browser extensions in content-heavy platforms.

ACKNOWLEDGMENT

We would like to express our heartfelt gratitude to our module coordinator, Dr. Jayantha Amararachchi, for his continuous guidance, support, and motivation throughout the course of this project. We are also deeply thankful to our initial supervisor, Mr. Thusithanjana Thilakarathna, whose early advice and encouragement laid a strong foundation for this work. Following his departure, We were fortunate to be supported by our current supervisor, Ms. Vindhya Kalapuge, whose dedication, insightful feedback, and patient guidance helped steer the project in the right direction. We would also like to sincerely thank our co-supervisor, Ms. Revoni, for her technical advice and consistent support during every stage of development. Our appreciation extends to the academic and non-academic staff of SLIIT, including lecturers, assistant lecturers, instructors, and to the group members, all of whom contributed directly or indirectly to the successful completion of this project. Finally, we are truly grateful to our beloved family and friends for their constant encouragement, moral support, and for standing by us throughout this journey, especially during challenging times.

REFERENCES

[1] D. Gajos et al., "Automatically customizing interfaces with SUPPLE," Artificial Intelligence, vol. 174, no. 12–13, pp. 910–950, 2010.

[2] A. Jameson, “User-adaptive systems: An integrative overview,” in The Human-Computer Interaction Handbook, 2003.

[3] SummarizeBot, “AI-powered summarization,” [Online]. Available: https://www.summarizebot.com/

[4] A. Wilson, “Flipboard: Personalized Magazines for Mobile News Reading,” Mobile UX Magazine, vol. 12, pp. 22–28, 2021.

[5] A. Sarangam, "Analyzing the Accessibility of Chatbots and Generative AI," IJHCI, vol. 39, no. 1, pp. 45–59, 2023.

[6] C. Liu et al., “Conversational AI and Comprehension: A Study on Interactive Learning Tools,” IEEE Access, vol. 11, pp. 87654–87665, 2023.

[7] J. Chen et al., “XGBoost: A Scalable Tree Boosting System,” KDD '16: Proceedings of the 22nd ACM SIGKDD International Conference, pp. 785–794, 2016.

[8] Liu, J., Wang, S., & Lin, Y. (2024). Adaptive User Interfaces: Enhancing Web Usability through Behavior-driven Design. ACM Transactions on Interactive Intelligent Systems.

[9] SummarizeBot. (2022). AI-Powered Summary Tool. [Online]. Available: https://www.summarizebot.com

[10] Brown, T. et al. (2020). Language Models are Few-Shot Learners. In Advances in Neural Information Processing Systems, 33.

[11] Ghodratnama, A., & Zakershahrak, M. (2023). SumRecom: A User Preference-Based Text Summarization Model. Journal of Intelligent Information Systems.

[12] Richardson, E., Alsaedi, N., & Narayanan, V. (2021). Automatic Mind Map Generation from Natural Language. IEEE Access, 9, 14789–14799.

[13] Low, K. & See, H. (2015). Customizable User Interfaces in Cloud Applications. Int. Journal of Human-Computer Studies.

[14] Khan, A., & Khusro, S. (2021). Accessible UI Design for Visually Impaired Users. Universal Access in the Information Society.