Agentic AI for Accessibility Remediation in Learning Management Systems

Ali Mazloum
University of South Carolina
Columbia, SC, USA
amazloum@email.sc.edu

Jose Gomez*
Fort Lewis College
Durango, CO, USA
jagomez@fortlewis.edu

Elie Kfoury
University of South Carolina
Columbia, SC, USA
ekfoury@email.sc.edu

Jorge Crichigno
University of South Carolina
Columbia, SC, USA
jcrichigno@cec.sc.edu

Abstract

Networking instructors increasingly rely on Learning Management Systems (LMSs) to distribute slide decks, PDFs, lab handouts, and other course materials, yet remediating those artifacts for accessibility remains labor-intensive. This paper presents an agentic AI system that improves Blackboard-hosted networking courses through a four-phase workflow: course navigation, Ally-based assessment, document remediation, and re-upload. The design combines parallel document processing for throughput with a three-agent validation pattern for reliability. In an initial evaluation on three networking courses, the system improved Blackboard Ally scores from 73% to 95%, from 82% to 96%, and from 66% to 89%. On a representative course, it collected 92 documents, processed 513 candidate images, used up to 20 concurrent subordinate agents, and completed the semantic remediation stage in about 28 minutes, corresponding to an estimated 20x speedup over a single-worker baseline. The validator accepted 82 of 83 documents without manual intervention. These results suggest that agentic AI can provide practical instructional support for networking education, while exposing a central tradeoff: higher throughput and higher confidence require additional agent invocations and therefore higher operational cost.

CCS Concepts

• **Human-centered computing** → *Accessibility systems and tools*; • **Applied computing** → *Education*; • **Computing methodologies** → *Artificial intelligence*.

*Corresponding author.

SIGCOMM Education Workshop 2026 (A4NE), Denver, Colorado, USA
2026. ACM ISBN 978-x-xxxx-xxxx-x/YYYY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

Keywords

Agentic AI, digital accessibility, multi-agent systems, networking education, learning management systems

ACM Reference Format:
Ali Mazloum, Jose Gomez, Elie Kfoury, and Jorge Crichigno. 2026. Agentic AI for Accessibility Remediation in Learning Management Systems. In *SIGCOMM Education Workshop 2026 (A4NE), August 17, 2026, Denver, Colorado, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Introduction

Digital accessibility in higher education is no longer optional. Academic institutions increasingly rely on Learning Management Systems (LMSs) to deliver course materials, assignments, assessments, and instructor feedback, yet many course artifacts remain inaccessible to students who depend on assistive technologies. This gap is especially consequential in networking courses, where instructors routinely distribute slide decks, PDFs, Packet Tracer files, Mininet topologies, Kubernetes scripts and manifests, screenshots, and lab documentation that are difficult to remediate manually at scale. When these materials contain untagged PDF files, images without alternative text, or malformed heading structures, they reduce the effectiveness of screen readers and alternative formats and create avoidable barriers to participation.

The literature shows that this problem is persistent and structural. Lomellini et al. [9] reviewed 91 sources on accessible and inclusive online learning and found that inaccessible content remains one of the central barriers faced by disabled students in online higher education, with recurring problems

Acknowledgment: This material is based upon work supported by the National Science Foundation under Grants Nos. 2400729 and 2417823.

including inaccessible documents, poor navigation, insufficient alternative text, and multimedia that lack effective accessible equivalents. Blackboard-specific evidence points in the same direction: Aljedaani et al. [1] found substantial accessibility concerns in the Blackboard ecosystem from the student perspective, reinforcing that LMS adoption alone does not guarantee accessible learning experiences.

Two forces motivate this work. First, accessibility is a compliance requirement. In 2024, the United States Department of Justice published a final rule under Title II of the Americans with Disabilities Act that requires public entities to ensure accessible web content and mobile applications [12]. For public universities, these obligations turn accessibility remediation into an operational necessity rather than a best-effort activity. Second, Blackboard Ally already detects many accessibility problems and reports scores at scale, but it does not fully automate the remediation step [2]. The result is an institutional workflow in which the backlog of flagged files can easily exceed what instructors or staff can fix manually.

The implementation burden is also well documented in the academic literature. Sanderson et al. [11] report that many faculty members lack practical knowledge about how to make digital learning materials accessible and identify a gap between accessibility legislation and day-to-day implementation in higher education. This is especially relevant for remediation tasks such as structuring documents, producing meaningful alternative text, and ensuring compatibility with assistive technologies, all of which require repetitive and detail-oriented work at a scale that is difficult to sustain manually across entire course sites.

At the same time, Laamanen et al. [7] show that detection alone is not enough. They emphasize that higher-education digital services continue to exhibit substantial accessibility errors even under stronger regulatory pressure, and they note an important limitation of automated evaluation tools: such tools can assist with large-scale scanning, but they cannot reliably judge whether content is actually understandable or whether alternative text is semantically useful. This observation is directly relevant to Blackboard remediation because it motivates a system that goes beyond flagging defects and also validates the substance of proposed fixes.

From a networking education perspective, this problem also has a practical instructional dimension. Advanced networking courses often evolve by accumulating lecture slides, protocol diagrams, Packet Tracer scenarios, Mininet topologies, Kubernetes manifests, scanned notes, and lab handouts across semesters. As a result, instructors inherit large collections of heterogeneous files that are pedagogically useful but difficult to remediate systematically. A multi-agent accessibility pipeline can therefore be viewed not only as a systems tool, but also as instructional support infrastructure that helps instructors modernize course delivery, preserve technical depth, and improve inclusive access to networking content without manually revisiting every artifact.

At the same time, the proposed workflow is not intended to be networking-specific in its core architecture. The same orchestration pattern could be applied to other LMS-hosted courses that rely on PDFs, slide decks, and other instructional documents. We evaluate the system in networking education because these courses provide a demanding setting: accessibility often depends on technical diagrams, command-line screenshots, device roles, and dense visual relationships that are harder to describe well than generic documents. We therefore position networking as both the motivating domain and a stress-test for a more general LMS accessibility remediation workflow.

This paper presents a multi-agent system designed to close that gap. The system takes a Blackboard course as input, uses Blackboard Ally to identify low-scoring files, remediates the affected files, and uploads corrected versions back into the course. Rather than relying on a single monolithic agent, the design decomposes the task into four phases and uses specialized agents for remediation. The system also introduces two architectural ideas: a boss/subordinate parallelization pattern to improve throughput, and a three-agent validation pattern to increase trustworthiness before applying automated fixes. The current evaluation is grounded in networking courses, specifically two sections of Advanced Networking and one section of Enterprise Networking Management, which makes the work relevant both as a multi-agent systems contribution and as an experience report on AI-supported course modernization in networking education.

The main claim of the paper is that the value of the system is best understood as a tradeoff among three objectives: improved accessibility outcomes, higher document-processing throughput, and acceptable operational cost.

This paper makes the following contributions:

- A phased system design for Blackboard accessibility remediation that combines local execution, LMS interaction, and document-type-aware repair.
- A multi-agent orchestration strategy that parallelizes document remediation and separates throughput concerns from validation concerns.
- An initial case-study evaluation across three Blackboard courses showing strong gains in Blackboard Ally scores.
- A discussion of the practical tradeoffs among cost, throughput, and reliability in agentic accessibility workflows.

The rest of the paper is organized as follows: Section 2 provides background on LMS accessibility workflows and the role of agentic AI in remediation. Section 3 describes the system architecture and implementation. Section 4 presents the experimental results. Section 5 concludes the paper and outlines future work.

2 Background

Digital accessibility in learning management systems involves three closely related challenges. First, accessibility requirements are defined at the level of user experience: documents, images, navigation structures, and media must all work with assistive technologies and alternative formats. Second, these materials are embedded in broader e-learning ecosystems, where instructors upload heterogeneous files over time and accessibility problems accumulate unevenly across courses. Third, remediation is not only a detection problem. It also requires locating problematic artifacts, repairing them in a file-type-aware way, and checking that the resulting content is structurally and semantically meaningful. Prior reviews of online learning and accessible e-learning environments show that these challenges are persistent and span both platform design and instructional practice [4, 9, 11].

2.1 Accessibility in Blackboard Workflows

For the purposes of this paper, the important Blackboard-specific context is operational rather than motivational. Prior work has already established that accessibility problems persist in LMS ecosystems and that the quality of learning resources depends not only on platform features, but also on how course artifacts are authored and described [1, 4]. Blackboard Ally is valuable because it scans course content against checks derived from standards such as Web Content Accessibility Guidelines (WCAG) [2, 13] and surfaces low-scoring files for instructor attention. However, the resulting workflow remains incomplete: automated evaluation supports large-scale screening, but instructors or support staff still need to inspect files, apply repairs, and redeploy corrected versions [3]. Once a course accumulates dozens of PDFs, slides, and handouts, the challenge is therefore not just finding defects, but managing a remediation pipeline that can move from detection to correction and back into the LMS.

2.2 Agentic AI for Accessibility Remediation

Agentic AI is a good fit for this problem because Blackboard remediation is not a one-step generation task. It is a workflow that includes navigation, file selection, document-type-specific repair, result checking, and course update. Recent agent frameworks treat Large Language Models (LLMs) not just as text generators, but as controllers that can plan subtasks, invoke tools, and operate across multi-step environments [6, 8]. Iong et al. [5] demonstrate that LLM-based agents can be coupled with structured web interaction components to automate browser-facing tasks. These ideas are directly relevant to LMS remediation, where the system must combine language understanding with tool use, document handling, and web navigation.

From this perspective, the goal is not to ask one model to make an entire course accessible in one pass. Instead, the goal is to build a system in which agents can access Blackboard, identify which files require intervention, repair those files according to document-specific accessibility expectations, and return the corrected files to the LMS. The current implementation focuses on PDF, PPTX, and DOCX course materials, combining rule-based structural repair with validated semantic fixes and validated image alt text generation. This shifts the problem from isolated prompting to workflow design, where success depends on coordination among perception, planning, tool invocation, and verification.

2.3 Multi-Agent Design, Parallel Processing, and Cost–Throughput Tradeoffs

Three concepts are central to the proposed system: multi-agent coordination, parallel processing, and the tradeoff between throughput and operational cost. First, the remediation workflow is naturally decomposable into different responsibilities, including Blackboard navigation, document assessment, document repair, and output validation. A multi-agent architecture separates these roles instead of forcing one monolithic agent to handle the full workflow. Recent work on LLM-based multi-agent systems reinforces this design intuition. Yu et al. [14], for example, show that dynamic task decomposition and asynchronous parallel execution can improve throughput for complex agent workflows. Zhou et al. [15] further show that communication structure and prompt design influence the resilience of multi-agent systems, which is directly relevant when automated outputs must be trusted before being applied to course materials.

Second, the dominant cost in wall-clock time occurs during document remediation. Since many files can be processed independently after Blackboard Ally identifies them, the workload is well suited to document-level parallelism. Third, increasing the number of agents reduces runtime but also increases the number of model invocations and therefore the API cost. The same effect appears when extra validation agents are added to improve confidence in the fixes. For this reason, the value of the system is best understood as a balance among three objectives: improving accessibility outcomes, increasing document-processing throughput, and maintaining acceptable operational cost.

3 System Architecture and Implementation

The proposed system remediates Blackboard courses through four phases. Figure 1 summarizes the workflow. The phases were designed to make the workflow explicit and modular

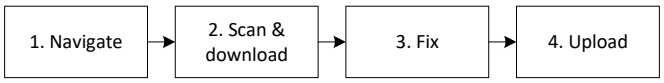

Figure 1: Four phases of the Blackboard accessibility remediation workflow: course navigation, Ally-based assessment and download, document remediation, and re-upload of corrected files.

rather than leaving the entire process to a single unconstrained agent. The implementation was built by providing structured guidance to an AI coding assistant, which produced reusable Python scripts for each phase.

3.1 Phase 1: Navigate

The system begins on a local workstation. It opens a web browser, authenticates into Blackboard, navigates to the target course, and constructs a tree representation of course contents. This phase establishes the inventory of files and locations that later phases depend on. Because Blackboard courses follow similar structural patterns, the system relies on reusable scripts rather than a free-running agent for repeated navigation actions. In this phase, the instructor provides the credentials needed to access Blackboard. Requiring instructor authentication preserves human oversight at the beginning of the process. As a privacy safeguard, the traversal is intentionally restricted so that it does not inspect student rosters, grades, or personally identifying student information, and personal identifiers are excluded or hashed before structured outputs are passed to language-model-based stages.

3.2 Phase 2: Scan and Download

In the second phase, the system navigates to the Blackboard Ally report for the course and extracts the list of low-scoring files. In the current implementation, the collection threshold is set to documents scoring below 85% in Blackboard Ally. Only those files are downloaded to the local workstation for remediation. This choice narrows the workload to the subset of course content for which Blackboard Ally indicates a likely accessibility problem, thereby reducing unnecessary processing.

3.3 Phase 3: Fix

Phase 3 is the core remediation stage. The document modifications themselves are applied locally using installed libraries for PDF, PPTX, and DOCX manipulation, rather than by uploading files to a separate remediation service. This phase is document-type aware: a PDF agent, for example, is expected to ensure tagging, structural organization, title assignment, language declaration, and alternative text coverage for embedded images. The same pattern extends to other file types with corresponding remediation expectations. The implementation also applies practical guardrails: files larger than 20 MB and PDFs longer than 150 pages are excluded from later remediation stages because they exceed the intended processing budget of the current workflow. Figure 2 illustrates this document-level remediation pipeline.

Although Blackboard Ally score improvement is the main external outcome metric in the current evaluation, the remediation process itself is not a black box. Different accessibility issues are handled with different repair strategies depending on the file type and the detected defect. For example, when repairing PDF structure issues, the workflow first checks whether the relevant structural fields are present and populated. If they are missing or incomplete, the system analyzes the document content, infers a section hierarchy from headings such as H1, H2, and H3, and then reconstructs the document structure skeleton accordingly. Similarly, for PowerPoint remediation, the workflow examines the visible content of each slide and assigns a slide title intended to reflect the slide's main topic. These examples illustrate that the system does not apply uniform edits blindly. Instead, it uses issue-specific remediation logic tied to document structure and content cues.

The current implementation is domain-aware mainly through the kinds of artifacts it encounters inside supported document types and through the remediation prompts used to describe them. In networking courses, those artifacts often include topology figures, protocol diagrams, command-line screenshots, and lab illustrations embedded within PDF, PPTX, or DOCX files. The system attempts to preserve their technical meaning by generating structural repairs and alternative text that reflect relationships, labels, and roles visible in the document context. However, the workflow does not yet include specialized parsers or reasoning modules for native networking artifacts such as standalone Packet Tracer files, Mininet topologies, or Kubernetes manifests outside supported document wrappers. In that sense, the current prototype is better understood as a general document-centered accessibility workflow evaluated on technically demanding networking content rather than as a networking-specialized accessibility engine.

The design uses parallelism because document repair dominates total runtime. A boss agent distributes files to multiple subordinate agents, each responsible for remediating a subset of documents independently. Figure 3 shows this orchestration pattern. This fan-out pattern increases throughput by allowing multiple documents to be processed concurrently rather than sequentially.

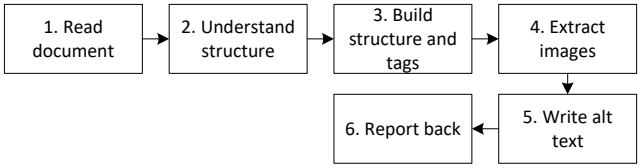

Figure 2: Remediation workflow. For each selected file, the agent reads the document, identifies its structure, builds semantic structure and tags, extracts images, generates alternative text, and returns the remediated result.

The phase also uses a three-agent validation pattern to improve trustworthiness. Two agents independently generate candidate fixes for the same document, and a third agent acts as a validator by checking whether the proposals agree. If they agree, the output is accepted. If they disagree, the task is retried once with fresh generation agents. This design increases confidence in the remediation result, but it also increases API usage and therefore cost.

3.4 Phase 4: Upload

In the final phase, the corrected files are uploaded back into the Blackboard course. This closes the remediation loop and enables the institution to re-run Blackboard Ally on the updated content to verify score improvements. Although the upload step appears straightforward, it required additional engineering because Blackboard allows an instructor to rename a file after upload while Ally may still reference the original filename. Some courses also contained multiple files with identical visible names across different modules. To handle these cases, the upload process uses an additional matching pass to map remediated files back to the correct Ally entries before submission.

3.5 Multi-Agent Remediation

The implemented architecture uses agents not only to generate fixes, but also to separate responsibilities that would otherwise be entangled in a single automation script, as documented in the accompanying repository [10].

The first design pattern is a boss/subordinate hierarchy for document remediation. As shown in Figure 3, the boss agent is responsible for work decomposition and task assignment, while subordinate agents perform the actual document remediation on disjoint subsets of files. This organization enables horizontal scaling with the number of subordinate agents and makes the remediation phase a natural target for throughput optimization.

Agent diversity in this system comes from three sources: model choice, tool access, and workflow role. First, different language models are assigned according to task complexity

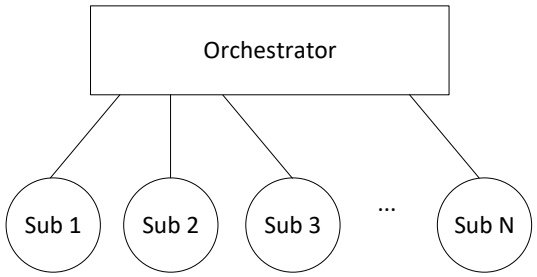

Figure 3: Boss/subordinate orchestration: a central agent distributes remediation tasks across multiple subordinate agents so that documents can be processed concurrently.

and cost sensitivity. The orchestrator, which decomposes the workflow and invokes subordinate agents, uses Claude Opus because it handles planning and coordination, whereas narrower subtasks such as alternative-text generation are delegated to Claude Haiku to reduce cost and latency. Second, agents differ in the tools available to them. For example, the agent that appends alternative text to images invokes Python routines specialized for image-level metadata insertion, while the agent that modifies PowerPoint slide titles relies on a different set of document-processing scripts. Third, agents operate with different task contexts inside the workflow, so the system's multi-agent character is not only a matter of parallel execution, but also of intentionally separating capabilities across planning, document repair, image remediation, and validation.

The subordinate agents are lightweight and specialized by task context. In practice, they operate on specific documents or small batches of documents rather than maintaining a global view of the entire course. This allows the system to exploit concurrency without requiring every worker to reason about the full course state.

The second design pattern is a three-agent validation workflow. Accessibility fixes can appear superficially correct while still failing substantively. During development, one failure mode was observed in which the model filled required semantic fields in a way that satisfied Blackboard Ally's structural checks but left their actual content blank. In that scenario, the system appeared to succeed even though the document had not been meaningfully remediated.

This validation workflow addresses that risk by adding internal redundancy. Two generator agents independently produce candidate fixes for the same document, and a third agent serves as the validator by comparing the two outputs before acceptance. As shown in Figure 4, the system accepts the remediation when the validator determines that the two outputs agree. If the validator detects disagreement, the system retries the task with two fresh generator agents. This

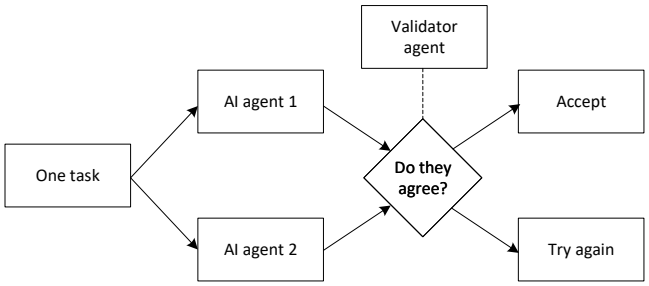

Figure 4: Three-agent validation workflow used in Phase 3. Two agents independently generate candidate fixes for the same document, and a validator compares the results. Matching outputs are accepted; disagreements trigger one retry with fresh agents.

design treats agreement as a proxy for confidence and creates a lightweight review mechanism inside the remediation pipeline itself.

The current arbitration mechanism is intentionally conservative. If disagreement persists after the second attempt, the system does not apply the correction automatically. However, the case is not discarded silently. At the end of the run, the system reports failed tasks to the user in a summary that includes the number of processed images and documents, the number of unresolved problems, and the apparent cause of each failure. In this sense, the current design already supports post-run human awareness, but not yet interactive adjudication. A next step is to expose disagreement explanations more explicitly and allow the course designer to review unresolved cases before deciding whether a proposed correction should be accepted, revised, or rejected.

4 Experimental Results

This section reports the empirical evidence used to assess the proposed system. It first describes the evaluation setup, then presents course-level accessibility improvements, followed by a representative workload and cost breakdown, and finally discusses the tradeoffs among throughput, reliability, and cost.

4.1 Experimental Setup

The system was developed and iteratively refined on ten previously taught courses, spanning five topics and three instructors, and then tested on three currently running networking courses. The evaluation uses Blackboard Ally scores as the primary outcome measure. For each course, the workflow begins with the existing course state, identifies low-scoring files, applies automated remediation, uploads corrected files, and records the updated course score.

Table 1: Reported Blackboard Ally score improvements across evaluated courses.

Course	Initial Score	Final Score
Advanced Networking (Section 1)	73%	95%
Advanced Networking (Section 2)	82%	96%
Enterprise Networking Management	66%	89%

The implementation used in the evaluation is sequenced as follows: First, the system collects Blackboard Ally items scoring below 85%. It then prepares document payloads for semantic validation, prefilters documents that already have acceptable title, language, and heading structure, and orders the remaining tasks largest-first. Semantic validation and image alt-text generation both use the same two-generator plus one-validator pattern, with dynamic concurrency capped at 20 active subagents. When the two generators produce identical outputs, the validator is skipped entirely. When the generators disagree, the validator decides whether the outputs are similar enough to accept; otherwise, the task is retried once and then skipped if disagreement persists. The pipeline is resume-friendly because every stage writes incremental manifests to disk.

The reported runs used 20 concurrent subordinate agents as the per-stage ceiling, with pool semantics so that the next pending document was dispatched as soon as a slot became free. Execution was human supervised. A single operator monitored the pipeline at the console, approved transitions between phases, and could pause or restart any stage when necessary. Before remediated files were uploaded back to Blackboard, each one was rescanned with a helper that checks whether any image descriptor in the PPTX or DOCX remained empty. The system applies hard skip gates at the preparation stage. PDF files longer than 150 pages and any file larger than 20 MB are excluded from the semantic and alt-text pipelines. In the representative courses, this excluded one PDF that exceeded the page cap and two PDFs that exceeded the size cap. An additional eight legacy ‘.ppt‘ files were skipped because the underlying remediation libraries operate only on ‘.pptx‘, and one document failed during image extraction. The remaining 83 documents entered the semantic two-generator plus one-validator pipeline, and the same 20-agent cap was reused for the alt-text stage.

4.2 Accessibility Score Improvements

Table 1 reports the course-level outcomes. These results indicate that the system can substantially improve course-level

Blackboard Ally scores. The gains are strongest for the two Advanced Networking sections, each of which reached at least 95%. The Enterprise Networking Management course improved by 23 percentage points, although it remained below the final scores of the two Advanced Networking sections. This difference suggests that course-specific factors such as document volume, file heterogeneity, or issue severity may influence the final reachable score. In this case, one contributing factor was the presence of hand-written notes uploaded by the instructor, which the current system does not interpret and therefore cannot remediate semantically.

4.3 Representative Course Breakdown

The Enterprise Networking Management course provides a more detailed view of workload, manual burden, and cost under the stricter quality-assurance configuration. Table 2 summarizes the major remediation actions observed in this representative course and explains why the same work is difficult to perform manually at scale.

These estimates show that the remediation workload is large enough to justify automation and structured enough to benefit from parallel processing. In particular, the image-level work alone becomes labor-intensive at 513 candidate images, while metadata and structure repairs require repetitive file-by-file inspection that is difficult to perform consistently across large course sites. The representative course also improved from 66% to 89% in Blackboard Ally and incurred an estimated operational cost of $220, indicating that the practical tradeoff is not simply score improvement, but score improvement relative to instructor labor saved and model cost incurred.

The implementation choices documented by the project help explain why the system remains operational at this scale. Documents that already contain acceptable metadata are prefiltered before entering the semantic pipeline. Non-trivial validation cases can be batched, and concurrency is capped at 20 active subagents so that the pool remains busy without overwhelming the execution environment. In addition, when semantic validation fails twice, the system preserves the original file instead of applying an unverified fallback and claiming a successful remediation.

4.4 Parallelism, Reliability, and Cost

The results highlight the central tradeoff of the system. Parallel processing reduces wall-clock time by allowing multiple documents to be remediated concurrently, while the three-agent validation workflow improves confidence in the proposed fixes. However, both mechanisms increase the number of model invocations and therefore the overall cost. On the Enterprise Networking Management course, the semantic stage operated on 83 documents and finished in approximately 28 minutes of wall-clock time at 20 concurrent agents. From the observed wall time and worker count, the average per-document agent latency was roughly 6.7 minutes, which yields an estimated sequential runtime of about 9 hours on a single worker. The corresponding speedup is close to the worker count, which indicates that the workload is highly parallelizable at the document level and that the pool spent most of its time saturated. We report this as an estimate rather than as a head-to-head benchmark because the same workload was not rerun sequentially.

This per-stage speedup does not translate directly to the same end-to-end course speedup. The total runtime is divided into three main chunks: downloading low-scoring documents from Blackboard, processing the documents, and uploading corrected files back through the web interface. Because the download and upload phases interact dynamically with the browser and remain largely sequential, each consumes about one third of the total runtime, with upload sometimes taking even longer than download. As a result, the overall course-completion speedup from parallel processing is lower than the semantic-stage speedup and is better described as an end-to-end improvement on the order of 20–25% for the current browser-driven workflow.

The validator behavior is also measurable. Across the three semantic-stage runs for which complete manifests are available, 113 documents entered the two-generator plus one-validator pipeline. On the representative course, the two generators agreed exactly on 21 of 83 documents, or about 25%, and the validator was bypassed entirely for those cases through a byte-equality shortcut. Of the remaining 62 documents for which the validator was invoked, it accepted the result on the first pass in 61 cases and reported disagreement in 1 case. That single disagreement was retried once with two fresh generators; the retry also diverged, and the system left the original file untouched in accordance with its no-partial-fix policy. No documents required manual intervention after the retry. Aggregated over the three courses with complete semantic manifests, the validator accepted 111 documents directly or via the shortcut, retried 1 document, skipped 1 document after the retry, and skipped 1 additional document for an unrelated payload-size error before validation could run.

4.5 Deployment and Usability Considerations

The current prototype is intended for instructor use on a local workstation rather than as a fully autonomous cloud service. In practice, an operator authenticates into Blackboard, selects a course, launches the four-phase workflow, and supervises transitions between phases. The system then

Table 2: Major remediation actions in the representative course and their estimated manual burden.

Remediation action	Typical file types	Representative workload	Why manual remediation is difficult	Estimated manual time per item	Estimated total manual burden
Alternative text generation for embedded images	PDF, PPTX, DOCX	513 images	Requires inspecting each figure, screenshot, or diagram and writing context-aware alternative text rather than generic labels.	1–2 min/image	8.6–17.1 hours
Heading structure correction	PDF, PPTX, DOCX	Within 82 semantically remediated documents	Requires reconstructing the logical document hierarchy and checking reading order for assistive technologies.	3–8 min/affected document	Several hours across affected files
Document title assignment	PDF, PPTX, DOCX	Within 82 semantically remediated documents	Often hidden in file metadata rather than visible content, so each document must be opened and checked individually.	1–3 min/affected document	1–4 hours across affected files
Language declaration	PDF, PPTX, DOCX	Within 82 semantically remediated documents	A small but easy-to-miss metadata repair that still requires per-file verification.	0.5–2 min/affected document	Under 3 hours across affected files
PDF tagging and semantic structure repair	PDF	PDF subset of the 82 remediated documents	Requires checking tag structure, lists, headings, and reading order rather than only visible formatting.	5–15 min/affected PDF	Several hours depending on PDF count
Final QA before re-upload	PDF, PPTX, DOCX	82 remediated documents	Requires checking that fixes are meaningful, non-empty, and safe to upload back into the course.	0.5–2 min/document	0.7–2.7 hours

navigates the course, collects low-scoring files from Blackboard Ally, downloads the selected artifacts, applies remediation locally, and uploads accepted corrections back into the course. This operating model was chosen to preserve human oversight during deployment and to avoid directly exposing course content to a separate remote document-processing service.

In order to implement the prototype, an instructor needs a local workstation with a supported web browser for Blackboard interaction, a Python environment with the document-processing dependencies needed for PDF, PPTX, and DOCX manipulation, and API credentials for the agentic remediation stages. The current implementation also assumes access to Blackboard Ally reports and permission to re-upload corrected files into the course site. In its present form, the system is therefore best understood as a prototype that can be used by technically capable instructors or instructional support staff.

The likely cost of deployment depends primarily on the number of files selected for remediation, the number of embedded images that require alternative text, and the level of validation used in the pipeline. In the representative Enterprise Networking Management course, the stricter validation configuration incurred an estimated operational cost of $220. This cost is driven mainly by the repeated model invocations used for semantic repair, image-description generation, and validation. As a result, higher-confidence configurations are more expensive, even though they reduce the likelihood of uploading weak or incomplete fixes.

Human-in-the-loop verification is therefore an important safeguard in the proposed workflow. The system is designed to apply changes automatically only when the remediation is sufficiently supported by its internal validation process. For example, in the image-remediation stage, two independent agents generate alternative-text candidates and a third agent checks whether the descriptions are similar enough to justify acceptance. When agreement is reached, the system treats the resulting description as high confidence and applies it automatically. When the descriptions diverge, the case is better understood as ambiguous rather than failed, and it is deferred to human review. The same principle applies more broadly to semantic remediation tasks: some accessibility fixes, such as inserting required metadata or applying validated structural corrections, are often robust enough for automation, whereas content-dependent interpretations require instructor verification when agents disagree.

4.6 Limitations

The current system does not remediate all accessibility issues. According to preliminary evaluations, out-of-scope or unresolved categories include color contrast problems, broken HTML links, optical character recognition (OCR), and video subtitle generation. These limitations help explain why

Blackboard Ally scores may improve substantially without necessarily reaching full compliance.

The workflow is also primarily file-centered rather than course-centered. It does not currently maintain a shared global context across multiple files in order to coordinate remediation decisions jointly. Although many accessibility fixes can be applied locally and independently, some decisions could benefit from broader course-level context when related materials are distributed across several files. Extending the workflow to model cross-file relationships and consistency is therefore a direction for future work.

The evaluation is also limited in scope. Three test courses are sufficient to demonstrate feasibility, but not enough to fully characterize generalization across departments, document types, and instructional styles. In addition, Blackboard Ally scores are useful for large-scale assessment, but they do not guarantee that the semantic content of every accessibility field is meaningful.

5 Conclusion and Future Work

This paper presented a multi-agent system for LMS accessibility remediation that combines phased workflow design, parallel document processing, and validation-oriented agent collaboration. Across three networking courses, the system improved Blackboard Ally scores substantially and reduced a manual workload that can otherwise take weeks or months.

The paper also highlights the tradeoff among throughput, reliability, and cost. Parallelism reduces runtime, validation improves trustworthiness, and both increase API expense. Even with these tradeoffs, the results show that multi-agent AI can serve as a practical remediation layer in LMSs.

Future work includes expanding the evaluation to a larger set of courses, measuring throughput more rigorously, comparing alternative agent configurations, and extending remediation coverage to classes that remain out of scope.

References

[1] Wajdi Aljedaani, Mohammed Alkahtani, Stephanie Ludi, Mohamed Wiem Mkaouer, Marcelo M. Eler, Marouane Kessentini, and Ali Ouni. 2023. The State of Accessibility in Blackboard: Survey and User Reviews Case Study. In *Proceedings of the 20th International Web for All Conference (W4A '23)*. Association for Computing Machinery, New York, NY, USA, 12. doi:10.1145/3587281.3587291 [Online]. Available: https://doi.org/10.1145/3587281.3587291.

[2] Anthology. 2026. Blackboard Ally for LMS. [Online]. Available: https://ally.ac/. Accessed: 2026-05-14.

[3] Jannatun Ara, Csilla Sik-Lányi, András Kelemen, and Gábor Kelemen. 2025. An inclusive framework for automated web content accessibility evaluation. *Universal Access in the Information Society* 24 (2025), 1581–1607. doi:10.1007/s10209-024-01164-5

[4] Paola Ingavélez-Guerra, Salvador Otón-Tortosa, José Hilera-González, and Mary Sánchez-Gordón. 2023. The use of accessibility metadata in e-learning environments: a systematic literature review. *Universal Access in the Information Society* 22 (2023), 445–461. doi:10.1007/s10209-021-00851-x

[5] Iat Long Iong, Xiao Liu, Yuxuan Chen, Hanyu Lai, Shuntian Yao, Pengbo Shen, Hao Yu, Yuxiao Dong, and Jie Tang. 2024. OpenWebAgent: An Open Toolkit to Enable Web Agents on Large Language Models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*. Association for Computational Linguistics, Bangkok, Thailand, 72–81. doi:10.18653/v1/2024.acl-demos.8 [Online]. Available: https://aclanthology.org/2024.acl-demos.8/.

[6] Yilun Kong, Jingqing Ruan, YiHong Chen, Bin Zhang, Tianpeng Bao, Shiwei Shi, Guoqing Du, Xiaoru Hu, Hangyu Mao, Ziyue Li, Xingyu Zeng, Rui Zhao, and Xueqian Wang. 2024. TPTU-v2: Boosting Task Planning and Tool Usage of Large Language Model-based Agents in Real-world Industry Systems. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*. Association for Computational Linguistics, Miami, Florida, US, 371–385. doi:10.18653/v1/2024.emnlp-industry.27 [Online]. Available: https://aclanthology.org/2024.emnlp-industry.27/.

[7] Merja Laamanen, Tarja Ladonlahti, Hannu Puupponen, and Tommi Kärkkäinen. 2024. Does the law matter? An empirical study on the accessibility of Finnish higher education institutions' web pages. *Universal Access in the Information Society* 23 (2024), 475–491. doi:10.1007/s10209-022-00931-6

[8] Chenliang Li, He Chen, Ming Yan, Weizhou Shen, Haiyang Xu, Zhikai Wu, Zhicheng Zhang, Wenmeng Zhou, Yingda Chen, Chen Cheng, Hongzhu Shi, Ji Zhang, Fei Huang, and Jingren Zhou. 2023. ModelScope-Agent: Building Your Customizable Agent System with Open-source Large Language Models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Association for Computational Linguistics, Singapore, 566–578. doi:10.18653/v1/2023.emnlp-demo.51 [Online]. Available: https://aclanthology.org/2023.emnlp-demo.51/.

[9] Amy Lomellini, Patrick R. Lowenthal, Chareen Snelson, and Jesús H. Trespalacios. 2025. Accessible and inclusive online learning in higher education: a review of the literature. *Journal of Computing in Higher Education* 37 (2025), 1306–1329. doi:10.1007/s12528-024-09424-2

[10] Ali Mazloum, Jose Gomez, Elie Kfoury, and Jorge Crichigno. 2026. lms-accessibility-agent. GitHub repository. [Online]. Available: https://github.com/gomezgaona/lms-accessibility-agent. Accessed: 2026-07-01.

[11] Norun Christine Sanderson, Siri Kessel, and Weiqin Chen. 2022. What do faculty members know about universal design and digital accessibility? A qualitative study in computer science and engineering disciplines. *Universal Access in the Information Society* 21 (2022), 351–365. doi:10.1007/s10209-022-00875-x

[12] U.S. Department of Justice. 2024. Nondiscrimination on the Basis of Disability; Accessibility of Web Information and Services of State and Local Government Entities. Final rule. [Online]. Available: https://federalregister.gov/d/2024-07758. Accessed: 2026-05-14.

[13] World Wide Web Consortium. 2023. Web Content Accessibility Guidelines (WCAG) 2.2. [Online]. Available: https://www.w3.org/TR/WCAG22/. Accessed: 2026-05-14.

[14] Junwei Yu, Yepeng Ding, and Hiroyuki Sato. 2025. DynTaskMAS: A Dynamic Task Graph-driven Framework for Asynchronous and Parallel LLM-based Multi-Agent Systems. *Proceedings of the International Conference on Automated Planning and Scheduling* 35, 1 (2025), 288–296. doi:10.1609/icaps.v35i1.36130

[15] Zhilun Zhou, Zihan Liu, Jiahe Liu, Qingyu Shao, Yihan Wang, Kun Shao, Depeng Jin, and Fengli Xu. 2026. ResMAS: Resilience Optimization in LLM-based Multi-agent Systems. *Proceedings of the AAAI Conference on Artificial Intelligence* 40, 41 (2026), 35176–35184. doi:10.1609/aaai.v40i41.40824