Designing for All: Lessons from Teaching Accessibility with AI Agents in a Software Engineering Course

Dr. Farzana Rahman, Syracuse University

Dr. Rahman is an Associate Professor of Teaching in the EECS Department at Syracuse University whose research focuses on effective and inclusive pedagogies in undergraduate and K–12 computing education. Her work examines active-learning strategies, project-based instruction, and accessible approaches to teaching next-generation computing that improve student learning, persistence, and engagement. She has received numerous honors for her contributions, including the 2024 Meredith Teaching Recognition Award, the 2023 College Educator of the Year Award from the Technology Alliance of Central New York, the 2025 SU Chancellor's Citation Award, and the 2025 IEEE Region 1 Teaching Award. Dr. Rahman leads multiple federally and industry funded research initiatives supported by NSF, Google Research, NCWIT, Google TensorFlow, and AAC&U. As the department's diversity spokesperson, she directs the Google Research–funded Research Exposure on Socially Relevant Computing (RESORC) program, which has engaged over 200 undergraduates in computing research and skill development. She serves as an Associate Editor for ACM Transactions on Computing Education and as an Associate Program Chair for ACM SIGCSE. Her scholarship has been published in ASEE, ACM SIGCSE, IEEE RESPECT, and IEEE FIE.

Yang Liu, Rochester Institute of Technology

Ph.D. candidate in Computing and Information Science at Rochester Institute of Technology. His research spans machine learning, accessibility-centered computing education, and CS education research. He applies statistical methods to uncover potential biases in human–computer interaction, with particular emphasis on identifying implicit biases that are often overlooked.

Daniel Krutz, Rochester Institute of Technology (COE)

Daniel Krutz is an Associate Professor at Rochester Institute of Technology, Department of Software Engineering and Center for Cybersecurity. Krutz has a secondary appointment as a Courtesy Associate Professor at the University of Florida, Department of Computer & Information Science & Engineering. Krutz is the Director of the Autonomy, WARfare, and Engineering (AWARE) Lab, which supports several externally funded projects for the NSF, NSA and the DOD. Krutz's research interests include Quantum Information Systems, Self Adaptive Systems, and Computing Education. Krutz is the recipient of the NSF CAREER Award (2022). Krutz received a research faculty fellowship at the US Air Force Research Laboratory (AFRL). Krutz continues to advise high level DOD programs in the areas of AI and strategic reasoning.

Designing for All: Lessons from Teaching Accessibility with AI Agents in a Software Engineering Course

Abstract

Accessibility continues to be underemphasized in software engineering education, leaving many graduates underprepared to design inclusive systems. This experience report paper reports on the design, implementation, and outcomes of a senior-level elective course in Accessible Software Engineering that integrates experiential learning with AI agent supported role play and the use of *Accessible Learning Labs (ALL)*, an established and validated platform for teaching accessibility concepts. The course combined hands-on *ALL* modules, structured technical assignments, and customized AI agents that simulated users with visual, cognitive, learning, and motor impairments. Through these activities, students engaged in accessibility-focused requirements gathering, design, implementation, and evaluation across both web and mobile platforms. Analysis of student projects, reflections, and feedback indicates growth in technical accessibility skills, improved ability to apply WCAG guidelines in practice, and increased awareness of human-centered design considerations. This paper does not present a controlled empirical study. Rather, the goal of this experience report paper is to present instructional design decisions, implementation experiences, discuss the pedagogical value of combining experiential labs with AI-mediated requirement analysis, and offers a replicable instructional model for integrating accessibility more deeply into software engineering curricula.

1 Introduction

As software systems increasingly shape access to education, employment, healthcare, and everyday professional and personal work, accessibility has emerged as a critical requirement for equitable digital engagement. Over one billion people worldwide live with some form of disability, yet many contemporary digital systems remain difficult or impossible for them to use independently [10, 15]. Despite the availability of established standards such as the Web Content Accessibility Guidelines (WCAG) [21], accessibility barriers persist across mainstream software products [5, 14]. Prior research has shown that policies and guidelines alone are insufficient to ensure accessible outcomes, particularly when developers lack experience translating policies and standards into concrete design and implementation decisions [5].

A primary challenge here is the limited emphasis placed on accessibility within undergraduate computing education. Surveys of computing faculty indicate that accessibility is rarely treated as

a core topic, and when it is addressed, instruction is often restricted to a brief discussion within a human–computer interaction course [18]. Software engineering curricula, in particular, tend to prioritize correctness, performance, and sustainability while neglecting inclusivity as a fundamental quality attribute. Consequently, computing graduates frequently enter professional practice with strong programming skills but limited preparation to design systems that accommodate diverse user abilities, reinforcing the perception of accessibility as an afterthought rather than an integral design responsibility.

At the same time, recent advances in artificial intelligence (AI), particularly large language models (LLMs), have introduced new possibilities for computing education. Prior studies have demonstrated that conversational AI systems can act as virtual teaching assistants [8, 11], support programming instruction and conceptual understanding [12], and simulate clients or stakeholders in requirements engineering contexts [13]. Within design education, persona-based and role-play interactions with AI systems have been shown to enhance empathy and deepen designers' understanding of diverse user needs [3]. However, the application of AI agents specifically to accessibility education in software engineering—particularly as simulated users representing a range of disabilities—remains largely unexplored.

This experience report paper addresses the need for practical guidance on integrating accessibility into software engineering education by presenting an experience report on the design and delivery of an upper-level elective course in Accessible Software Engineering. The course integrates experiential learning with AI-supported role play to help students engage with accessibility from both technical and human-centered perspectives. Rather than reporting results from a controlled empirical study, this paper documents curricular design decisions, implementation experiences, and reflective observations gathered during course deployment. We provide a detailed account of the course structure, including hands-on labs, accessibility-focused assignments, and the use of AI agents configured as simulated users to support empathy-building and user-centered thinking. Additionally, by presenting a comprehensive account of instructional design decisions and lessons learned, this paper contributes a replicable and extensible model for integrating accessibility into software engineering education. The findings highlight the value of embedding accessibility throughout the development life-cycle and demonstrate how AI-mediated learning and experiential practices can help bridge the gap between abstract accessibility standards and meaningful, user-centered software design.

2 Why Integrate Accessibility into Computing Education?

Accessibility remains underrepresented in most computer science and software engineering programs. Surveys of computing faculty consistently show that accessibility is infrequently taught, and when it is included, coverage is often superficial [10]. This reinforces a narrow view of software quality that overlooks the diverse needs of real users. In practice, inaccessible software can exclude users, reduce autonomy, and create ethical and legal risks for organizations. Teaching accessibility as a core competency aligns with professional standards, industry expectations, and broader societal goals of equity and inclusion. Students need opportunities to understand how design decisions affect users with different abilities and to practice translating guidelines into usable, inclusive systems. Our course addressed this challenge by embedding

accessibility across lectures, labs, and projects rather than isolating it as a standalone topic or module. Accessibility was framed as both a technical discipline—requiring knowledge of standards, tools, and assistive technologies—and a human-centered practice grounded in empathy and user engagement. This integrated approach helped students recognize accessibility as an essential dimension of professional software engineering.

3 Course Design and Structure

We offered an Accessible Software Engineering course as a six-week upper-level elective for senior students in computing-related majors. The course enrolled 24 students and met twice weekly for three-hour sessions. The compressed format necessitated an immersive, hands-on instructional design that maximized student engagement and practical learning.

Rather than relying primarily on traditional lectures, the course followed an experiential learning model. Each week was centered on a specific accessibility theme and included web-based labs designed to simulate real-world accessibility challenges. Topics included color blindness, vision deficiency, speech and auditory accessibility, cognitive and learning disabilities, screen reader interaction, motor and dexterity challenges, and localization for global usability. These labs combined readings, interactive activities, and reflective quizzes to reinforce learning.

A central component of the course was a culminating team project in which students designed and developed a fully functional web or mobile application. Accessibility was treated as a mandatory requirement throughout the project lifecycle. Students conducted accessibility requirement analysis, iterative design, implementation, and evaluation using both automated tools and simulated user feedback. The integration of AI agents into this process allowed students to gather requirements from the perspectives of users with disabilities, ensuring that design decisions were grounded in user needs rather than assumptions. The course content emphasized four interconnected areas:

Accessibility Design Principles: Students were introduced to WCAG guidelines and the POUR framework (Perceivable, Operable, Understandable, Robust). Instruction highlighted the distinction between universal design and accessibility design, emphasizing that inclusive solutions often require nuanced trade-offs tailored to specific user needs.

Coding Tools and Development Practices: Students applied ARIA roles and attributes [20], semantic HTML, alternative text strategies, keyboard navigation techniques, and color contrast analysis. Automated testing tools such as axe-core, WAVE, and Lighthouse were used to identify and remediate accessibility issues.

Assistive Technologies: Students explored screen readers (NVDA, JAWS, VoiceOver), switch devices, voice control systems, and eye-tracking technologies. Emphasis was placed on understanding how assistive technologies interpret code and how design choices affect interoperability.

Accessibility Reporting and Evaluation: Students learned to conduct accessibility audits, interpret evaluation reports, and incorporate peer review and usability testing using simulated or AI-generated personas.

4 Technical Content and Assessment Design

A core goal of the course was to ensure that accessibility concepts were taught not only at a conceptual level, but also through concrete technical practice and structured assessment. The course schedule was carefully aligned so that each technical topic introduced in class was reinforced through targeted assignments, hands-on labs, and reflective activities. This alignment allowed students to progressively build competency in accessible software development while continuously connecting theory to practice.

4.1 Technical Topics Covered

Throughout the six-week course, students engaged with a broad range of technical accessibility topics spanning web and mobile development, using the *Accessible Learning Labs (ALL)*[1] modules Early labs introduced foundational accessibility concerns related to sensory and physical impairments. The course modules have demonstrated their technical and educational proficiency [6, 7, 9, 16, 17] in previous work. Using the [hidden] labs, students were provided with structured, hands-on opportunities to engage directly with accessibility challenges before implementing solutions in their own projects. Rather than introducing accessibility solely through standards or lectures, the *ALL* modules required students to actively experience the effects of inaccessible design—for example, by interacting with interfaces affected by color blindness, localization barriers, cognitive overload, or limited input modalities—and then apply corrective strategies. Each lab followed an active learning sequence that combined background context, guided exercises, reinforcement activities, and reflective assessment, enabling students to connect technical guidelines with lived user experiences. By integrating *ALL* throughout the course, students developed a more holistic understanding of accessibility that encompassed not only how to implement accessible features, but why these design choices matter in practice. Subsequent technical content expanded the scope of accessibility beyond individual impairments. Students studied localization and literacy accessibility, focusing on adapting interfaces for diverse languages, cultures, and reading levels, as well as inclusive design considerations related to identity and gender representation in software systems. These topics emphasized that accessibility extends beyond disability alone and intersects with broader inclusivity concerns. This experiential grounding helped prepare students to meaningfully engage with later activities, including AI-supported role-play and project-based development, where accessibility considerations were treated as integral to software design rather than abstract compliance requirements.

As the course progressed, students transitioned from guided labs to more advanced technical assessment tasks in the form of homework. They learned to conduct systematic accessibility audits using professional tools such as Accessibility Insights for Web, Lighthouse, WAVE, and axe-core [2]. Instruction emphasized semantic HTML, landmark roles, heading structure, tab order, alternative text, color contrast, form labeling, and ARIA usage. Students were also introduced to assistive technologies, including screen readers and voice-based input, and learned how these tools interpret underlying code structures.

The final technical module extended accessibility concepts to mobile application development through a team project. This is the module where student had to work with customized AI agents for requirement analysis and design development of their proposed project. As part of the project,

students evaluated designed Android and iOS based mobile applications and evaluated them using Accessibility Scanner and TalkBack, identifying common mobile-specific issues such as insufficient touch target size, missing content descriptions, poor contrast, and inaccessible custom views. They were required to propose concrete code-level fixes using Android accessibility APIs, reinforcing the need of accessibility principles across platforms.

4.2 Assessment Through Structured Assignments

Student learning was assessed through a sequence of four major homework assignments, each designed to reinforce specific technical competencies while promoting reflection and critical analysis. The first two assignments focused on experiential learning and conceptual grounding. Students completed multiple *ALL* labs and submitted structured reflection reports analyzing the accessibility challenges encountered and lessons learned. These reflections encouraged students to state connections between user experience, design decisions, and accessibility principles.

One homework (HW1) introduced evaluative and analytical skills by requiring students to critique existing websites using established usability and interface design principles. By applying Shneiderman's Eight Golden Rules [19] alongside accessibility considerations, students practiced identifying strengths and weaknesses across low-level implementation details, mid-level interaction patterns, and high-level information architecture. This assignment helped students understand accessibility as an integral component of usability rather than a checklist.

Another homework (HW2) emphasized rigorous accessibility evaluation and reporting. Students conducted both automated and manual accessibility testing on real-world websites, using assistive technologies to uncover issues not detectable by automated tools alone. They documented findings through Usability Aspect Reports (UARs), which required evidence-based analysis, WCAG mapping, severity assessment, and justification. This assignment mirrored professional accessibility audit practices and strengthened students' ability to communicate accessibility issues to technical and non-technical stakeholders.

A third homework (HW3) focused on mobile accessibility, requiring students to analyze Android applications using industry-standard tools. By identifying issues and proposing specific fixes at the code or design level, students demonstrated their ability to use and apply accessibility guidelines in a mobile development context. Optional manual testing with TalkBack further deepened understanding of non-visual interaction flows.

4.3 Summary of Topics, Assignments, and Assessed Skills

The table 1 below summarizes the alignment between instructional topics, associated assignments, tools used, and the specific accessibility competencies assessed throughout the course. This mapping illustrates how technical content and assessment were deliberately scaffolded to support progressive learning in accessible software development. The combination of technical content and progressively complex assessments supported deep learning in accessible software development. Early experiential labs built empathy and awareness, while later assignments demanded technical precision, critical thinking, and professional-quality documentation. By repeatedly engaging with both user perspectives and concrete implementation

details, students learned to view accessibility as a continuous design responsibility rather than a final step. These assessments required students not only to identify accessibility issues but also to propose actionable solutions grounded in standards and best practices. This emphasis on remediation strengthened students' confidence in applying accessibility principles in real-world development settings. The alignment between course topics, assignments, and project work ensured that students exited the course with transferable skills applicable to both web and mobile software engineering contexts.

5 AI Agents as Pedagogical Tools

An important exploration conducted in this paper was the use of AI agents as experiential learning tools. These agents were developed using the institution-provided *Mentor.ai* platform [4], an AI learning agent generator, which securely supports student learning through controlled use of large language models such as GPT-4o, Google Gemini, and LLaMA. The platform allowed instructors to design custom AI mentors with defined personas, constraints, and learning objectives.

Table 1: Alignment of course topics, assignments, tools, and assessed accessibility competencies.

Topic	**Assignment(s)**	**Technologies**	**Accessibility Skills**
Sound and speech accessibility	Lab-Accessibility to Sound and Speech	*ALL*-Lab1	Understanding non-visual communication barriers; designing alternatives to audio-only feedback; awareness of captions/transcripts
Screen reader accessibility	*ALL*-Lab-Accessibility with Screen Readers	*ALL*-Lab 3; screen reader-focused interactions	Semantic structure awareness; importance of labels, headings, and reading order; screen reader-compatible UI design
Dexterity and motor accessibility	*ALL*-Lab-Accessibility to Dexterity	*ALL*-Lab4; keyboard navigation	Keyboard operability; focus management; avoiding precision-dependent interactions
Cognitive accessibility	Lab-Accessibility with Cognitive Impairments	*ALL*-Lab5	Reducing cognitive load; simplifying workflows; clarity and consistency in UI layout
Localization and global accessibility	HW1 + Lab-Accessibility to Localization	*ALL*-Lab9	Cultural/language adaptation; avoiding hard-coded text

Continued on next page

Topic	Assignment(s)	Technologies	Accessibility Skills
Literacy accessibility	HW1 + Lab-Accessibility to Literacy	*ALL*-Lab11	Plain language usage; readability; comprehension across literacy levels
Identity and inclusive design	HW1 + Lab-Accessibility to Identity	*ALL*-Lab 12	Inclusive language; representation; accessibility beyond disability-only models
Usability and accessibility critique	HW2 + Website evaluation	Heuristic analysis; WCAG	Holistic critique skills; linking accessibility to usability, interaction patterns, and information architecture
Web accessibility auditing (structure)	HW2	Accessibility Insights; Lighthouse	Identifying structural issues: landmarks, headings, forms, color contrast, alt text; iterative improvement
Manual accessibility testing	HW2	Assistive tech (screen readers, voice input)	Identifying issues missed by automation; assistive technology analysis; task-based evaluation
Accessibility reporting and communication	HW2 (UARs)	WCAG 2.x; usability aspect reports (UARs)	Evidence-based documentation; WCAG mapping; severity assessment; communicating actionable remediation
Mobile accessibility (Android)	HW3	Accessibility Scanner; (optional) TalkBack	Mobile-specific issues: touch targets, content descriptions, navigation order, contrast, custom views
Accessibility remediation (mobile)	HW3 (Fixes)	Android and iOS accessibility APIs	Translating findings into concrete code-level and design-level fixes

5.1 Role of AI Agents in Learning

The AI agents were designed to simulate or facilitate role-play users with specific disabilities, enabling students to engage in structured role-play exercises focused on accessibility requirements gathering. Each agent adopted a consistent persona, background, and set of accessibility needs. For example, one agent simulated a blind student navigating a university course registration system, while another represented a user with cognitive impairments interacting with a scheduling application. Through conversational interaction with the students, agents described challenges such as unlabeled buttons, inaccessible widgets, excessive cognitive load, or reliance on visual-only cues. Students were encouraged to ask clarifying questions, document requirements, and propose design solutions. These dialogues mirrored real-world stakeholder interviews and helped students practice translating user feedback into actionable engineering decisions.

5.2 Example AI Mentor Personas and Prompts

Below are examples of few AI mentor configurations and prompts we used in the course.

Example 1: Blind User Persona

System Prompt: "You are an AI mentor simulating a blind university student who relies on a screen reader (NVDA) and keyboard navigation. Describe your experiences navigating web applications honestly and consistently. Focus on accessibility barriers and how they affect your independence."

Student Prompt: "Can you walk me through how you register for courses online and where you encounter difficulties?"

Example 2: User with Cognitive and Learning Disabilities

System Prompt: "You are an AI mentor representing a user with cognitive and learning disabilities, including difficulty processing dense information and complex navigation. Explain challenges related to cognitive load, clarity, and consistency."

Student Prompt: "What aspects of this dashboard make it hard for you to understand or complete tasks?"

Example 3: User with Motor Impairments

System Prompt: "You are an AI mentor simulating a user with limited fine motor control who relies on keyboard navigation and assistive input devices. Highlight issues related to timing, focus order, and interaction precision."

Student Prompt: "How do small buttons or drag-and-drop interactions affect your ability to use this interface?"

5.3 Integration into Projects

Insights gathered from interactions between various AI agents and students directly informed student team projects. Teams were required to document accessibility requirements derived from these conversations and demonstrate how their design decisions addressed them. Final projects were evaluated based on WCAG compliance, usability with assistive technologies, and alignment with user-centered requirements identified through AI agent interactions.

5.4 Developed Student Projects

As the culminating component of the course, student teams designed and implemented accessible web or mobile applications that reflected both technical accessibility requirements and human-centered design considerations. Here are a few sample projects developed by students that applied course concepts to real-world design contexts.

One team developed a web-based scheduling application designed to support users with visual and cognitive impairments. The interface emphasized semantic HTML structure, consistent heading hierarchy, and complete keyboard navigability. Based on simulated user feedback, the

team redesigned calendar views to avoid dense visual layouts, introduced proper navigation alternatives, and provided descriptive labels for all interactive elements.

A second team focused on improving the accessibility of an existing Android habit-tracking application. Using Accessibility Scanner and manual TalkBack testing, students identified missing content descriptions, inconsistent navigation order, and small touch targets that hindered non-visual and motor-impaired users.

Another project addressed accessibility challenges in campus navigation by developing a web tool that prioritized keyboard navigation, screen reader compatibility, and clear orientation cues. The application provided step-by-step directions with redundant textual and structural cues rather than relying on visual maps alone. Students incorporated feedback from simulated users with motor and visual impairments, leading to design choices such as simplified interaction flows, predictable navigation patterns, and clear error messaging.

Another team created a task management application aimed at users with learning and literacy-related challenges. The design emphasized plain language, consistent terminology, and minimal visual clutter. Tasks were presented in small, manageable steps, and icons were paired with text to avoid reliance on visual symbolism alone. The project demonstrated how accessibility considerations can improve usability for a broader audience, not only users with formally identified disabilities.

6 Discussion and Lessons Learned

The integration of customized AI learning agents and *ALL* [1] yielded significant pedagogical benefits. The conversational, scenario-driven format transformed accessibility from a compliance-focused checklist into a human-centered design practice. Students reported that interacting with simulated users helped them understand the real-world consequences of design decisions and challenged their assumptions about usability.

From a learning outcomes perspective, students demonstrated strong technical competence in applying accessibility guidelines and tools. More importantly, they developed increased empathy and confidence in advocating for accessibility. The role-play format also introduced a gamified element that enhanced engagement and motivation. Challenges included ensuring that AI personas remained consistent and avoided reinforcing stereotypes. Instructor oversight and careful prompt design were essential to maintain educational integrity.

- **Pedagogical benefits and student learning:** The integration of AI agents yielded significant pedagogical benefits that extended beyond traditional content delivery. By engaging students in authentic, simulated dialogues with users representing diverse disabilities, the course promoted experiential and reflective learning rather than passive consumption of accessibility guidelines. These interactions transformed abstract WCAG principles into concrete, lived experiences, allowing students to directly observe how specific design decisions affect usability, independence, and user autonomy. The conversational nature of the AI agents enabled students to iteratively test assumptions, ask follow-up questions, and refine requirements in real time, closely mirroring professional

requirements engineering practices. Moreover, exposure to a broad spectrum of disabilities—including visual, cognitive, motor, learning, and literacy-related challenges—helped students develop a more holistic understanding of inclusive design. The role-play format also introduced a gamified element that increased engagement, re-framing accessibility from a perceived constraint into an intellectually stimulating and creative design challenge.

- **Technical competency development:** Students demonstrated substantial growth in technical proficiency related to accessible software development. Across assignments and final projects, students effectively applied WCAG-aligned principles, utilized automated and manual accessibility evaluation tools, and implemented accessible interface components using semantic structure, ARIA attributes, and platform-specific accessibility APIs. Final project artifacts consistently passed automated accessibility audits and exhibited meaningful compatibility with assistive technologies such as screen readers and keyboard navigation. Importantly, students moved beyond surface-level compliance by proposing and implementing remediation strategies grounded in both standards and user-centered reasoning, reflecting an ability to apply accessibility knowledge in realistic development contexts.
- **Empathy and accessibility awareness:** Role-play interactions with AI agents supported the development of empathetic awareness by allowing students to engage directly with simulated representations of users' lived experiences. Rather than relying on hypothetical scenarios, students encountered nuanced accessibility challenges articulated through dialogue, which helped humanize abstract requirements. Student reflections indicated an increased appreciation for the social and ethical dimensions of accessibility, as well as greater confidence in identifying accessibility barriers that may not be immediately visible to non-disabled developers. This heightened awareness positioned students to act as accessibility advocates within future team-based and professional environments.
- **Engagement, motivation, and reflection:** Students reported high levels of engagement and satisfaction with the course structure, frequently citing the AI agent interactions as a distinguishing and impactful element of their learning experience. The interactive format encouraged curiosity, sustained attention, and self-directed exploration, particularly when compared to static lectures or checklist-based instruction. Many students described the course as transformative, noting a shift in how they conceptualized software quality and professional responsibility. Reflective components embedded within assignments further reinforced learning by prompting students to articulate insights, challenges, and evolving perspectives on inclusive design.
- **Professional readiness and transfer of learning:** Several students expressed interest in pursuing careers related to accessibility engineering, inclusive user experience design, or socially responsible computing. Others reported plans to incorporate accessibility practices into capstone projects, internships, or workplace settings. The alignment of course activities with industry-relevant tools, workflows, and documentation practices (e.g., accessibility audits and usability reports) supported the transfer of learning beyond the classroom. Collectively, these outcomes suggest that integrating AI-supported experiential learning into software engineering curricula can help prepare students to engage with accessibility as

an ongoing professional responsibility rather than a one-time compliance task.

- **Scalability and instructional sustainability:** The use of AI agents as simulated users presents a scalable approach to accessibility education, particularly in contexts where access to real users with disabilities may be limited by time, cost, or ethical constraints. AI-supported role play allowed all student teams to engage in repeated, individualized requirements-gathering interactions without increasing instructional overhead. This scalability makes the approach suitable for larger class sizes and for institutions with limited resources, while still preserving the pedagogical benefits of user-centered engagement.
- **Consistency in learning experiences:** AI agents enabled a consistent baseline experience across student teams, ensuring that all students were exposed to comparable accessibility scenarios and learning opportunities. Unlike traditional stakeholder interviews, which can vary widely in depth and quality, AI-mediated interactions provided equitable access to core learning experiences. This consistency supported fair assessment while still allowing students to explore diverse design paths and ask open-ended questions.
- **Support for iterative design:** The low-stakes nature of interacting with AI agents encouraged experimentation and learning from failure. Students were more willing to propose tentative ideas, test alternative designs, and revise assumptions without fear of causing harm or discomfort. This iterative cycle aligns closely with modern software engineering practices and reinforces accessibility as an evolving design process rather than a one-time validation step.
- **Integration across the software development lifecycle:** By embedding accessibility considerations into requirements gathering, design, implementation, evaluation, and reporting, the course reinforced accessibility as a continuous concern throughout the software development lifecycle. Students learned that accessibility decisions made early in design have downstream effects on implementation complexity and usability outcomes, fostering systems-level thinking that extends beyond isolated technical fixes.
- **Broadening accessibility and inclusion knowledge:** Course activities intentionally expanded students' understanding of accessibility beyond traditional disability-focused models to include literacy, localization, identity, and cognitive diversity. This broader framing helped students recognize accessibility as an intersectional concern that overlaps with usability, equity, and social justice. As a result, students began to view inclusive design as relevant to a wider range of users and contexts than they had previously considered.
- **Limitations of AI-mediated simulations:** Despite their benefits, AI agents cannot fully replicate the complexity, diversity, and nuance of real human experiences. Some accessibility challenges—particularly those shaped by personal history, cultural context, or intersecting identities—may not be fully captured through simulation. Acknowledging these limitations was an important instructional component, reinforcing the need for future engagement with real users and participatory design methods when possible.
- **Implications for curriculum design and faculty adoption:** The outcomes of this course suggest that accessibility can be meaningfully integrated into existing software engineering curricula without displacing core technical content. The modular structure of labs,

assignments, and AI-supported activities makes the approach adaptable to other courses such as web development, mobile computing, or capstone design. For faculty, the use of AI agents may lower barriers to adoption by providing structured, reusable learning artifacts that support accessibility instruction even in the absence of prior specialization.

- **Emerging challenges and considerations:** While the use of AI agents proved beneficial, the experience also highlighted important considerations. Careful prompt design and instructor oversight were necessary to ensure that AI personas remained consistent, realistic, and did not inadvertently reinforce stereotypes. Additionally, students were explicitly reminded that AI agents serve as pedagogical simulations rather than substitutes for engagement with real users with disabilities. These considerations underscore the importance of positioning AI tools as complements to, rather than replacements for, inclusive design practices grounded in authentic user engagement.

7 Outcomes

The AI agent–based interactions played a central role in how students approached accessibility requirements for their term projects. Rather than relying solely on documentation or hypothetical user stories, students engaged in structured conversations with AI agents representing users with visual, cognitive, learning, and motor impairments. These interactions prompted students to ask clarifying questions, reconsider initial assumptions, and translate user-described challenges into concrete technical requirements. For many teams, this process influenced early design decisions, such as interface layout, navigation flow, and input mechanisms, and continued to inform revisions throughout implementation.

Through repeated engagement with the AI agents, students developed greater confidence in applying accessibility concepts in practice. While WCAG guidelines provided an essential reference point, students often reported difficulty interpreting how specific criteria should be implemented in real systems. The simulated user interactions helped contextualize these standards, allowing students to see how compliance-related decisions affected usability, efficiency, and independence from a user's perspective. As a result, accessibility shifted from a checklist-oriented task to an integral part of design reasoning and problem solving.

Across the course, preliminary analysis of student reflections, project artifacts, and informal feedback suggests students demonstrated measurable growth in technical competence and empathetic awareness. Final projects consistently met accessibility audit requirements and successfully integrated assistive technologies such as screen readers and voice controls. More importantly, students reported a deeper appreciation for the lived experiences of users with disabilities and expressed confidence in their ability to advocate for accessibility in future professional contexts. Student Reflections and Professional Readiness: Feedback from participants highlighted high levels of satisfaction and engagement, with many describing the AI-mediated interactions as transformative. Several students indicated a new or strengthened interest in pursuing careers related to accessibility engineering, inclusive UX design, or socially responsible computing.

8 Acknowledgements

This material is based upon work supported by the NSF under grant #1825023, #2145010, and #2111152.

References

[1] Accessible learning labs. https://all.rit.edu/.

[2] ABASCAL, J., ARRUE, M., AND VALENCIA, X. Tools for web accessibility evaluation. In *Web accessibility: a foundation for research*. Springer, 2019, pp. 479–503.

[3] GU, H., CHANDRASEGARAN, S., AND LLOYD, P. Synthetic users: insights from designers' interactions with persona-based chatbots. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing 39* (2025), e2.

[4] IBL.AI. Mentor ai platform, 2025. Accessed: 2026-01-13.

[5] KELLY, B., SLOAN, D., PHIPPS, L., PETRIE, H., AND HAMILTON, F. Forcing standardization or accommodating diversity? a framework for applying the wcag in the real world. W4A '05, Association for Computing Machinery, p. 46–54.

[6] KHAN, S., MOSES, H., MALACHOWSKY, S., AND KRUTZ, D. Experiential learning in undergraduate accessibility education: Instructor observations. *J. Comput. Sci. Coll. 38*, 8 (jun 2023), 54–68.

[7] KHAN, S., MOSES, H., MALACHOWSKY, S., AND KRUTZ, D. Experiential learning in undergraduate accessibility education: Instructor observations. *Journal of Computing Sciences in Colleges 38*, 8 (2023), 54–68.

[8] KIESLER, N., AND SCHIFFNER, D. Large language models in introductory programming education: Chatgpt's performance and implications for assessments, 2023.

[9] L-MESSERLE, K., MALACHOWSKY, S., AND KRUTZ, D. E. Open questions for empathy-building interventions for inclusive software development. *J. Comput. Sci. Coll. 38*, 8 (jun 2023), 201–213.

[10] LAZAR, J., GOLDSTEIN, D. F., AND TAYLOR, A. *Ensuring digital accessibility through process and policy*. Morgan kaufmann, 2015.

[11] LIU, M., AND M'HIRI, F. Beyond traditional teaching: Large language models as simulated teaching assistants in computer science. SIGCSE 2024, Association for Computing Machinery, p. 743–749.

[12] MA, I., KRONE-MARTINS, A., AND VIDEIRA LOPES, C. Integrating ai tutors in a programming course. In *Proceedings of the 2024 on ACM Virtual Global Computing Education Conference V. 1* (New York, NY, USA, 2024), SIGCSE Virtual 2024, Association for Computing Machinery, p. 130–136.

[13] MAXIM, B. R., BRUNVAND, S., AND DECKER, A. Use of role-play and gamification in a software project course. In *2017 IEEE Frontiers in Education Conference (FIE)* (2017), pp. 1–5.

[14] POWER, C., FREIRE, A., PETRIE, H., AND SWALLOW, D. Guidelines are only half of the story: accessibility problems encountered by blind users on the web. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2012), CHI '12, Association for Computing Machinery, p. 433–442.

[15] RENKER, K. World statistics on disabled persons, 1982.

[16] SHI, W., MALACHOWSKY, S., EL-GLALY, Y., YU, Q., AND KRUTZ, D. E. Presenting and evaluating the impact of experiential learning in computing accessibility education. In *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering: Software Engineering Education and Training* (New York, NY, USA, 2020), ICSE-SEET '20, Association for Computing Machinery, p. 49–60. Distinguished Paper Award.

[17] SHI, W., MOSES, H., YU, Q., MALACHOWSKY, S., AND KRUTZ, D. E. All: Supporting experiential accessibility education and inclusive software development. *ACM Trans. Softw. Eng. Methodol.* (Sept 2023).

[18] SHINOHARA, K., KAWAS, S., KO, A. J., AND LADNER, R. E. Who teaches accessibility? a survey of us computing faculty. In *Proceedings of the 49th ACM Technical Symposium on Computer Science Education* (2018), pp. 197–202.

[19] SHNEIDERMAN, B., AND PLAISANT, C. *Designing the User Interface: Strategies for Effective Human-Computer Interaction*, 4th ed. Addison-Wesley Professional, 2005.

[20] W3C. Accessible Rich Internet Applications (WAI-ARIA) 1.2. Tech. rep., World Wide Web Consortium (W3C), June 2023. W3C Recommendation.

[21] WORLD WIDE WEB CONSORTIUM. *Web Content Accessibility Guidelines 2.2*. W3C, 2023. Recommendation.