Minimizing Data Exposure in Higher Education LLM Applications: Evaluating the Model Context Protocol (MCP) for Preserving Privacy in Academic Advising

Bill Dong[1[0009-0008-1982-7717]] and Jessica Liebowitz[1,2[0000-0002-9601-5615]]

[1] Brandeis University, Waltham MA 02453, USA
[2] Northeastern University, Boston MA 02115, USA
yingxuandong@brandeis.edu, jkl@brandeis.edu

Abstract. Large language models (LLMs) are increasingly used in academic advising and student dashboards. However, traditional full-record integrations send complete student profiles to third-party model providers. This paper checks if the Model Context Protocol (MCP) can lower exposure by making the model retrieve only necessary fields through logged tools. Using fully synthetic U.S. undergraduate records, we compare a traditional full-record tool design that includes seven profile fields in every request with an MCP per-field tool design across two advising tasks. We measure privacy risk with a tiered leakage score and evaluate response quality using an LLM-as-a-judge approach. Across three runs, the traditional approach gives the highest average leakage score of 30, while MCP reduces leakage to about 10 mainly by avoiding access to identifiers. However, this comes with a drop in quality that depends on the task (overall from about 5.0 to 3.1). These findings show that MCP can significantly reduce the exposure of confidential student information while still providing high-quality advising for tasks that require less data. Lastly, the tiered leakage metric and MCP tool-call logs together make privacy exposure transparent and measurable. The results motivate future work to understand and control field-level retrieval in MCP-based advising, explaining when and why models over- or under-retrieve specific student fields to support safer and more useful deployments.

Keywords: Model Context Protocol (MCP), Data Minimization, Data Leakage Measurement, Educational AI Governance

1 Introduction

Universities are increasingly including large language models (LLMs) in academic advising systems, learning dashboards, and other tools. While personalizing guidance can improve outcomes, it often relies on sensitive education record data. In a typical advising situation, important context may include identifiers like student ID and name, academic details like major and GPA. When these details are shared with third-party model providers, institutions face increased privacy, governance, and compliance risks.

Regulatory guidelines stress the need to limit what is shared. In the United States, the Family Educational Rights and Privacy Act (FERPA) restricts the disclosure of

personally identifiable information from education records without consent. It also requires vendors to use student data only for approved purposes [1]. The European Union's General Data Protection Regulation (GDPR) similarly focuses on limiting purpose and minimizing data. It requires that personal data be adequate, relevant, and restricted to what is necessary for a specific task [2]. These rules raise concerns about common LLM integration methods that send more student data than is needed for a question.

Discussions about privacy risk in these integrations often remain at a general level. They rarely measure which fields were exposed and how sensitive those fields are. This paper innovatively offers a clear evaluation metric that takes sensitivity into account. It features a tiered leakage score that quantifies exposure by weighting record fields based on their sensitivity (for example, direct identifiers, academic progress, and general attributes). This metric allows for a clear and measurable view of privacy exposure.

Most LLM deployments use a full-record method. The application gathers a complete student profile and includes it in each API request, regardless of the prompt. This method is easy to use but tends to overshare and provides little insight into which fields were actually required. We explore an alternative design based on the Model Context Protocol (MCP), where the model starts with just the student's question and retrieves specific fields through clear, per-field tools operating within the institution's environment, with each access logged for auditing. Despite growing interest in tool-based LLM architectures, we find no published evaluation to our knowledge of MCP applications in student advising use case.

This paper addresses a specific research question: when an advising LLM has separate per-field MCP tools for accessing student data, does it reveal less sensitive information to the model provider compared to the traditional full-record method? Additionally, how does the quality of responses compare between an LLM using MCP and one that does not? We generated fully synthetic undergraduate records that closely resemble the characteristics of real undergraduate data to simulate an academic advisor LLM that answers two common questions, "Am I on track to graduate in four years?" and "Should I consider changing my major?", under both approaches. We measure exposure using the tiered leakage score that weighs each field based on its significance. We also evaluate response quality using an LLM-as-a-judge approach.

2 Related Work

2.1 LLM-Driven Educational Chatbots and Agents

Recent work views LLM-based educational systems as tools that combine memory, tool use, and planning. They support tasks like tutoring, feedback, and student assistance, while also facing ongoing challenges in deployment, such as privacy, hallucination, and integration with current platforms [3]. For example, MoodleBot includes an LLM chatbot within an LMS and employs retrieval-augmented generation (RAG) alongside user feedback, reporting strong alignment with course content in the classroom [4]. These systems mainly focus on improving helpfulness, grounding, and user experience, but there is less emphasis on considering data exposure to third-party model providers as an important design factor.

Previous work on secure GenAI adoption highlights risks from using public cloud LLMs, including unclear retention and training policies, governance issues, and compliance worries. This drives the need for stronger institutional controls or private, on-premise deployments [5]. These concerns are particularly important for student advising systems, where a full-record approach greatly increases the risk of oversharing sensitive student data.

2.2 Tool-Based and Retrieval-Augmented Architectures

Technically, our work is closest to retrieval and tool augmented LLM systems. RAG combines parametric generation with non-parametric retrieval to improve factuality and allow for updatable, inspectable evidence [7]. Toolformer demonstrates that language models can learn through self-supervision when to call external APIs and how to incorporate results [6]. This improves performance while maintaining generality. Building on these ideas, our contribution reframes tools not just as a capability mechanism but also as a privacy boundary. We propose an MCP per-field API design that minimizes the student data exposed to the model provider while allowing for auditable access and evaluation of privacy and utility.

2.3 Novelty and Contribution

To our knowledge, MCP has not previously been evaluated in an academic advising setting in the research literature. Prior work emphasizes chatbots, RAG, and application layer safety, but it does not examine MCP as a privacy protection technique for advising systems. This paper's innovation is to bring MCP into the advising context and empirically compare it to the full-record design. By running this first analysis, we can therefore specify which research questions must be answered to responsibly leverage MCP for LLM-integrated student affairs advising.

3 Methods

3.1 Synthetic Dataset Construction

To avoid using real student records while still showing realistic characteristics, we generated completely synthetic undergraduate profiles that represent typical U.S. four-year students. Each experimental run produced a new group of 50 students, and we repeated this process for three separate runs, resulting in 150 unique synthetic students in total. We conducted analyses for each run individually to maintain their independence.

Each student record was saved as a JSON object with seven fields: (1) student_id, a pseudo-identifier that starts with "S" followed by six digits; (2) full_name, which is a synthetic name; (3) email, which is generated from the name using an institutional domain; (4) major; (5) year, indicating class level; (6) gpa; and (7) credits_earned. We created names using a Node.js package called @faker-js/faker, and we generated emails from normalized names, handling collisions by adding numeric suffixes.

We generated synthetic student data by replicating the distributional characteristics of real undergraduate populations across four dimensions: major, class year, GPA, and credits earned. We sampled majors to match NCES field-of-study distributions while excluding the "Other and not classified" category [8]. The class year was sampled based on NCES undergraduate class-level proportions [11]. GPA values were taken from a left-skewed distribution centered between 3.0 and 4.0 to reflect realistic grade distributions [9], and credits earned were sampled within typical ranges according to class standing (Freshman: 0-29; Sophomore: 30-59; Junior: 60-89; Senior: 90+).

3.2 Experimental Setup

We simulated an AI academic advisor accessible through a user interface. This assumes the backend can recognize the logged-in student and retrieve the full synthetic record. We used the advisor model Claude-sonnet-4-20250514. For every student, we instructed the model to respond to two fixed prompts, "Am I on track to graduate in four years?" and "Should I consider changing my major?". These questions were selected because they reflect common advising issues. Prompt 1 relies on detailed academic progress, while Prompt 2 can often be answered with more general information. We evaluated two architectural conditions:

- Condition 1: Traditional full-record integration. For every student and prompt, the application sent the full student record to the LLM provider as part of the request context. This included student_id, full_name, email, major, year, gpa, and credits_earned, regardless of which fields were necessary for the answer.
- Condition 2: MCP per-field API integration. In this MCP setup, the LLM did not receive the complete record. Instead, it accessed individual fields by calling specific MCP tools available through an MCP server (one tool for each field): get_student_id, get_name, etc. The model decided which tools to use and the order they were called; the server only returned the requested field values and logged each tool use. Since the backend already knows which student is logged in, the system assumes identifier tools are not needed for the prompts, and ideally, the model should not call for them. For analysis, the effective exposure in MCP was defined as the fields actually retrieved through tool calls in that interaction.

In each run, we generated 50 students and collected responses for both prompts under both conditions. This resulted in 4 responses per student per run, totaling 200 responses per run, along with MCP tool-call logs indicating which fields were accessed.

3.3 Data Leakage Measurement

To measure privacy risk, we grouped student record fields into three sensitivity levels:

- Tier 1 (most sensitive - direct identifiers): full_name, email, student_id
- Tier 2 (academic performance and progress): gpa, credits_earned
- Tier 3 (least sensitive - general attributes): major, year

For each response, we computed a Leakage Score (LS) based on how many fields from each tier were exposed to the LLM provider during that interaction:

$$LS = 3\times(\#\ Tier\ 1\ fields) + 2\times(\#\ Tier\ 2\ fields) + 1\times(\#\ Tier\ 3\ fields) \quad (1)$$

Exposing all seven fields gives the highest LS of 15. In the traditional condition, LS is always 15 because the complete record is always included. Under MCP, LS changes based on the interaction and the tools used.

We reported leakage at two aggregate levels for each run and condition: the Total Leakage Score (TLS), which is the total of LS across all student-prompt interactions in that condition and run; and the Average Leakage Score (ALS), which is the average leakage per student across both prompts. This is calculated by adding each student's two prompt LS values and then averaging over the 50 students.

3.4 Response Quality Evaluation (LLM-as-a-Judge)

To determine if MCP's reduced exposure affected the quality of advising, we used an LLM-as-a-judge approach [10]. Claude generated responses for both conditions and also served as the judge. This avoided self-enhancement bias. For every student and prompt, the judge reviewed the student profile as the ground truth, the question, and the two responses (Traditional and MCP). Then, the judge assigned each response a score from 1 (very poor) to 5 (excellent). Scores needed to be in a strict JSON format to enable automated analysis. We defined quality in two ways:

1. Accuracy: This means correctly using the student's major and year, and when relevant, GPA and credits. Any fabricated or contradictory details received low scores.
2. Relevance: This refers to how well the response answers the specific prompt, avoiding generic or off-topic replies.

To avoid position bias, we evaluated each pair of responses twice, changing the order (Traditional first and MCP first). We then averaged the two scores assigned to each response to create a single score. Finally, we calculated the average quality by prompt and condition within each run (Prompt 1 vs. Prompt 2, and Overall).

4 Results

4.1 Data Leakage

Table 1. Average Leakage Scores and Total Leakage Scores by Run and Integration

	Traditional		MCP	
Runs	**ALS**	**TLS**	**ALS**	**TLS**
Run 1	30	1500	10	500
Run 2	30	1500	9.98	499

Run 3	30	1500	10	500

In all three runs, the traditional approach always showed the complete student profile for each interaction. This fixed the leakage at LS = 15 per prompt and ALS = 30 per student (for two prompts). In contrast, the MCP approach varied the leakage based on which tools the model selected. This resulted in much lower exposure in every run (Run 1: ALS 10, TLS 500; Run 2: ALS 9.98, TLS 499; Run 3: ALS 10, TLS 500).

On average, the traditional ALS was 30, while the MCP ALS was ~9.99. This shows about a 66-67% reduction in per-student leakage with MCP. Total leakage showed a similar trend: over the 3 runs, the traditional totaled 4500 compared to 1499 for MCP.

4.2 Response Quality

Table 2. Mean Response Quality Scores by Run, Prompt, and Integration

	Traditional			**MCP**		
Runs	**P1**	**P2**	**Overall**	**P1**	**P2**	**Overall**
Run 1	5	5	5	2.302	4.170	3.236
Run 2	4.917	5	4.959	2.021	4.133	3.077
Run 3	4.990	5	4.995	2.083	4.260	3.172

Quality scores show a task-dependent trade-off. For Prompt 1 ("Am I on track to graduate in four years?"), the traditional responses were generally close to the maximum, with run means between 4.917 and 5. In contrast, MCP responses were much lower, with run means ranging from 2.021 to 2.302. Overall, the paper reports a mean quality for Prompt 1 of about 4.97 for traditional responses compared to around 2.14 for MCP.

For Prompt 2 ("Should I consider changing my major?"), the difference was smaller. Traditional responses scored an average of 5, while MCP responses averaged about 4.19, which still indicated relatively high quality.

When combining both prompts and all runs, traditional responses averaged ~4.98, while MCP averaged ~3.16. Most of the drop came from the data-intensive first prompt.

5 Discussion

5.1 Data Leakage vs. Response Quality

This experiment highlights a clear trade-off between the traditional and MCP approaches in terms of privacy and utility. In the traditional model, each interaction reveals the entire student profile, resulting in the highest ALS. In contrast, the MCP greatly limits exposure since the model only retrieves selected fields. Our logs show that the model never accessed identifier fields like student ID, name, or email, which accounts for most of the reduction in data exposure. The remaining difference came from occasionally skipping other academic or contextual fields. However, this decrease

in data exposure has a downside: the overall quality of responses declines under MCP. This indicates that simply reducing access does not ensure same advising performance.

5.2 Task Dependence

The effect of MCP on quality depends a lot on the advising task. For the first question, "Am I on track to graduate in four years?" the traditional method performs much better. This is because accurate guidance needs precise information about academic progress, especially credits earned and GPA, along with more structured reasoning about requirements. Under MCP, responses are not as strong since the model sometimes fails to retrieve all important fields before answering. This leaves it with an incomplete understanding, which reduces accuracy and relevance. In contrast, for Prompt 2, "Should I consider changing my major?" MCP is close to the traditional method. Basic reasoning and general academic advice are usually enough in this situation. The model can provide good suggestions with just the student's current major and class year, along with some broad principles. This makes Prompt 2 need less context. These results indicate that the MCP works best for tasks that don't depend heavily on detailed academic data.

5.3 Transparency and Governance Benefits

Beyond reducing exposure, MCP offers a governance benefit through visibility. Unlike the traditional approach, where the full record is included into the prompt with little insight into its use, MCP clearly logs each data access. This allows institutions to track what was accessed, when it was accessed, and how often tools are used. This clear record helps with accountability and better supports data minimization, which is highly valued in higher education, given the sensitive nature of student data.

5.4 Limitations and Future Work

The study has several limitations. The dataset is entirely synthetic and does not reflect the variability and complexity of real student records. For instance, real-world student records contain far more than seven fields, including completed courses, ongoing courses, and sensitive information such as student gender and ethnicity. In addition, the study only evaluates two advising prompts, so the findings may not generalize to other advising workflows such as course planning or prerequisite checking. Future work will test MCP across a broader range of advising tasks.

Results may also vary across models, as both the responses and evaluations came from a single LLM. Future work will use separate judge models to evaluate response quality, including cross-model comparisons with systems such as ChatGPT or Gemini. This will enable examination of whether the observed quality differences vary depending on the choice of evaluator. Future work will also broaden examination of the quality criteria, beyond accuracy and relevance to include important advising qualities like tone and empathy.

Furthermore, the MCP tools were simple and only allowed per-field retrieval. A key question for our future research is why the quality drop for Prompt 1 was so severe

under MCP. Future research will begin by investigating whether the failure to call necessary tools is the primary cause of the decline in the quality of model responses. Alternative tool designs could better balance privacy and utility: for example, returning GPA ranges (3.0-3.5) or narrative summaries ("on track for graduation") rather than exact values. Finally, the model should be tested with real advisors in actual institutional settings to determine if it genuinely reduces their workload while protecting student privacy.

6 Conclusion

This paper demonstrates that MCP-based field-level access controls can substantially reduce student data exposure to third-party LLM providers without sacrificing advising quality across all tasks. Our comparison shows that selective field access reduced average leakage scores from 30 to about 10, primarily by avoiding direct identifiers. This privacy gain came with minimal quality trade-offs for lower-data tasks like major changes, though performance declined predictably for data-intensive graduation audits.

Beyond these empirical findings, this work establishes a methodological foundation for evaluating privacy-utility trade-offs in educational AI systems. The tiered leakage metric and logged tool calls make data exposure transparent and auditable, which is critical for institutional governance. To our knowledge, this is the first empirical evaluation of MCP as a privacy protection mechanism in academic advising, where prior work has focused primarily on chatbot interfaces and application-layer safety.

Future research should investigate which student fields are genuinely necessary for different advising tasks, why models sometimes over- or under-retrieve information, and how to optimize MCP implementations for both safety and performance. As institutions integrate LLMs into student-facing systems, understanding these field-level access patterns will be essential for responsible deployment.

Acknowledgments. We are especially grateful to Professor Timothy Hickey and Professor Nianwen Xue, both of the Department of Computer Science at Brandeis University, for their engagement with this project and insightful feedback throughout this work.

Disclosure of Interests. The authors have no competing interests to declare.

References

1. U.S. Department of Education: FERPA | Protecting Student Privacy. (Accessed: 2026-02-01)
2. European Union: Regulation (EU) 2016/679 (General Data Protection Regulation), Art. 5. EUR-Lex. (Accessed: 2026-02-01)
3. Chu, Z., et al.: LLM Agents for Education: Advances and Applications. arXiv preprint arXiv:2503.11733 (2025)

4. Neumann, A.T., et al.: An LLM-Driven Chatbot in Higher Education for Databases and Information Systems. IEEE Transactions on Education 68(1), 103–116 (2025). doi:10.1109/TE.2024.3467912
5. Jayaram, Y.: Private LLMs for higher education: secure GenAI for academic & administrative content. American International Journal of Computer Science and Technology 6(4), 28–38 (2024). doi:10.63282/3117-5481/AIJCST-V6I4P103
6. Schick, T., et al.: *Toolformer: Language Models Can Teach Themselves to Use Tools.* arXiv preprint arXiv:2302.04761 (2023)
7. Lewis, P., et al.: Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. arXiv preprint arXiv:2005.11401 (2021)
8. National Center for Education Statistics: Table 322.10. Bachelor's degrees conferred by postsecondary institutions, by field of study: Selected academic years, 1970–71 through 2021–22. Digest of Education Statistics 2023. U.S. Department of Education. (Table prepared September 2023.) Available at https://nces.ed.gov/programs/digest/d23/tables/dt23_322.10.asp, last accessed 2026/01/31
9. Weiss, G.M., et al.: An Analysis of Grading Patterns in Undergraduate University Courses. In: 2023 IEEE 47th Annual Computers, Software, and Applications Conference (COMPSAC), Torino, Italy, pp. 310–315 (2023). doi:10.1109/COMPSAC57700.2023.00048
10. Gu, J., et al.: A Survey on LLM-as-a-Judge. arXiv preprint arXiv:2411.15594v6 [cs.CL] (2025)
11. Campbell, T., Wescott, J.: Profile of Undergraduate Students: Attendance, Distance and Remedial Education, Degree Program and Field of Study, Demographics, Financial Aid, Financial Literacy, Employment, and Military Status: 2015–16. Web Tables, U.S. Department of Education / National Center for Education Statistics, NCES 2019-467 (Jan 2019)