

SmartCourse: A Contextual AI-Powered Course Advising System for Undergraduates

1st Yixuan Mi*

*Department of Computer Science,
Wenzhou-Kean University,
Wenzhou, China*

*Department of Computer Science,
Kean University,
Union, NJ, United States
miy@kean.edu*

2nd Yiduo Yu

*Department of Computer Science,
Wenzhou-Kean University,
Wenzhou, China*

*Department of Computer Science,
Kean University,
Union, NJ, United States
yuyid@kean.edu*

3rd Yiyi Zhao

*Department of Computer Science,
Wenzhou-Kean University,
Wenzhou, China*

*Department of Computer Science,
Kean University,
Union, NJ, United States
zhaoyiy@kean.edu*

Abstract—We present SmartCourse, an integrated course management and AI-driven advising system for undergraduate students (specifically tailored to the Computer Science (CPS) major). SmartCourse addresses the limitations of traditional advising tools by integrating transcript and plan information for student-specific context. The system combines a command-line interface (CLI) and a Gradio web GUI for instructors and students, manages user accounts, course enrollment, grading, and four-year degree plans, and integrates a locally hosted large language model (via Ollama) for personalized course recommendations. It leverages transcript and major plan to offer contextual advice (e.g., prioritizing requirements or retakes). We evaluated the system on 25 representative advising queries and introduced custom metrics—PlanScore, PersonalScore, Lift, and Recall—to assess recommendation quality across different context conditions. Experiments show that using full context yields substantially more relevant recommendations than context-omitted modes, confirming the necessity of transcript and plan information for personalized academic advising. SmartCourse thus demonstrates how transcript-aware AI can enhance academic planning. The source code is publicly available at this GitHub repository.

Index Terms—Academic advising, Course recommendation, Large Language Models (LLMs), Degree planning

I. INTRODUCTION

Traditional academic advising tools often provide generic guidance, lacking integration with student-specific data such as transcripts and degree plans [1]. Existing advising solutions tend to be generic and fail to incorporate detailed student context (such as completed courses or degree requirements) [2], [3]. In particular, there is a lack of advising systems that directly integrate a student’s own transcript and major plan into the recommendation process. More recently, advances in AI and LLMs suggest new possibilities for personalized advising [4]. Chatbot-based advisors can offer tailored course selection advice and career guidance [5]. Our system supports three types of users—students, instructors, and administrators—each interacting through role-specific interfaces. Motivated by these trends, SmartCourse addresses these limitations by embedding personalized advising directly into the academic workflow.

We investigate how transcript and degree plan impact course recommendation relevance. To this end, we design controlled experiments that systematically omit these inputs and analyze the resulting drop in recommendation quality.

This paper makes the following contributions:

- **Contextual Advising Architecture:** We develop SmartCourse, a modular platform that tightly couples academic operations with AI-driven advising. SmartCourse combines core academic services with AI advising in a unified interface.
- **AI-Powered Recommendation Engine:** SmartCourse integrates a locally hosted LLM (via Ollama) to generate course suggestions. The LLM generates suggestions based on contextual prompts. Automated email notifications can be sent when new recommendations or updates are available.
- **Experimental Evaluation with Context Ablation:** We formulate a set of 25 realistic advising queries (e.g. elective choices for AI specialization, courses for a cyber security graduate track, GPA-improvement strategies) and evaluate SmartCourse under four context conditions (full context, no transcript, no plan, question-only). We introduce relevance metrics – PlanScore, PersonalScore, Lift, and Recall – to quantify how well the recommendations align with the student’s outstanding degree requirements, as well as latency in seconds.
- **Results and Insights:** Incorporating transcript and plan context leads to consistently higher recommendation quality, while omitting them harms performance. These findings highlight the importance of contextual information in personalized advising, showing clear improvement over generic recommendations (see Table I for a summary of evaluation scores).

The remainder of the paper reviews related work (Section II), describes the system design (Sections III–IV), presents the experimental setup and results (Sections V–VI), and discusses key findings, limitations, and future directions (Section VII).

* Corresponding Author

II. RELATED WORK

Course recommendation in education has been studied through collaborative filtering, matrix factorization, and content-based methods [6]–[8]. These techniques often rely on preference data or grade history but typically ignore structured curricular requirements [9]. Meanwhile, academic dashboards and degree audit tools help students monitor progress and register for courses, but rarely offer personalized or adaptive suggestions. SmartCourse addresses this gap by aligning recommendations with formal degree plans and transcript records.

More recently, LLMs have been explored for educational use cases [10], such as chatbot-based tutoring, FAQ answering, or summarizing curriculum content [11]. While LLMs can offer natural-language interactions, they often lack access to structured academic records, leading to generic or hallucinated advice [12].

SmartCourse builds upon these strands by embedding a locally hosted LLM into an end-to-end course management system [13], [14]. Unlike prior work, it fuses transcript, degree plan, and user queries into contextual prompts, enabling personalized and curriculum-aligned advising within a unified platform.

III. SYSTEM OVERVIEW

A. Architecture Overview

SmartCourse is organized into several interacting components. It maintains student and instructor accounts (with roles and majors), supports course management (registration, prerequisites, and grades), and enforces a four-year degree plan for the specific major. A central AI Recommendation Engine uses a local LLM to answer student questions based on their academic record. The system provides both a text-based CLI and a user-friendly Gradio web interface for interaction. Fig. 1 illustrates the overall SmartCourse architecture (data inputs, modules, and UI layers).

B. Core Components

SmartCourse comprises the following building blocks:

- **User Accounts:** Credentials, user roles (student, instructor or administrator), and declared majors are stored in a secure account store with hashed passwords.
- **Course Enrollment & Grading:** A course-catalog repository lists all available courses. Students enroll through either interface; instructors record grades, which are written to an enrollment ledger that tracks course codes and grades per student.
- **Degree Plan Management:** For each major (e.g. CPS) a standard four-year plan repository defines required courses by year. The system compares this plan against the transcript to monitor progress and identify outstanding requirements.
- **AI Recommendation Engine:** When a student poses a question, SmartCourse constructs a prompt that fuses the question text, current transcript, and four-year plan context. The prompt—structured as transcript, plan, and

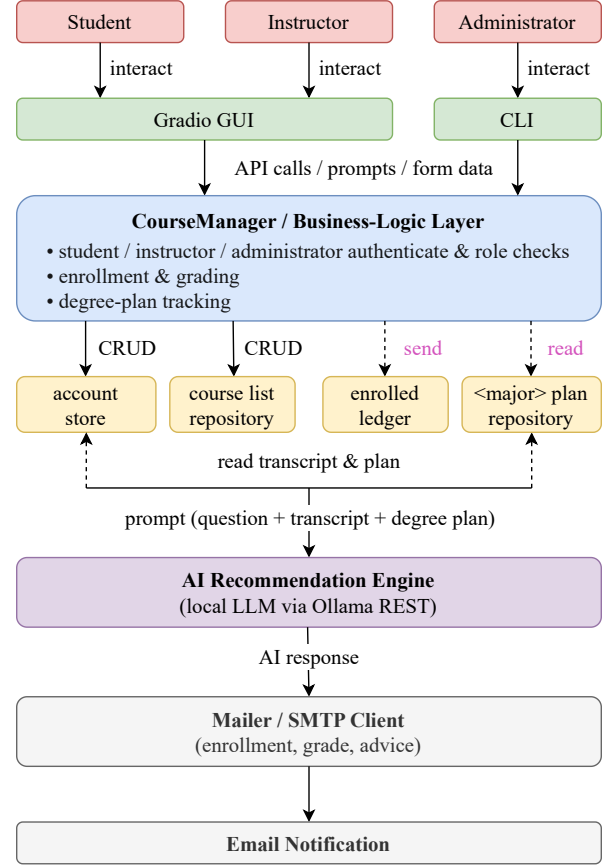


Fig. 1. SmartCourse system architecture. Users interact via CLI (administrator) or Gradio GUI (instructor & student), managed by the *CourseManager* layer. Data is stored in flat-file repositories, with an LLM providing recommendations and a mailer handling notifications.

question—is sent to a locally hosted LLM (specifically `llama3.1:8b` via the Ollama runtime). The returned answer is parsed and filtered into valid course recommendations (Fig. 2).

- **Interfaces:** We illustrate these interfaces below to provide a concrete view of the system’s user experience. Administrators operate through a dedicated CLI to maintain system-level configurations. Their responsibilities include managing the course catalog (adding / removing courses), auditing user accounts, viewing system logs, and switching the local LLM model when needed. Fig. 3 shows the administrator CLI menu, which emphasizes control and audit capabilities over advising interactions. Instructors (Fig. 4) and students (Fig. 5) interact via a Gradio-based web GUI that supports chat-style queries and form views. For example, a student can ask “Which electives should I take next semester to prepare for a machine-learning track?” and will receive tailored suggestions.
- **Notifications:** A background mailer module dispatches

Seeking Academic Advice from AI:

Your Problem:

Based on my academic record, what courses should I register for next semester?

Question AI

AI ADVICE (🕒 44.9s)

Based on your academic record and the four-year plan provided, here are the courses you should register for next semester:

1. **MATH 2416: Calculus II**
 - This course is required as it follows MATH 2415 (Calculus I) in your major requirements.
2. **CPS 3962: Object-Oriented Analysis and Design**
 - This course fulfills a major requirement and should be included in your registration.

These courses align with your degree plan, covering both mathematics and computer science components essential for your major.

Fig. 2. AI-generated advising response based on integrated transcript and degree plan context. *This response was rendered as a schematic interface for clarity and does not depict a live system screen.*

Welcome, admin@smartcourse.com!

Administrator Menu:

1. View All Accounts
2. Add Course to Course List
3. Remove Course from Course List
4. Check Logs
5. Change Localhost AI Model

Enter Your Choice (1 - 5):

Fig. 3. Administrator interface for managing accounts, courses, and AI model configurations via CLI. *Simulated terminal view created for illustrative purposes; not a direct screenshot.*

SmartCourse Management System

Welcome, instructor@smartcourse.com!

Instructor Menu:

View Student Courses

Assign Grade

Logout

Fig. 4. Instructor interface for assigning grades and viewing student course records. *Drawn representation of the interface layout; the actual GUI may differ visually.*

SmartCourse Management System

Welcome, user@smartcourse.com!

Student Menu:

Enroll in Course

View My Courses

Drop Course

Ask AI for Advice

Logout

Fig. 5. Student interface for registering and dropping courses, viewing progress, and requesting AI-based advising. *This interface depiction is illustrative and not captured from a running system.*

email notifications—e.g. new recommendations, enrollment confirmations, or grade postings—to the relevant users through a configurable SMTP gateway.

These components work together to deliver an end-to-end advising workflow. Fig. 1 illustrates the data flow from user query, through the LLM engine, to the user interface.

IV. IMPLEMENTATION

A. Technology Stack and Data Storage

SmartCourse is implemented entirely in Python. For ease of deployment and maintenance, it relies on lightweight *flat-file repositories* rather than a full database:

- a *secure account store* that keeps hashed user credentials, roles, and declared majors;
- a *course-catalog repository* that lists every available course with its code and title;
- a *degree-plan repository* that records the standard four-year curriculum for each major; and
- an *enrollment ledger* that tracks, for each student, the courses taken and the grades awarded.

All repositories are plain UTF-8 text files and are read or written through a thin data-access layer.

B. Functional Modules

- A data module parses files into in-memory structures. The CLI (for admins) and GUI (for students / instructors) authenticate via a secure store.
- On student queries, SmartCourse fuses the transcript, plan, and question into a structured prompt, which is sent via Python subprocess to a local Ollama-based LLM.
- A lightweight mailer sends notifications such as recommendations or grades.

The entire system is parameterized: SMTP credentials, model name, and repository paths are configurable. Logs are maintained for auditing, and the system supports configurable parameters such as SMTP credentials, model name, and repository paths to ensure deployment flexibility.

V. EXPERIMENTS

A. Experimental Setup

To evaluate the importance of context in course advising, we designed a set of 25 hypothetical student queries and compared SmartCourse’s recommendations under four input conditions. This setup constitutes an ablation study: by selectively removing transcript or degree-plan inputs, we aim to quantify their individual and combined contributions to recommendation quality. Two illustrative prompts are:

- “What elective courses should I choose next semester to strengthen my AI foundation, considering the AI courses I have already taken?”
- “Which electives would best prepare me for a Ph.D. track in Machine Learning?”

For every prompt we compared four context settings:

- **Full Context** — the LLM receives both the student’s transcript and four-year degree plan.
- **No Transcript** — only the degree plan is provided.
- **No Plan** — only the transcript is provided.
- **Question-Only** — the model sees the question text alone.

All experiments were conducted using the `llama3.1:8b` model hosted locally through the Ollama runtime environment, ensuring consistent responses across all context conditions.

B. Dataset

The study centers on one CPS student profile and its associated academic artifacts:

- a *degree-plan repository* listing **39** required courses;
- a *transcript record* showing **21** of those courses already completed, including several low grades (e.g., B– or below), leaving **18** outstanding;
- a *course-catalog repository* containing **75** courses; and
- a *query set* of **25** representative advising questions.

Table I summarizes these statistics. Because no external ground-truth answers exist, we treat the degree plan as the reference set and evaluate recommendation quality using the relevance metrics defined in Section V-C, supplemented by manual inspection.

C. Evaluation Metrics

Let \mathcal{R} be the set of courses recommended by the LLM, \mathcal{P} the set of outstanding degree-plan requirements (i.e. courses in the plan that the student has *not yet* taken), and \mathcal{L} the set of courses the student *has* taken but with a low grade (below B–). Using these sets we compute:

- **PlanScore** — fraction of recommendations that satisfy an unmet plan requirement:

$$\text{PlanScore} = \frac{|\mathcal{R} \cap \mathcal{P}|}{|\mathcal{R}|}.$$

- **PersonalScore** — fraction of recommendations that either meet an unmet plan requirement or suggest retaking a low-grade course:

$$\text{PersonalScore} = \frac{|\mathcal{R} \cap (\mathcal{P} \cup \mathcal{L})|}{|\mathcal{R}|}.$$

- **Lift** — improvement from personalising to low grades:

$$\text{Lift} = \text{PersonalScore} - \text{PlanScore}.$$

- **Recall** — coverage of the student’s remaining plan courses:

$$\text{Recall} = \frac{|\mathcal{R} \cap \mathcal{P}|}{|\mathcal{P}|}.$$

- **Latency** — wall-clock time (in seconds) for the LLM to produce a complete answer.

PlanScore gauges alignment with curriculum requirements; PersonalScore adds sensitivity to prior performance; Lift isolates the benefit of that personalization; and Recall measures how comprehensively the remaining plan is covered. Latency captures practical responsiveness. Each of the 25 questions was evaluated in all four context modes, and metric means with 95% confidence intervals were obtained via 10,000 bootstrap iterations. Our metric design follows calls in recommender-systems research to evaluate beyond accuracy alone, incorporating coverage and user-need alignment [15], [16].

VI. RESULTS

SmartCourse successfully generated course recommendations for all queries under the different modes.

Under the full context setting, the system recommended an average of ~ 6.6 courses per query. The mean **PlanScore** was about **0.53**, and the mean **PersonalScore** was **0.78**. Consequently, the average **Lift** (Personal – Plan) was about **0.25**. The average **Recall** was **0.15**, and the mean latency was about **48** seconds.

By contrast, omitting context degraded performance significantly. In **No Plan** mode (transcript only), the mean **PlanScore** dropped to **0.03** and **Recall** to **0.01**. In other words, very few needed courses were identified without providing the plan. In **No Transcript** mode (plan only), **PlanScore** remained relatively high (**0.60**) because almost all recommendations could come from the plan, but **Recall** (**0.17**) was only slightly better than full context. The **Question-Only** mode (no context) performed worst: **PlanScore** and **PersonalScore** were both **0.04**, with **Recall** near **0.00**. In many question-only cases the LLM either returned no course suggestions or unrelated advice.

These trends (Fig. 6) illustrate that providing both transcript and plan context is crucial for relevant recommendations. The higher **PlanScore** and **Recall** in the full context mode confirm that SmartCourse’s advice aligns well with curriculum requirements. Removing the degree plan leaves the model guessing without structure (hence almost no plan coverage), while removing the transcript (providing only the plan) tended to produce plan-based suggestions but without regard to the student’s actual performance or prerequisites.

Although the **noTranscript** mode (plan only) yielded slightly higher **PlanScore** (**0.60** vs. **0.53**) and **Recall** (**0.17** vs. **0.15**) than full context, we interpret this result with caution. This reflects that, when only the degree plan is available, the model tends to suggest courses directly from the plan, without accounting for the student’s progress or past performance.

TABLE I
AVERAGE RECOMMENDATION QUALITY ACROSS 25 ADVISING QUESTIONS UNDER FOUR CONTEXT SETTINGS.

Mode	#Rec	PlanScore	PersonalScore	Lift	Recall	Latency (s)
full	6.56	0.53	0.78	0.25	0.15	47.65
noPlan	2.24	0.03	0.19	0.16	0.01	25.36
noTranscript	6.20	0.60	0.69	0.09	0.17	34.34
question	0.04	0.04	0.04	0.00	0.00	21.52

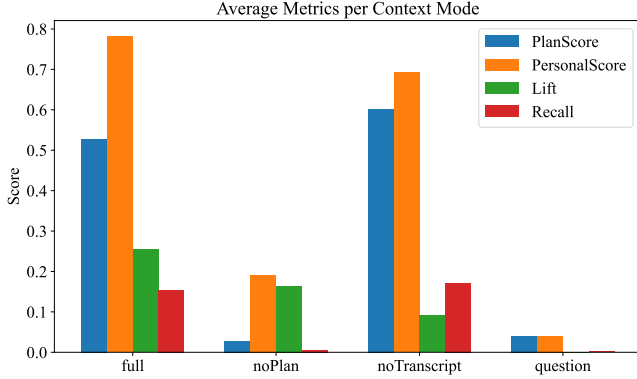


Fig. 6. Comparison of average evaluation metrics (PlanScore, PersonalScore, Lift, Recall) across the four context modes. Notably, the full-context mode yields the highest PersonalScore and Lift, while PlanScore is slightly lower than the plan-only mode. This reflects the tradeoff between personalized and strictly curriculum-based recommendations.

In contrast, the full context includes the student’s transcript, enabling more personalized suggestions—such as repeating a low-grade course or avoiding completed ones. This nuance reduces raw alignment scores like **PlanScore**, but enhances overall recommendation quality, as captured by the higher **PersonalScore (0.78)** and **Lift (0.25)** in full context.

VII. DISCUSSION, LIMITATIONS, AND FUTURE WORK

A. Discussion

The experimental results highlight the value of SmartCourse’s design. With full context, the system achieved the best overall recommendation performance across key metrics. It consistently suggested appropriate and personalized courses that help the student progress toward graduation. The LLM consistently suggested relevant electives, including upper-level courses and appropriate retakes based on the student’s transcript. It also occasionally proposed retaking a course in which the student earned a low grade, which is a personalized insight a human advisor might provide. In contrast, the No Plan mode essentially lost track of degree requirements, producing mostly irrelevant electives. The near-zero recommendations in the Question-Only mode underline that the LLM needs structured context to answer these academic advising queries meaningfully.

Qualitatively, the advice in full-context mode was coherent. The LLM generally followed logical prerequisites and course progression. We observed, however, occasional hallucinations:

suggestions that were not in the official course list or not relevant to the major. For example, in one case the model mentioned a non-CPS elective outside the curriculum. These instances reflect known LLM limitations: without explicit filtering, the model can invent plausible but incorrect items [17]. In practice, we mitigate this by post-filtering recommended courses against the known course catalog.

B. Current Limitations

Despite encouraging results, several limitations should be acknowledged:

- **Limited Evaluation Scope:** Our experiments used a single student profile and a fixed set of hypothetical questions. Real student needs are far more diverse. The metrics here capture curriculum alignment but not student satisfaction or learning outcomes. More extensive testing with varied majors, true student data, and feedback would be needed.
- **LLM Hallucinations and Bias:** As noted, the LLM can produce incorrect or inappropriate suggestions. We rely on filtering to remove non-existent courses, but subtle biases (e.g., favoring popular electives) may persist. Future work must carefully monitor and correct AI biases.
- **Privacy and Data Security:** SmartCourse processes sensitive academic records. In a deployed environment, strict compliance with FERPA [18] / GDPR [19] is essential. (For example, student data must be encrypted and access controlled.) These issues are not fully addressed in the prototype.
- **Incomplete Evaluation Metrics:** Our relevance metrics assume the degree plan is “ground truth.” They do not penalize the system for omitting useful elective suggestions outside the plan. Also, in edge cases where a student has finished nearly all courses, Recall becomes ill-defined. Metrics like precision or qualitative user studies are needed for a fuller evaluation.
- **Latency:** The response time in full-context mode (~48 seconds) remains the highest among all settings, which may limit real-time usability in practice. While not affecting correctness, it limits usability in interactive advising. Potential optimizations are discussed in Section VII-C.

These limitations point to future research directions. In particular, robustness against AI errors, scalability to multiple programs, and compliance with educational data standards will be critical for real-world use.

C. Directions for Improvement

Building upon these limitations, we outline several directions to strengthen SmartCourse's robustness, personalization, and usability.

- **Broader Major Support:** Currently we use a fixed CPS plan. We aim to generalize SmartCourse to other majors by loading corresponding plan files. This will require curating degree requirements for each program.
- **User Feedback Loop:** Incorporating student or advisor feedback (e.g. a "like / dislike" on recommendations) could refine future suggestions.
- **Rich Contextual Data:** Beyond transcripts, incorporating extracurricular activities or official course descriptions could ground suggestions more effectively.
- **Interface Enhancements:** Developing a full web portal (beyond Gradio) with interactive schedule planners, and real-time notification dashboards, would improve usability.
- **Performance Optimization:** The high latency observed under full context mode could hinder real-time use. Techniques such as prompt caching, model distillation, or asynchronous prefetching could mitigate this [20].

SmartCourse represents a first step toward AI-augmented academic advising. While the current results validate its core design, transitioning from research prototype to institutional deployment will require empirical evaluation through pilot studies with real students and advisors.

VIII. CONCLUSION

We have presented SmartCourse, a novel academic advising system that integrates course administration with AI-driven recommendations. By leveraging a student's transcript and four-year plan, SmartCourse generates contextually relevant course suggestions for a variety of academic queries. Our architecture (Fig. 1) combines traditional enrollment management with an AI engine, and experimental results confirm that context-aware advising significantly outperforms context-free modes. While challenges remain (LLM reliability, data privacy, scalability), SmartCourse demonstrates the promise of combining institutional data with language models to enhance student guidance. Future work will focus on system refinement, user studies, and broader integration with university information systems. Future extensions will explore integration with student feedback loops and university portals.

REFERENCES

- [1] J. K. Drake, "The role of academic advising in student retention and persistence," *About Campus*, vol. 16, no. 3, pp. 8–12, 2011. doi: 10.1002/abc.20062.
- [2] J. P. Campbell, P. B. DeBlois, and D. G. Oblinger, "Academic Analytics: A New Tool for a New Era," *EDUCAUSE Review*, vol. 42, no. 4, pp. 40–57, Jul./Aug. 2007. Available: <https://er.educause.edu/articles/2007/7/academic-analytics-a-new-tool-for-a-new-era>
- [3] G. Siemens, "Learning analytics: The emergence of a discipline," *American Behavioral Scientist*, vol. 57, no. 10, pp. 1380–1400, 2013. doi: 10.1177/0002764213498851.
- [4] S. Abdelhamid, J. Bangura, and S. Shah, "Advisely: AI-Powered Academic Advising Using Large Language Models (LLMs)," in *Proceedings of the 14th International Conference on New Perspectives in Science Education (NPSE'14)*, Florence, Italy, Mar. 21, 2025. doi: 10.18608/npse14.7058.
- [5] A. K. Goel and L. Polepeddi, "Jill Watson: A virtual teaching assistant for online education," in C. Dede, J. Richards, and B. Saxberg, Eds., *Learning Engineering for Online Education: Theoretical Contexts and Design-Based Examples*, 1st ed., Routledge, 2018, ch. 7. doi: 10.4324/9781351186193-7.
- [6] A. Elbadrawy and G. Karypis, "Domain-aware grade prediction and top-n course recommendation," in *Proceedings of the 10th ACM Conference on Recommender Systems (RecSys '16)*, Boston, MA, USA, 2016, pp. 183–190. doi: 10.1145/2959100.2959133.
- [7] J. Xu, T. Xing, and M. van der Schaar, "Personalized course sequence recommendations," *IEEE Transactions on Signal Processing*, vol. 64, no. 20, pp. 5340–5352, Oct. 2016. doi: 10.1109/TSP.2016.2595495.
- [8] A. Esteban, A. Zafra, and C. Romero, "Helping university students to choose elective courses by using a hybrid multi-criteria recommendation system with genetic optimization," *Knowledge-Based Systems*, vol. 194, p. 105385, Apr. 2020. doi: 10.1016/j.knsys.2019.105385.
- [9] E. Shao, S. Guo, and Z. Pardos, "Degree planning with PLAN-BERT: Multi-semester recommendation using future courses of interest," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 17, pp. 14920–14929, May 2021. doi: 10.1609/aaai.v35i17.17751.
- [10] H. Van Deventer, M. Mills, and A. Evrard, "From interests to insights: An LLM approach to course recommendations using natural language queries," arXiv preprint arXiv:2412.19312, 2024. Available: <https://arxiv.org/abs/2412.19312>.
- [11] E. Kasneci, K. Sessler, S. Küchemann, M. Bannert, D. Dementieva, F. Fischer, U. Gasser, G. Groh, S. Günnemann, E. Hüllermeier, S. Krusche, G. Kutyniok, T. Michaeli, C. Nerdel, J. Pfeffer, O. Poquet, M. Sailer, A. Schmidt, T. Seidel, M. Stadler, J. Weller, J. Kuhn, and G. Kasneci, "ChatGPT for good? On opportunities and challenges of large language models for education," *Learning and Individual Differences*, vol. 103, p. 102274, 2023. doi: 10.1016/j.lindif.2023.102274.
- [12] H. Li, J. Yu, X. Cong, Y. Dang, D. Zhang-li, Y. Zhan, H. Liu, and Z. Liu, "Exploring LLM-based student simulation for metacognitive cultivation," arXiv preprint arXiv:2502.11678, 2025. Available: <https://arxiv.org/abs/2502.11678>.
- [13] K. Lekan and Z. A. Pardos, "AI-Augmented Advising: A Comparative Study of GPT-4 and Advisor-based Major Recommendations," *Journal of Learning Analytics*, vol. 12, no. 1, pp. 110–128, Mar. 2025. doi: 10.18608/jla.2025.8593.
- [14] A. Aguila, N. Nhan, N. D. Nguyen, K. T. Huynh, A. Mai, L. Tan, N. Tan, and T. Nguyen, "Large language model in higher education: Leveraging Llama2 for effective academic advising," in *Proceedings of the 2024 International Conference on Advanced Technologies for Communications (ATC'2024)*, Ho Chi Minh City, Vietnam, Oct. 2024. doi: 10.1109/ATC63255.2024.10908321.
- [15] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl, "Evaluating collaborative filtering recommender systems," *ACM Transactions on Information Systems*, vol. 22, no. 1, pp. 5–53, Jan. 2004. doi: 10.1145/963770.963772.
- [16] G. Shani and A. Gunawardana, "Evaluating recommendation systems," in F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, Eds., *Recommender Systems Handbook*, Boston, MA: Springer US, 2011, pp. 257–297. doi: 10.1007/978-0-387-85820-3_8.
- [17] Z. Ji, N. Lee, R. Frieske, T. Yu, D. Su, Y. Xu, E. Ishii, Y. J. Bang, A. Madotto, and P. Fung, "Survey of hallucination in natural language generation," *ACM Computing Surveys*, vol. 55, no. 12, art. 248, pp. 1–38, Mar. 2023. doi: 10.1145/3571730.
- [18] U.S. Department of Education, "Protecting student privacy," *Student Privacy Policy Office, Privacy Technical Assistance Center*, [Online]. Available: <https://studentprivacy.ed.gov/ferpa>
- [19] B. Wolford, "What is GDPR, the EU's new data protection law?," *GDPR.eu*, [Online]. Available: <https://gdpr.eu/what-is-gdpr/>
- [20] Z. Xi et al., "The rise and potential of large language model based agents: A survey," arXiv preprint arXiv:2309.07864, 2023. Available: <https://arxiv.org/abs/2309.07864>.