HCI Accessibility Principles-Driven Design of AI Programming Assistants for Developers with Special Needs

Yifan Jiang

{b23042004@njupt.edu.cn}

School of Computer Science, Nanjing University of Posts and Telecommunications, Nanjing, Jiangsu, China

Abstract. AI programming assistants are rapidly developing, but they do not meet all the needs of users, especially those with special needs, such as people with disabilities. This study follows the human-computer interaction (HCI) accessibility principle to design an AI programming assistant. In this web prototype, users can select a matching user profile according to their needs after logging in, and then enter a personalized interface. The user profiles are divided into four categories for different needs, and the personalized interfaces for these four categories have different focuses. Visual Impairment Mode emphasizes voice input and output, Mobility Impairment Mode focuses more on hand operations, including keyboard and gesture interactions, Cognitive Support Mode adds detailed guidance steps, and Deaf and Hard-of-Hearing Mode is dominated by visual information. The prototype balances usability and inclusivity and is valuable for integrating accessible design into AI programming aids. See the specific code in: https://github.com/jyf593/HCI-accessibl-web.git.

Keywords: HCI, Accessibility, AI programming assistants.

1 Introduction

Artificial intelligence programming assistants, such as ChatGPT [1] and GitHub Copilot [2], are quickly evolving. The tools leave the developers with a significantly smaller workload by solving programming problems with the help of Large Language Model (LLM) query-based interactions. There is no doubt that such tools greatly reduce the learning and practicing programmer barriers, allowing developers to be more productive and minimize errors. Nonetheless, the current AI-based development tools are not capable of addressing the needs of all users, particularly the physically challenged and the novices in programming. One of the estimates is that almost 15 percent of the global population is disabled, as well as the fact that the number of disabled persons is rising because of the joint impact of population growth

ICIAAI 2026, May 22-24, Singapore, Singapore

DOI 10.4108/eai.22-5-2026.2365079

worldwide and as a result of the improving life expectancy [3]. These tools are unfriendly enough to developers with special needs and cannot fully enjoy AI-assisted programming technologies.

Current studies on Human-Computer Interaction (HCI) and Accessible Computing offer crucial information on improving the inclusiveness of digital tools. According to statistics, even today, the existing AI programming assistants remain poorly usable by the majority of developers. The greatest dilemma is that users find it hard to explain the logic behind the code generated by the model, the failures of the models, and the logic of the relationship between prompts and outputs [4]. Debevc et al., who emphasized that HCI factors can enhance accessibility of m-Learning among persons with special needs [5], summarized several design principles of people with special needs, such as visual magnification, high contrast themes, audio navigation, alternative text, and simplified content structures. These principles can also be applied in the context of programming, which offers systematic design bases of multimodal content presentation, simplified navigation structures, customizable fonts and colors, screen readers, and voice interaction. The majority of products in the market serve the requirements of users with disabilities by enhancing the development environment, e.g., providing customized IDE plugins [6]. As the example of GitHub Copilot is concerned, it requires users to undergo frequent and sudden context switching when using it. But the blind users need a consistent and stable context to work efficiently [7], thus such tools are less popular among them. The existing AI programming assistants seldom take these principles of accessibility into account when they are designed, which leads to the fact that individuals with special needs can barely use them.

To address these issues, this paper suggests a remedy for AI programming assistants to suit developers with special needs. According to the current provisions of the HCI accessibility, the study creates a web prototype with multimodal interaction, customizable visualization, simplified instructions, and keyboard-first navigation to meet the unique user norm requirements of such developers. It will accomplish this through providing both theoretical and practical contributions to the creation of an inclusive AI development environment and equitable access to modern technologies to support a programming environment.

2 System Design and Interface Modules

2.1 User Authentication and Profile-Based Mode Selection

In order to design the web prototypes more suitably, this research is informed by the DevGPT [8], a base of the database created on the prompt response of ChatGPT, along with the writing of code snippets that enable the developers to study how effectively ChatGPT can generate code and solve problems. In a bid to elaborate on the needs, this paper lists four groups of fundamental problems that influence the access experience of special needs developers as revealed in Table 1. The results of the analysis indicate that, in spite of the fact that the available tools are extremely mature in terms of functionality, they are usually deprived of multimodal feedback and accessibility, which gives a clear idea of how the interface should be designed in the future.

To address the problems mentioned, this study developed a web-based AI programming assistant prototype that complies with the accessibility design principles. The core structure of the system is a three-step process of “Login, User Profile Selection, Personalized Interface”,

which ensures that the user is automatically matched with the appropriate mode before entering the system. In order to increase the interpretability of the prototype page, small icons are widely used.

Table 1. Key Usability Issues in Existing AI Programming Assistants and Implications for Accessible Web Interface Design.

Issue Category	Description	Design Implications for Accessible Web Interfaces	Ref.
Transparency	Users cannot understand why specific code is generated or how prompts influence output.	Provide layered explanations and visible context summaries alongside AI output.	[9]
User Control	Fine-grained control requires repeated prompt rewriting	Introduce structured prompt templates and adjustable interaction controls	[10]
Context & Focus	Dynamic updates cause unexpected focus shifts, especially for screen-reader users.	Maintain predictable focus behavior and stable semantic structure	[11]
Interaction & Cognitive Load	Complex interactions increase difficulty for motor and cognitively impaired users.	Simplify workflows through keyboard-first design and progressive disclosure.	[12]

First, there is the login page, as shown in Figure 1(a). Users can use their own account or the demo account when logging in. In the upper right corner of the page, the user can change the theme color, font size, and enable voice announcements through three controls. Meanwhile, in the bottom left corner of the page, the user is prompted to use the keyboard, crucial for those with motor disabilities who rely on keyboard operation.

After logging in, the user can choose a profile according to their situation, as shown in Figure 1(b). The user profile selection interface offers four options: Visual Impairment Mode, Mobility Impairment Mode, Cognitive Support Mode, and Deaf/Hard-of-Hearing Mode. After the selection is completed, each user profile will automatically jump to a personalized page and provide differentiated interaction strategies according to their specific ability needs. Users can select profiles using both mouse clicks and keyboard actions. Pressing the initial letter of the mode you want to select will automatically redirect you to the corresponding page, e.g., pressing 'V' will automatically select Visual Impairment Mode. Similarly, in the upper right corner of the interface, this paper provides the same control as on the login page, which also supports keyboard manipulation by using the tab key to focus on the user's profile.

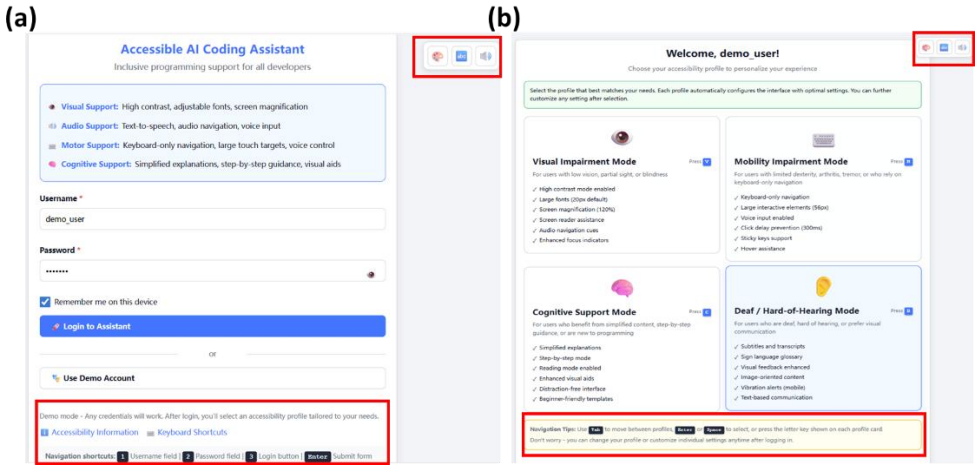

Fig. 1. User login and selection (a) Login Page Details (b) User Profile Selection Interface (Picture credit: Original).

2.2 Visual Impairment Support Mode

Visual Impairment Mode is designed for visually impaired users. Once entered on the page, it will trigger automatic voice announcements, and the header will display the speaker logo, as shown in Figure 2(a). In order to better meet the needs of visually impaired users, the page occupies a large proportion of the screen, and the theme of the page is black on yellow highlight mode. Users can also choose black on white mode and blue on yellow mode, as shown in Figure 2(c)(d)(e). The sidebar on the left side of the page provides users with personalized settings. The Voice Control menu provides users with voice input and output functions, and also adjusts the speed of voice broadcasting. When the user clicks on the voice input control, the header shows a recording logo, as shown in Figure 2(b), which is also applicable to other pages.

The main part includes basic input and output boxes, plus standardized prompts and step-by-step wizards. By clicking on the control in Quick Templates, the system will automatically enter the corresponding prompts into the input box. This part of the design takes into account the fact that some visually impaired people's comprehension is different from that of able-bodied people due to the influence of the environment [13]. The output box is also equipped with a voice announcement, and users can save the output files locally. In addition to this, when the user focuses on a control, this control is highlighted by a purple rectangle for better usability.

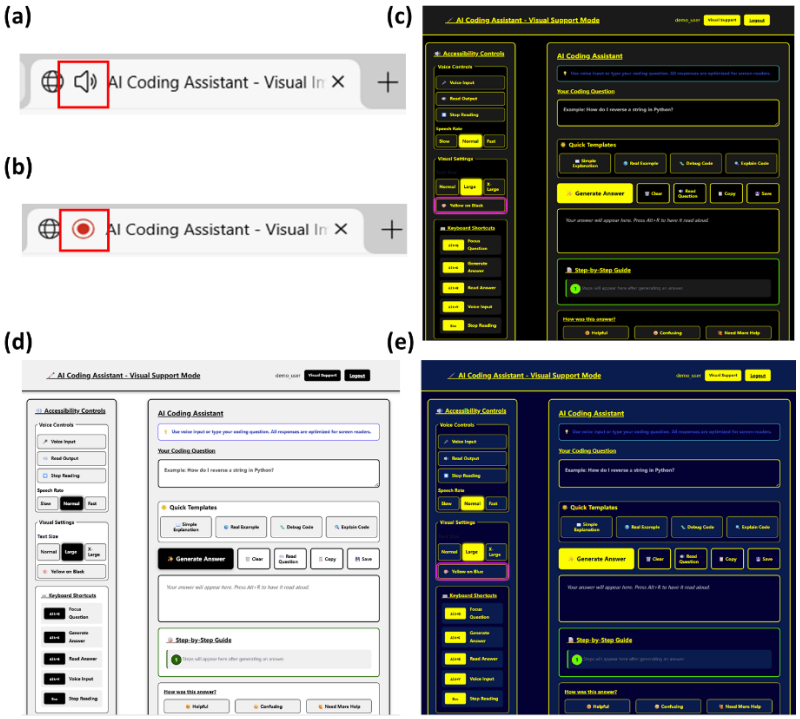

Fig. 2. Visual Support Mode: (a) Voice Output. (b) Voice Input. (c) Yellow on Black Mode. (d) White on Black Mode. (e) Yellow on Blue Mode. (Picture credit: Original).

2.3 Mobility Impairment Support Mode

The Mobility Impairment Mode is proposed to satisfy the interaction requirements of physically challenged (motor disability) developers when using assisted programming. Given that the hand manipulation and the use of a mouse may prove challenging to users, this mode focuses on the use of the keyboard. Simultaneously, the system preserves the voice input and voice output functions and improves them.

In this page, the interactivity of the fundamental multimodal interface brings about the alternative ways of interaction through the provision of alternative forms of control rather than the use of keyboard and voice. The system incorporates a gesture recognition system designed on MediaPipe Hands [14], which detects and analyzes the movements of hands in real-time, as in Figure 3. MediaPipe Hands uses one RGB camera to follow hands in video streams and estimates key points and skeletal location of the hand in the video stream, hence enabling predictive gesture recognition. The Gesture Control control is a control that the user can click on the left to enable the camera to operate in gesture control. There is a menu on the lower-left which suggests hand movements. The fist clenching will mean the voice input, and opening the palm means voice input cessation. In order to avoid false triggering due to unacquired actions or interference with the environment, the system proposes a gesture determination system that utilizes duration. The system will not consider it a valid command unless the user can hold onto the said gesture longer than the preset time. Meanwhile, there is a progress bar at the bottom of the interface that gives real-time feedback concerning the recognition of the action given.

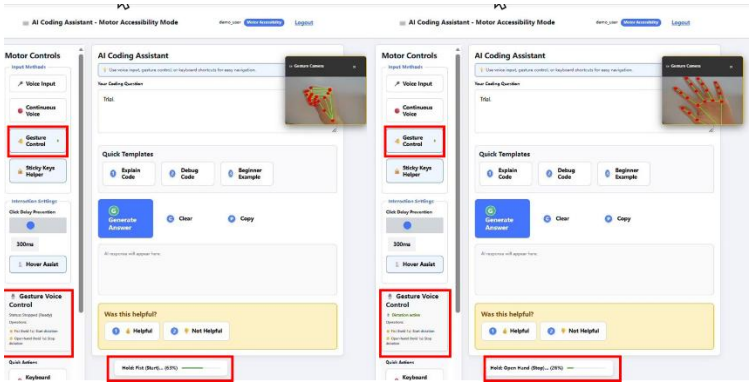

Fig. 3. Motor Accessibility Mode (Picture credit: Original).

2.4 Cognitive Support Mode

Cognitive Support Mode is targeted at beginner developers and at those who may have cognitive disability. The rationale behind the inclusion of these two categories of users in the same module of this study is that they require clear instructions. The output of the model is usually informative in the context of the use of AI programming assistants. This surely makes the cognitive load on the users worse to bear, hence the focus on this page is given to the interpretability and usability of the system. On the left-hand side of the webpage, users can have a Content Simplification menu with various options of simplification to choose, and Figure 4 demonstrates how the page would look once the options have been chosen. Beginner Templates include built-in prompts, which appear automatically in the input box when the user clicks a button, which boosts AI responses. An instructional guide is also used to allow users to navigate the system and AI output with ease.

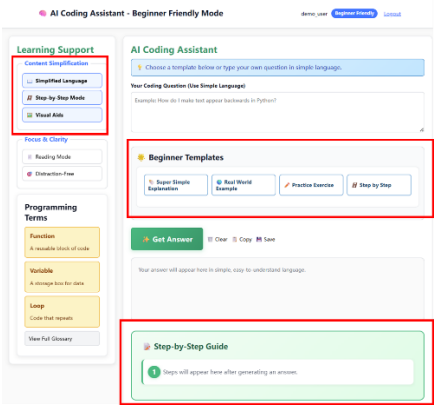

Fig. 4. Cognitive Support Mode (Picture credit: Original).

2.5 Deaf and Hard-of-Hearing Support Mode

In the special requirements of hearing-impaired developers in utilizing AI programming assistants, the current research embarked on developing Deaf and Hard-of-Hearing Mode, a

collection of interactive interfaces that work on the basis of visual information. Because hearing-impaired users are not able to use voice feedback, voice input and output are removed, which was customary in the past. To enhance the flow of the page information, a Full Transcript place is established in the maize interface, as in Figure 5(a). This focus is ever visible without thesaurus of context switching. There exists a Sign Language Support menu bar on the left side of the page, and when the user clicks the Programming Glossary control, a Programming Terms Glossary appears, as seen in Figure 5(b). The glossary puts the commonly used terms in programming in a convenient manner, which the user would find within a time span, and in addition offers relevant sign language resources to support the user's hurdles of understanding the terms.

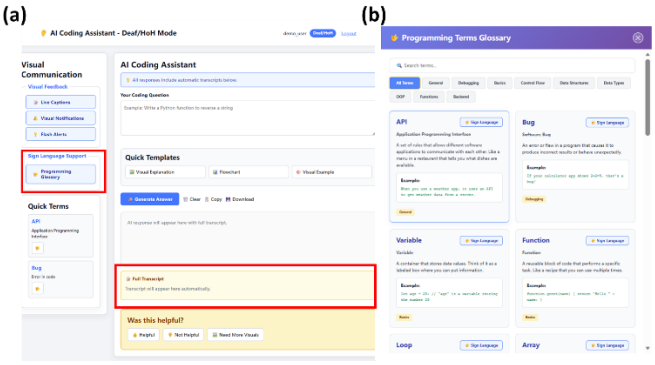

Fig. 5. Deaf/HoH Mode (Picture credit: Original).

3 Conclusions

This paper develops an AI codebase project, which is an assistant web that supports special needs programmers according to the principles of accessibility and human-computer interaction. Through the systematic breaking down of the lack of accessibility of the current AI programming tools, this research turns the abstract principles of accessibility into available interface and interaction strategies and helps serve the divergent needs of the visually, auditory, motor, and cognitively impaired human developers along the lines of user profiling mechanisms. This paper will develop a prototype AI programming assistant for developers who need special needs based on the access design principles and human-computer interaction (HCI) theory. Through the systematic examination of the gaps of current AI programming tools in accessibility terms, the paper will concretize the abstract accessibility principles into the concrete layout interface patterns and interaction strategies, and suggest a user portrait-oriented adjustment mechanism for the support of the differentiated needs of visually, auditorally, motor, and cognitively impaired programmers in the programming process. The paper presents design concepts on how HCI can be used in extremely interactive and informationally aggressive situations in AI programming. Nevertheless, there are still limitations in this study, and the prototype system has not been tested on a large scale in a real-world of large number of special-needs developers over an extensive time span due to the nature of users. Further studies can

include more users, which will provide additional verification of the design framework within other development contexts.

References

[1] GitHub, GitHub Copilot: Your AI pair programmer. https://copilot.github.com/ (2023)

[2] OpenAI, ChatGPT. https://chat.openai.com/ (2023)

[3] M. Nweiser, K. Dajnoki, An overview insight into employment of disabilities at the workplace around the world. Anali Ekon. Fak. Subotici 60, 153–171 (2024)

[4] J.T. Liang, C. Yang, B.A. Myers, A large-scale survey on the usability of AI programming assistants: Successes and challenges, in Proceedings of the 46th IEEE/ACM International Conference on Software Engineering, Melbourne, Australia (2024), pp. 1–13

[5] M. Debevc, M. Verlič, P. Kosec, Z. Stjepanovič, How can HCI factors improve accessibility of m-learning for persons with special needs?, in Proceedings of the International Conference on Universal Access in Human-Computer Interaction, Berlin, Heidelberg (2007), pp. 539–548

[6] Mountapmbeme A, Okafor O, Ludi S. Addressing accessibility barriers in programming for people with visual impairments: A literature review[J]. ACM Transactions on Accessible Computing (TACCESS), 2022, 15(1): 1-26.

[7] Flores-Saviaga, B.V. Hanrahan, K. Imteyaz, S. Clarke, S. Savage, The impact of generative AI coding assistants on developers who are visually impaired, in Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems, Yokohama, Japan (2025), pp. 1–17

[8] T. Xiao, C. Treude, H. Hata, K. Matsumoto, DevGPT: Studying developer–ChatGPT conversations, in Proceedings of the 21st International Conference on Mining Software Repositories, Lisbon, Portugal (2024), pp. 227–230

[9] Abdul, J. Vermeulen, D. Wang, B.Y. Lim, M. Kankanhalli, Trends and trajectories for explainable, accountable and intelligible systems: An HCI research agenda, in Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems, Montreal, Canada (2018), pp. 1–18

[10] S. Amershi, D. Weld, M. Vorvoreanu, A. Fourney, B. Nushi, P. Collisson, E. Horvitz, Guidelines for human-AI interaction, in Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems, Glasgow, UK (2019), pp. 1–13

[11] P.T. Chiou, A.S. Alotaibi, W.G. Halfond, Detecting and localizing keyboard accessibility failures in web applications, in Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, Athens, Greece (2021), pp. 855–867

[12] P.A. Kirschner, J. Sweller, F. Kirschner, J. Zambrano, R., From cognitive load theory to collaborative cognitive load theory. Int. J. Comput.-Support. Collab. Learn. 13, 213–233 (2018)

[13] D.M. Cates, M.J. Traxler, D.P. Corina, Predictors of reading comprehension in deaf and hearing bilinguals. Appl. Psycholinguist. 43, 81–123 (2022)

[14] F. Zhang, V. Bazarevsky, A. Vakunov, A. Tkachenka, G. Sung, C.L. Chang, M. Grundmann, MediaPipe hands: On-device real-time hand tracking. arXiv preprint arXiv:2006.10214 (2020)