

INTEGRATION OF AI INTO PEER FEEDBACK: STUDENTS' PERCEPTIONS  
AND EXPERIENCES IN A PROGRAMMING COURSE

A THESIS SUBMITTED TO  
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES  
OF  
MIDDLE EAST TECHNICAL UNIVERSITY

BY

MOTAHHAREH BASHIRZADEH

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS  
FOR  
THE DEGREE OF MASTER OF SCIENCE  
IN  
COMPUTER EDUCATION AND INSTRUCTIONAL TECHNOLOGY

APRIL 2026



Approval of the thesis:

**INTEGRATION OF AI INTO PEER FEEDBACK: STUDENTS'  
PERCEPTIONS AND EXPERIENCES IN A PROGRAMMING COURSE**

submitted by **MOTAHHAREH BASHIRZADEH** in partial fulfillment of the requirements for the degree of **Master of Science in Computer Education and Instructional Technology, Middle East Technical University** by,

Prof. Dr. Naci Emre Altun  
Dean, **Graduate School of Natural and Applied Sciences** \_\_\_\_\_

Prof. Dr. Saniye Tuğba Tokel  
Head of the Department, **Computer Education  
and Instructional Technology** \_\_\_\_\_

Assoc. Prof. Dr Erkan Er  
Supervisor, **Computer Education  
and Instructional Technology, METU** \_\_\_\_\_

**Examining Committee Members:**

Prof. Dr. İbrahim Soner Yıldırım  
Computer Education and Instructional Technology, METU \_\_\_\_\_

Assoc. Prof. Dr. Erkan Er  
Computer Education and Instructional Technology, METU \_\_\_\_\_

Assoc. Prof. Dr. Alper Bayazıt  
Department of Medical Education and Informatics, Ankara  
University \_\_\_\_\_

Date: 21.04.2026

**I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.**

Name Last name : Motahhareh Bashirzadeh

Signature :

## **ABSTRACT**

### **INTEGRATION OF AI INTO PEER FEEDBACK: STUDENTS' PERCEPTIONS AND EXPERIENCES IN A PROGRAMMING COURSE**

Bashirzadeh, Motahhareh  
Master of Science, Computer Education and Instructional Technology  
Supervisor : Assoc. Prof. Dr. Erkan Er

April 2026, 126 pages

As Artificial Intelligence (AI) becomes increasingly integrated into Computer Science Education (CSE), many existing approaches have focused on automating feedback processes, primarily emphasizing error detection and correction. However, this focus often overlooks the pedagogical importance of formative feedback that supports student understanding. This study investigates the role of AI not as a replacement for human instruction, but as a supportive "co-pilot" designed to enhance the quality and depth of peer feedback.

The research consists of two main components. First, a systematic literature review on feedback mechanisms identifies a gap where automation frequently prioritizes efficiency over meaningful engagement. Based on these findings, the study proposes the Multi-Layered AI-Augmented Peer Feedback (MLA-PF) framework, consisting of three layers: diagnostic (identifying structural and conceptual issues), augmentation (supporting constructive peer feedback generation), and human-in-the-loop (ensuring pedagogical alignment).

Second, an AI-assisted platform grounded in this framework was evaluated through a pilot study. Findings indicate that students perceived the AI-assisted feedback as effective, with qualitative evidence suggesting meaningful engagement rather than superficial correction. Students made substantive revisions to logical errors and control flow, with over 85% incorporating feedback into their work. By repositioning AI as a tool to scaffold critical thinking, this research offers a grounded approach to enhancing programming education, providing a foundation for future scalability and impact on learning outcomes.

**Keywords:** Artificial Intelligence (AI), Automated Feedback, Peer Feedback, Programming Education, Formative feedback.

## ÖZ

### **PROGRAMLAMA DERSİNDE AKRAN GERİ BİLDİRİMİNE YAPAY ZEKÂNIN ENTEGRASYONU: ÖĞRENCİLERİN ALGILARI VE DENEYİMLERİ**

Bashirzadeh, Motahhareh  
Yüksek Lisans, Bilgisayar ve Öğretim Teknolojileri Eğitimi  
Tez Danışmanı: Doç Dr. Erkan Er

Nisan 2026, 126 sayfa

Yapay Zeka (YZ), Bilgisayar Bilimleri Eğitimi'ne her geçen gün daha fazla entegre olurken, mevcut yaklaşımların çoğu öncelikle hata tespiti ve düzeltilmesine odaklanarak geri bildirim süreçlerini otomatikleştirmeye yoğunlaşmıştır. Ancak bu odaklanma, öğrencinin anlamasını ve öğrenmesini destekleyen biçimlendirici (formative) geri bildirim pedagojik önemini çoğu zaman göz ardı etmektedir. Bu çalışma, YZ'nin rolünü insan öğretiminin bir alternatifi olarak değil, akran geri bildirim kalitesini ve derinliğini artırmak için tasarlanmış destekleyici bir "yardımcı pilot" (co-pilot) olarak incelemektedir.

Araştırma iki temel bileşenden oluşmaktadır. İlk olarak, geri bildirim mekanizmaları üzerine yapılan sistematik bir literatür taraması, otomasyonun sıklıkla anlamlı katılım yerine verimliliğe öncelik verdiği bir boşluğu ortaya koymaktadır. Bu bulgulara dayanarak çalışma, üç katmandan oluşan Çok Katmanlı YZ Destekli Akran Geri Bildirimi (MLA-PF) çerçevesini önermektedir: tanı katmanı (yapısal ve kavramsal sorunların belirlenmesi), zenginleştirme katmanı (yapıcı akran geri bildirim oluşturulmasının desteklenmesi) ve insan döngüsü katmanı (pedagojik uyumun sağlanması).

İkinci olarak, bu çerçeveye dayalı YZ destekli bir akran geri bildirim platformu geliştirilmiş ve bir pilot çalışma aracılığıyla değerlendirilmiştir. Bulgular, öğrencilerin YZ destekli geri bildirim etkili ve yararlı bulduklarını göstermekte; nitel ve davranışsal kanıtlar ise yüzeysel düzeltmelerden ziyade anlamlı bir etkileşime işaret etmektedir. Öğrenciler; mantıksal hatalar, kontrol akışı ve okunabilirlik gibi konularda önemli kod revizyonları yapmış ve %85'ten fazlası geri bildirimleri çalışmalarına dahil ettiklerini belirtmiştir. YZ'yi hazır çözümler sunan bir araç yerine eleştirel düşünmeyi destekleyen bir yapı iskelesi (scaffold) olarak konumlandıran bu araştırma, programlama eğitimi geliştirmek için pedagojik temelli bir yaklaşım sunmaktadır.

Anahtar Kelimeler: Yapay Zekâ (YZ), Otomatik Geri Bildirim, Akran Geri Bildirimi, Programlama Eğitimi, Biçimlendirici Geri Bildirim.

## Dedication

To my family,

For your love, support, and unwavering belief in me.

And in loving memory of my dear uncle.

## **ACKNOWLEDGMENTS**

I would like to express my sincere gratitude to my supervisor for his guidance, support, and kindness throughout this study. I am also grateful for the opportunity to contribute as a member of the TÜBİTAK project under his supervision.

Finally, I would like to thank my family for their unlimited love, encouragement, and unwavering support throughout this journey. Their belief in me has been my greatest source of strength

This study is supported by the Scientific and Technological Research Council of Türkiye (TÜBİTAK) under the 3501 – Career Development Program, project number 323K213 (Project title: “Implementation of Large Language Models in Dialogue-Based Peer Feedback and Investigation of Its Effects”).

## TABLE OF CONTENTS

ABSTRACT.....	v
ÖZ      vii	
ACKNOWLEDGMENTS .....	x
TABLE OF CONTENTS.....	xi
LIST OF TABLES .....	xv
LIST OF FIGURES .....	xvii
LIST OF ABBREVIATIONS .....	xviii
CHAPTERS	
1      INTRODUCTION .....	1
1.1    Background of the Study .....	1
1.2    Statement of the Problem.....	3
1.3    Purpose of the Study .....	5
1.4    The Significance of the Study.....	6
2      LITERATURE REVIEW .....	9
2.1    INTRODUCTION .....	9
2.2    Theoretical Foundations of Formative Feedback .....	11
2.3    The Dynamics of Peer Feedback .....	13
2.4    Generative AI (GenAI) in Education.....	14
2.5    The Potential of AI in Peer Feedback .....	16
2.6    The Research Gap .....	17
3      METHODOLOGY .....	19
3.1    Phase 1 Methodology (Addressing RQ1): .....	20
3.1.1   Search Strategy .....	20

3.1.2	Inclusion And Exclusion Criteria .....	20
3.1.3	Identification of Relevant Publications .....	21
3.1.4	Data Extraction .....	24
3.2	Phase 2 Methodology (Addressing RQ2 & RQ3): .....	25
3.2.1	Research Design .....	25
3.2.2	Context, Participants and Sampling .....	26
3.2.3	Data Collection Instruments and Process .....	26
3.2.4	Validity and Reliability .....	27
3.2.5	Data Analysis.....	28
3.2.6	AI-Based Peer Feedback Platform .....	29
4	RESULTS AND DISCUSSION.....	31
4.1	Results of Systematic Literature Review (Answering RQ1).....	31
4.1.1	Analysis of Research Goals in Reviewed Studies.....	31
4.1.2	Analysis of Theoretical Grounding of Feedback Practices in Reviewed Studies .....	32
4.1.3	Analysis of Research Methodologies Employed in Reviewed Studies 34	
4.1.4	Analysis of Research Data Collected in Reviewed Studies .....	35
4.1.5	Analysis of Instructor Roles in Feedback Process in Reviewed Studies	37
4.1.6	Analysis of Technologies Used for Feedback Provision in Reviewed Studies	38
4.1.7	Analysis of Feedback Enhancement Strategies in Reviewed Studies	40
4.1.8	Analysis of Students' use of Feedback for Revision in Reviewed Studies	42

4.1.9	Analysis of Research Findings in Reviewed Studies.....	44
4.1.10	Analysis of Research Implications in Reviewed Studies.....	46
4.1.11	Synthesis of the Literature Review .....	49
4.1.12	A Conceptual Framework For AI-Augmented Peer Feedback in Programming Education .....	51
4.1.13	Alignment of the Proposed Framework with Research Gaps .....	62
4.1.14	Operationalization of the Framework: The AI-Based Peer Feedback Platform66	
4.2	Results of Pilot Study (Answering RQ2 & RQ3) .....	85
4.2.1	Result of Thematic Analysis (Qualitative Findings).....	86
4.2.2	Perceived Strengths and Weaknesses of the Feedback Process.....	86
4.2.3	Technical Accuracy and Helpfulness of Feedback .....	90
4.2.4	Clarity and Understandability of Feedback .....	92
4.2.5	Changes Made to Code Based on Feedback .....	93
4.2.6	Suggested Improvements to the Feedback Activity.....	95
4.3	Quantitative Results .....	97
4.3.1	Descriptive Statistics.....	97
4.3.2	Frequency of Feedback Usage .....	98
5	DISCUSSION .....	101
5.1	Pedagogical Grounding of the MLA-PF Framework (RQ1) .....	101
5.2	Perceived Effectiveness and Usefulness of AI-Assisted Feedback	102
5.3	Students' Experiences with AI-Assisted Peer Feedback .....	103
5.4	Integration of Mixed-Methods Findings .....	104
5.5	Implications and Limitations .....	104

REFERENCES .....	107
A. Student Questionnaire .....	117

## LIST OF TABLES

### TABLES

Table 3.1 Inclusion and Exclusion Criteria.....	20
Table 3.2 Case processing summary for the reliability analysis.....	27
Table 3.3 Reliability Statistics (Cronbach’s Alpha) .....	28
Table 4.1 Frequency and percentage of research goals .....	31
Table 4.2 Frequency of Theoretical and Conceptual Frameworks .....	33
Table 4.3 Frequency and percentage of research methodologies .....	34
Table 4.4 Frequency and percentage of data collection approaches.....	36
Table 4.5 Code frequencies about the instructors’ role .....	38
Table 4.6 Frequency and Percentage of Feedback Delivery Types.....	39
Table 4.7 Frequency and Percentage of Feedback Enhancement Strategies .....	41
Table 4.8.Frequency and Percentage of Students’ Feedback Use .....	43
Table 4.9.Frequency of Themes and Specific Findings.....	45
Table 4.10 Frequency of Themes and Specific Implications.....	46
Table 4.11.Roles and Responsibilities within the MLA-PF Cycle .....	61
Table 4.12.Alignment Between the Proposed Framework and the Literature Review. ....	62
Table 4.13.Thematic Analysis of Perceived Strengths of the Feedback Process ....	86
Table 4.14.Thematic Analysis of Perceived Weaknesses of the Feedback Process	88
Table 4.15 Quality of the Feedback .....	90
Table 4.16.Thematic Analysis of Feedback Clarity and Understandability .....	92
Table 4.17. Thematic Analysis of Code Changes Based on Feedback.....	94

Table 4.18. Thematic Analysis of Suggested Improvements .....	95
Table 4.19. Descriptive Statistics for the AI Experience Variable.....	97
Table 4.20. Descriptive Statistics for Feedback Usage Frequency .....	98
Table 4.21 How often did you take into consideration feedback comments (including the chatbot) when you revised your code?.....	98

## LIST OF FIGURES

### FIGURES

Figure 3.1 The flowchart of the screening and selection process .....	23
Figure 4.1 Landing page of the Platform .....	67
Figure 4.2 Instructor dashboard page.....	68
Figure 4.3.Create New Course .....	69
Figure 4.4 Managing enrollments .....	70
Figure 4.5 Creating new Assignment.....	71
Figure 4.6.Disabling or Enabling AI Features .....	72
Figure 4.7.Disabling or Enabling AI Features .....	72
Figure 4.8.Creating Rubric.....	73
Figure 4.9.Assessment Criteria .....	73
Figure 4.10 Managing Peer Reviews .....	74
Figure 4.11 Managing Peer Reviews .....	74
Figure 4.12 Submissions list .....	75
Figure 4.13 Getting AI Review .....	76
Figure 4.14 Student Page Dashboard.....	77
Figure 4.15.Submitting Assignment .....	78
Figure 4.16.Assignment Details.....	79
Figure 4.17 Submission Details .....	79
Figure 4.18 Peer review form .....	80
Figure 4.19.AI Review The Analysis.....	81
Figure 4.20.Getting Suggestion From Ai.....	81
Figure 4.21.Getting Suggestion From AI .....	82
Figure 4.22.Feedback Details .....	82
Figure 4.23 Peer feedback page .....	83
Figure 4.24 Chatbot for criteria-specific feedback .....	84
Figure 4.25 Example chatbot interactions for criteria-specific feedback .....	85

## **LIST OF ABBREVIATIONS**

### **ABBREVIATIONS**

**AI** Artificial Intelligence

**LLM** Large Language Model.

**MLA-PF** Multi-Layered AI-Augmented Peer Feedback

**MKO** More Knowledgeable Other

**SRL** Self-Regulated Learning

**ZPD** Zone of Proximal Development

# CHAPTER 1

## INTRODUCTION

### 1.1 Background of the Study

Learning outcomes are an essential part of education, and feedback plays a critical role in improving these outcomes. Accordingly, feedback is considered to have a powerful influence on learning and achievement (Hattie & Timperley, 2007). Feedback provides essential information to students about their performance and understanding, with the goal of enhancing their learning processes and task performance (Nicol & Macfarlane-Dick, 2006). Feedback focus can be numerous ranging from the tasks to processes, from self-regulation to motivation (Hattie & Timperley, 2007).

Feedback can be categorized as formative or summative feedback (Bhat & Bhat, 2019; Shute, 2008). Formative feedback refers to the guidance during the learning process to help students improve their work before it is finalized (Shute, 2008), while summative feedback is an evaluation provided at the end of a task to judge performance, often resulting in a grade (Brookhart, 2018). While summative feedback provides a clear benchmark of achievement, the weight of research evidence favors formative feedback for its ability to help students actively bridge the gap between their current and desired performance, thereby fostering long-term learning (Sadler, 1998).

In the literature, (formative) feedback is predominantly provided by either teachers or peers (Sadler, 1998). While the role of feedback in learning is unquestionable regardless of the feedback source, in practice, there are concerns about the quality of feedback, its impact on learning, and students' perceptions of it in higher education

(Hattie & Timperley, 2007). For example, while teacher feedback is known to be more accurate (Sadler, 1998), it is often less understood by students due to its advanced academic language, thus, leading to low uptake of feedback (Carless & Boud, 2018a) On the other hand, peer feedback is often more accessible and easier for students to understand because it uses shared, non-technical language (Cho & MacArthur, 2010). However, students may perceive it as less authoritative or accurate than teacher feedback, causing a reluctance to trust and apply it (Wei & Liu, 2024). As a result, current discussions in the literature increasingly argue that traditional feedback models are insufficient (Boud & Molloy, 2013a; Winstone & Carless, 2019a), suggesting that the practice of feedback needs to be fundamentally transformed to overcome these persistent limitations and realise the true potential of feedback.

One educational domain where feedback is particularly critical is computer science, specifically within programming courses (Keuning et al., 2018). Programming tasks require continuous, iterative efforts in writing, debugging, and revising code. During this process, formative feedback is essential for students' productive struggle and facilitating learning (Keuning et al., 2018). Notably, the literature on programming feedback diverges from general educational research regarding the source of feedback (Keuning et al., 2018; Sadler, 1998). Instead of a human-centric approach, the field has largely relied on technology, with a heavy emphasis on developing systems for the automated assessment of code (Combéfis, 2022). While automation may have its own advantages, especially in facilitating feedback for large classrooms (Keuning et al., 2019), this approach often lacks the pedagogical roots and principles deeply established in the broader educational feedback field.

The recent advancements in artificial intelligence (AI), particularly in generative AI (GenAI) have had a transformative effect on the field of education (Francis et al., 2025). One area where GenAI can have a particularly significant impact is feedback, as this technology possesses the unique capability to mimic human-like language and generate detailed, context-aware responses. Consequently, in the last few years, there has been an abundance of research investigating the potential of GenAI to

enhance feedback practices (Alier et al., 2024). Yet, it remains unclear how to leverage this powerful technology in a pedagogy-grounded way to scaffold humans in feedback processes effectively. Specifically, there is limited understanding of how GenAI can support peer feedback to mitigate its limitations while preserving its collaborative learning benefits. Addressing this gap, as elaborated in the following sections, this thesis investigates the design and impact of a pedagogically grounded, GenAI-supported peer feedback approach in introductory programming education.

## **1.2 Statement of the Problem**

Feedback plays a critical role in programming education, where learning is inherently iterative and error-driven. In programming contexts specifically, feedback helps students identify bugs, recognize misconceptions about computational logic, evaluate the correctness and efficiency of their code, and regulate their problem-solving strategies (Keuning et al., 2018; Qian & Lehman, 2017; Hattie & Timperley, 2007). Beyond merely evaluating whether code produces a correct output, effective feedback must guide students toward understanding why their code fails and how to improve it, which can be achieved by formative feedback (Shute, 2008). Formative feedback in programming has been shown to enhance students' self-regulation (Nicol & Macfarlane-Dick, 2006), which is a key competency for novice programmers. As (Keuning et al., 2018) emphasize, the quality and specificity of feedback in programming tools significantly influence whether students develop deep understanding or merely surface-level fixes, underscoring the need for feedback mechanisms that go beyond pass/fail test results to provide meaningful, actionable guidance.

A very reliable feedback source is instructors since they can provide comprehensive and meaningful feedback to students as subject matter experts. However, teacher feedback is not always feasible especially in programming courses, where enrollment numbers continue to increase annually (Paris, 2022). This increase creates significant practical constraints that limits the instructor's ability to provide

timely feedback and interact with every student (Castellanos et al., 2026). Consequently, although instructor feedback remains educationally desirable, it is often not scalable in contemporary learning environments.

Unsurprisingly, automated assessment systems have become the predominant feedback mechanism in programming education (Ihantola et al., 2010; Paiva et al., 2022). While originally developed for industry contexts and widely adopted to scale feedback delivery (Douce et al., 2005), this approach introduces significant pedagogical limitations (Keuning et al., 2018; Messer et al., 2024). The most critical drawback is that these systems prioritize technical correctness and output accuracy over learning quality (Hahn et al., 2021). Instead of addressing students' reasoning processes, they primarily evaluate whether code executes without errors or bugs. Consequently, they often fail to provide the formative guidance necessary to help learners understand the underlying causes of their mistakes, offering limited educational support beyond simple evaluation (Paiva et al., 2025; Akbari Pordanjani & Salehi, 2025).

One practical solution to address the limitations of both instructor constraints and automated assessment is peer feedback. This approach offers a middle ground, reducing teacher workload while injecting the human insight that automated systems lack. It is a particularly good fit for programming education, given the established success of collaborative methodologies like pair programming (Williams et al., 2000). By allowing students to evaluate each other's work, peer feedback promotes reflection, engagement, and the same collaborative problem-solving skills valued in the industry (Topping, 2009).

Nevertheless, the efficacy of peer feedback is often undermined by issues of reliability and validity. A primary concern is that students, as novices, often lack the deep subject matter expertise required to evaluate complex concepts accurately (Panadero & Alqassab, 2019). Consequently, their feedback tends to focus on surface-level features rather than underlying logic, leading to inconsistencies compared to expert evaluations. To mitigate the risk of inaccurate or biased reviews,

instructors must invest significant time in scaffolding and training students to provide meaningful feedback, making the process resource-intensive to implement effectively (Liu & Carless, 2006).

Recent advances in (GenAI), particularly Large Language Models (LLMs), introduce a promising technological solution to these challenges. Unlike traditional automated systems, LLMs can generate natural-language explanations, allowing feedback to function as conversational guidance rather than mere evaluation (Guo et al., 2024). While studies show GenAI is effective for general feedback, a significant gap remains: there is currently a lack of a clear pedagogical framework to guide how AI should be structured to fulfill the specific role of a “peer” within the programming feedback loop (Banihashem et al., 2025). Without such a framework, AI feedback risks being technically accurate but educationally ineffective, failing to support the social and dialogic aspects of peer-based learning (Banihashem et al., 2025).

Considering these challenges, a major gap in programming education persists: the absence of a feedback mechanism that is both pedagogically guided and highly scalable (Banihashem et al., 2025). Existing methods either rely heavily on instructors and peers, limiting consistency and reach, or depend on automated systems that offer limited support for deep conceptual understanding (Hahn et al., 2021; Panadero & Alqassab, 2019).

### **1.3 Purpose of the Study**

The primary goal of this study is to propose design guidelines for AI-assisted peer feedback in higher education programming courses. These guidelines are then used as the pedagogical foundation for the development of a peer feedback platform to ensure that AI is integrated into the feedback process in a pedagogically grounded way. Furthermore, this study aims to pilot the AI-assisted peer feedback platform in a classroom setting to explore students’ perceptions and experiences. To achieve these goals, the following research questions are addressed:

- RQ1: What are the pedagogical design guidelines for effectively integrating Generative AI into peer feedback systems for programming education?
- RQ2: How do students perceive the effectiveness and usefulness of AI assisted peer feedback in improving code functionality, organization, and style?
- RQ3 : What are students' experiences of AI assisted peer feedback?

The first question is answered through a systematic literature review, while the other two questions were addressed through piloting the peer feedback platform.

#### **1.4 The Significance of the Study**

This study makes several contributions to the fields of programming education and AI-assisted learning, addressing gaps at the intersection of pedagogy, technology, and feedback practice.

First, the study contributes a pedagogical framework for AI assisted peer feedback that is specifically designed for the programming education context. While existing research has explored the use of generative AI for providing feedback, much of this work has been technology-driven, focusing on what AI can do rather than what it should do from a learning perspective (Banihashem et al., 2025). By grounding the design of the AI feedback system in established feedback and learning theories, this study moves beyond ad hoc implementations and offers a principled, replicable model for structuring AI as a peer feedback agent. This framework can inform future researchers and practitioners seeking to integrate AI into feedback processes in a pedagogically meaningful way.

Second, the study bridges the gap between theoretical design and practical implementation. The pedagogical framework is not presented as an abstract model; rather, it is operationalized through the development of a functional online platform in which a LLM scaffolds feedback activities. This design-to-implementation approach demonstrates how theoretical principles can be translated into a working

learning environment, providing a concrete example for educators and developers aiming to build similar systems.

Third, the study generates some evidence on students' perceptions and experiences with AI assisted peer feedback in programming courses. Despite growing interest in AI-generated feedback, there remains limited research examining how students actually experience and respond to AI acting in a peer-like role, particularly in the context of code review (Guo et al., 2024). By capturing student perspectives on the usefulness, effectiveness, and overall experience of interacting with the AI-supported platform, this study offers initial insights that can guide future iterations of AI-assisted feedback tools and inform instructional decision-making.

Finally, from a practical standpoint, the study addresses a pressing need in programming education: the demand for feedback mechanisms that are both scalable and pedagogically sound. As programming course enrollments continue to rise, the limitations of instructor-only and traditional peer feedback models become increasingly pronounced. By proposing and piloting a model in which AI augments rather than replaces the peer feedback process, this study offers a viable path forward for institutions seeking to maintain feedback quality without placing unsustainable demands on instructors or relying solely on automated assessment tools that lack formative depth.



## CHAPTER 2

### LITERATURE REVIEW

#### 2.1 INTRODUCTION

Effective learning is largely shaped by the quality and timeliness of the feedback students receive. Decades of educational research underscore that when learners are provided with clear and timely guidance, their understanding deepens and academic achievement improves (Hattie & Timperley, 2007). The literature on formative feedback consistently identifies common characteristics of effective feedback, indicating that feedback should be specific, constructive, and provide guidance for improvement (Shute, 2008). However, the optimal form and implementation of feedback may vary depending on the subject area and the nature of students' work. One domain in which feedback practices are particularly distinct and critical is computer programming.

In general terms, effective feedback should clarify the student's current level of understanding, identify the gap between their current state and the desired learning goal, and provide actionable steps to bridge that gap (Hattie & Timperley, 2007). This process fosters metacognition, encouraging students to reflect on their own learning strategies and become more self-regulated learners (D. J. Nicol & Macfarlane-Dick, 2006). However, in computer programming, the feedback process takes on additional complexity. Programming requires students not only to understand abstract computational ideas but also to translate those ideas into working code. As a result, feedback differs from that provided for tasks such as essay writing. Whereas feedback on an essay typically focuses on argumentation and organization, feedback on a programming artifact must address multiple aspects simultaneously, including the learner's understanding of algorithms, the syntactic correctness of the code, its efficiency, and its stylistic quality. (Kerman et al., 2024). For example, a

program might produce the correct output but be inefficiently written, difficult to maintain, or stylistically poor, all crucial aspects for a developing programmer to grasp.

Given the distinct nature of feedback in programming, research in this area has progressed through several stages. Early studies primarily focused on automated, binary feedback simply determining whether the code runs correctly or not (Keuning et al., 2018). This initial approach evolved to provide more pedagogically meaningful information, incorporating feedback on syntax, code style, and efficiency, moving beyond simple correctness (Keuning et al., 2018). More recently, research has shifted toward intelligent tutoring systems and AI-driven approaches that offer semantic feedback, capable of interpreting the student's intent, identifying logical errors, and providing tailored hints to guide them toward a correct solution (Crow et al., 2018).

This technological trajectory, however, has revealed significant pedagogical challenges. The field is dominated by a technology-centric, human-scarce approach that often prioritizes automation over argumentation and technical correctness over the students' learning experience. This has led to a theoretical disconnect, where feedback tools are rarely grounded in established feedback theories, and a corresponding lack of deep student engagement. In this context, peer feedback emerges as a compelling, yet underexplored, solution. It brings back the human and dialogic elements of feedback that are often absent in fully automated systems. Engaging in peer reviews is not only a scalable practice but also a potent learning activity; students deepen their own understanding by articulating feedback for others, shifting the focus from mere error correction to deeper cognitive engagement and metacognition. The challenge, however, lies in ensuring the quality and consistency of peer feedback.

## **2.2 Theoretical Foundations of Formative Feedback**

Formative feedback is widely recognized as a cornerstone of effective pedagogy, primarily due to its role in bridging the gap between a learner's current performance and their desired learning goals (Sadler, 1989). This section introduces three foundational concepts of feedback: the Zone of Proximal Development (ZPD), Scaffolding, and Self-Regulated Learning (SRL).

The theoretical foundations of formative feedback are grounded in sociocultural learning theory, particularly the Zone of Proximal Development (ZPD) proposed by Lev Vygotsky. The ZPD describes the distance between what learners can achieve independently and what they can accomplish with guidance from a More Knowledgeable Other (MKO) (Vygotsky, 1978). Within this framework, learning is supported through scaffolding, which refers to temporary guidance that helps learners progressively develop their understanding and skills (Wood et al., 1976). In collaborative learning environments, such scaffolding can emerge through peer interactions, where students provide feedback and support each other's learning processes (Topping, 2005). However, because peers often possess similar levels of expertise, the quality and effectiveness of scaffolding may vary. Consequently, structured mechanisms that guide feedback interactions may be necessary to ensure that peer feedback remains constructive and supportive of learning within the learner's ZPD.

Closely associated with the Zone of Proximal Development is the concept of scaffolding, which refers to the temporary support provided to learners as they develop new competencies (Wood et al., 1976). Scaffolding enables learners to accomplish tasks that they would not be able to complete independently and gradually supports the development of higher levels of understanding. Effective scaffolding is adaptive, providing more intensive guidance during the early stages of a task and progressively fading as learners gain autonomy and competence (Wood et al., 1976). In collaborative learning environments, scaffolding may emerge through peer interactions in which learners guide and support one another's understanding

(Hmelo-Silver et al., 2007). Through discussion, explanation, and feedback, peers can contribute to each other's learning processes and help structure problem-solving activities. However, because peers often possess similar levels of expertise, the quality and effectiveness of such scaffolding may vary. Within AI-assisted peer feedback contexts, artificial intelligence can provide an additional layer of structured support that helps students organize their critiques and interpret complex suggestions. Rather than simply delivering direct answers, AI systems can guide students in reflecting on their peers' work and constructing more meaningful feedback. In this way, AI-supported feedback can function as a scaffold that promotes deeper cognitive engagement and supports learning processes rather than merely correcting errors. The ultimate objective of formative feedback is to foster Self-Regulated Learning (SRL), a process in which students actively take control of their cognitive, behavioral, and motivational learning processes (Zimmerman, 2000). SRL emphasizes learners' ability to set goals, monitor their progress, regulate strategies, and reflect on their performance throughout the learning process. High-quality feedback functions as an external regulatory signal that encourages students to evaluate their current understanding and adjust their learning strategies accordingly. In this way, feedback supports learners in monitoring their progress and refining their approaches to problem solving and task completion. Within AI-assisted peer feedback environments, artificial intelligence can provide more immediate and consistent feedback cues during the learning process. Such timely feedback can supply learners with additional information about their performance and guide them in reflecting on their work. From an SRL perspective, the availability of rapid and structured feedback can support key phases of the self-regulation cycle, particularly the forethought, performance monitoring, and self-reflection phases (Pintrich, 2000; Zimmerman, 2000). By supporting these processes, feedback mechanisms can help students internalize standards of quality, develop stronger metacognitive awareness, and gradually move toward greater academic independence.

### **2.3 The Dynamics of Peer Feedback**

While peer feedback is a widely adopted pedagogical strategy, its effectiveness is determined by a complex interplay of cognitive and interpersonal variables. Within higher education, peer feedback is defined as a communication process through which learners engage in dialogue about the quality of their work and the standards expected of them (Liu & Carless, 2006).

The literature distinguishes between benefits for the feedback “provider” and the “receiver.” For the receiver, peer feedback offers diverse perspectives and a more accessible linguistic framework than instructor feedback. However, recent studies suggest the most significant cognitive gains are experienced by the provider (Nicol et al., 2014). The act of reviewing requires the evaluator to engage in high-level critical thinking and error detection, which reinforces their own understanding of the subject matter (Topping, 2005).

A primary barrier to the effectiveness of peer feedback is the perceived lack of trust and epistemic authority. Students often exhibit expert-dependency, viewing the instructor as the primary or sole source of valid knowledge within the learning environment. As a result, learners may question the credibility of feedback provided by their peers, leading to situations in which even accurate feedback is ignored (Panadero, 2016; David Nicol & Debra Macfarlane-Dick, 2006). This lack of epistemic trust can prevent students from engaging deeply with peer feedback, thereby limiting its formative potential and reducing opportunities for meaningful reflection and learning.

Because student evaluators are still developing domain knowledge and expertise, they may provide incorrect suggestions or overlook fundamental misconceptions, a challenge sometimes described as “the blind leading the blind” in peer-learning contexts (Roll et al., 2011). As a result, the quality of peer feedback can vary considerably across learners. Research by Lu and Law (2012) indicates that without external scaffolding or technological support, peer feedback quality may remain

inconsistent, making it difficult for students to distinguish between constructive advice and inaccurate or misleading suggestions..

Peer feedback is an inherently social process and is therefore susceptible to interpersonal tensions. Learners may experience feedback anxiety, fearing that providing critical evaluations could negatively affect their social relationships with peers (Strijbos et al., 2021). As a result, students may avoid offering direct criticism in order to maintain positive interpersonal dynamics. These pressures can lead to politeness bias, a phenomenon in which learners provide vague or overly positive comments that lack the specific and actionable detail necessary for meaningful academic improvement (Liu & Carless, 2006). Consequently, socio-emotional dynamics may reduce the depth and effectiveness of peer feedback interactions. These limitations highlight the need for structured mechanisms that can support learners in providing constructive and reliable feedback within peer learning environments (Nicol & Macfarlane-Dick, 2006).

## **2.4 Generative AI (GenAI) in Education**

The emergence of GenAI, particularly LLMs, has introduced new possibilities for automated instructional support in educational contexts. Unlike traditional rule-based systems that rely on predefined responses and structured decision rules, GenAI systems can generate context-aware natural language responses by leveraging large-scale language representations (Brown et al., 2020; Holmes et al., 2019). This capability enables such systems to provide nuanced explanations, feedback, and instructional guidance, making them particularly well suited for complex educational tasks such as formative assessment and feedback generation (Barus et al., 2025).

Recent research highlights the versatility of Large Language Models (LLMs) in replicating complex instructional tasks. Beyond simple automated grading, LLMs can generate hints and provide natural language feedback that can be adapted to individual student needs (Kasneci et al., 2023). Studies also indicate that advanced

language models can achieve substantial agreement with human evaluators when assessing complex student work, such as essays, and can provide feedback that students perceive as comparable to that of human instructors (Zhai, 2022). By identifying specific reasoning errors in student work such as in programming or writing LLMs offer a scalable approach to addressing the feedback bottleneck often present in large higher education courses (D. Nicol, 2010). Current educational frameworks are increasingly shifting from the perspective of “AI as a replacement” toward models of Human–AI collaboration, often described as a Co-Pilot approach. This perspective emphasizes that artificial intelligence should augment rather than replace human agency in learning processes (Fang & Zhang, 2025). Within peer feedback contexts, this means that AI does not replace the peer’s evaluative voice; instead, it functions as a cognitive assistant that supports students in refining their critiques, suggesting areas for improvement, and structuring more constructive feedback. Such human-in-the-loop systems preserve the social and dialogic benefits of peer interaction while leveraging the analytical capabilities and consistency of AI systems (Banihashem et al., 2026a; Molenaar, 2022)

Despite its potential, GenAI is constrained by several technical and ethical challenges. One of the most widely discussed issues is hallucination, where language models generate confident but factually incorrect or misleading information, potentially disrupting the learning process (Kasneci et al., 2023). Furthermore, the black box nature of many large language models, where the reasoning behind a particular response or feedback is not transparent, raises concerns about accountability and explainability in educational contexts. Another important concern involves algorithmic bias. If the training data used to develop AI systems contains historical or social biases, these biases may be reflected in the feedback generated by the system, potentially reinforcing existing educational inequalities (Baker & Hawn, 2022). These ethical and technical limitations highlight the importance of carefully designed AI-supported educational systems that integrate pedagogical guidance and human oversight.

## 2.5 The Potential of AI in Peer Feedback

The convergence of peer assessment and GenAI represents a significant shift in collaborative learning environments. By positioning AI as a mediating agent within the peer feedback loop, researchers aim to preserve the social and dialogic benefits of student interaction while leveraging technological support to address longstanding challenges associated with peer feedback processes. In particular, AI systems can assist in reducing issues related to feedback accuracy, trustworthiness, and cognitive overload by guiding students in producing more structured and constructive feedback (Darvishi et al., 2024; Topping et al., 2025). In this way, AI-assisted peer feedback systems attempt to combine the pedagogical strengths of peer learning with the analytical and generative capabilities of modern AI technologies. A primary function of AI in this intersection is to serve as a pre-check mechanism. Before a student's feedback is sent to their peer, AI can analyze the comment for constructiveness, specificity, and tone (Topping et al., 2025). This real-time quality assessment addresses the "blind leading the blind" problem by flagging erroneous technical suggestions or overly vague praise (e.g., "Good job"). By providing meta-feedback to the reviewer, the AI helps improve the quality of feedback before it reaches the recipient, thereby increasing the overall epistemic trust in the system (Dai et al., 2023).

Moreover, peer assessment is a cognitively demanding task that requires students to simultaneously analyze complex work and formulate coherent critiques. This process can lead to cognitive overload, where the quality of the review declines due to the substantial mental effort required (Sweller, 1988). According to Cognitive Load Theory, learners possess limited working memory resources that can easily be overwhelmed when they must process multiple elements of a complex task at once (John Sweller, 1988). AI-assisted platforms can help reduce this extraneous cognitive load by providing structural scaffolding mechanisms such as adaptive rubrics, sentence starters, or automated summaries of the work being reviewed (Kasneci et al., 2023). These supports allow students to allocate more of their

germane cognitive load toward deeper intellectual evaluation of their peers' ideas rather than the mechanical aspects of composing feedback. By structuring the feedback process in this way, AI-mediated systems can support more effective engagement with peer assessment tasks while promoting meaningful learning.

Several emerging systems have begun to operationalize these theoretical perspectives in practical educational settings. Experimental platforms such as EvaluMate (Guo, 2024) employ Large Language Models (LLMs) to scaffold student feedback and support the peer review process in real time. These systems aim to guide learners in producing more structured and constructive critiques by providing automated prompts, suggestions, or feedback validation mechanisms. Similarly, the RIPPLE platform integrates AI to help equalize feedback quality across large cohorts, ensuring that students receive at least one high-quality, AI-supported review regardless of the varying expertise levels among peers (Topping et al., 2025). Together, these experimental systems demonstrate the growing potential of AI-mediated peer feedback environments and provide an empirical foundation for exploring pedagogically grounded approaches to integrating AI into peer assessment practices.

## **2.6 The Research Gap**

Despite the rapid technological advancements in GenAI and the well-established pedagogical value of peer feedback, a critical disconnect remains in the current literature (Kasneci et al., 2023; Topping, 2005). This gap can be understood along two primary dimensions: the lag between raw technological capabilities and their pedagogically grounded implementation (Fang & Zhang, 2025; Selwyn, 2024), and the absence of standardized design frameworks that guide how AI should be integrated into peer feedback processes.

Furthermore, the increasing adoption of AI in educational contexts raises important questions regarding the role of human instructors in AI-mediated learning

environments. While AI technologies can automate certain feedback functions, excessive reliance on automation risks diminishing the pedagogical role of the instructor and overlooking the importance of human judgment in guiding learning processes (Panadero et al., 2016). Consequently, there is a growing need to move toward augmentation rather than replacement, positioning AI as a supportive co-pilot that enhances human agency rather than substituting for it.

Although existing studies demonstrate that AI systems can generate automated feedback or streamline assessment tasks (Zhai, 2022), relatively little research has explored how AI can be systematically designed to support the interpersonal and reflective dynamics of peer feedback (Lodge et al., 2024). This limitation is particularly evident in programming education, where current AI interventions often prioritize technical correctness rather than supporting the cognitive development of the student reviewer (Prather et al., 2024). Consequently, there remains a need for pedagogically grounded design guidelines that explain how AI can effectively scaffold peer feedback processes while preserving the collaborative and human-centered nature of learning.

To address this gap, the present study develops and evaluates design guidelines for AI-assisted peer feedback in higher education programming courses, focusing on how AI can support dialogue-based peer feedback while maintaining the central role of human instructors in the learning process.

## **CHAPTER 3**

### **METHODOLOGY**

This study adopts a design-oriented research approach (Reeves, 2006), which is suited for educational research that aims to address practical problems through the systematic design and evaluation of learning interventions grounded in theory. Unlike purely experimental designs that test predefined hypotheses, design-oriented research begins with the analysis of an identified educational problem, draws on existing theory and evidence to develop a solution, and evaluates that solution in an authentic learning context. This approach is particularly appropriate for the present study, which addresses a recognized gap in programming education: the absence of a feedback mechanism that is both pedagogically grounded and scalable.

Specifically, the study unfolds in two interconnected phases. In the first phase, a systematic review of the literature was conducted to synthesize existing knowledge on peer feedback, AI in education, and programming pedagogy. The outcome of this phase is a conceptual framework along with a set of pedagogical design guidelines that address the first research question. In the second phase, these guidelines informed the design and development of an online peer feedback platform, which was then piloted in a higher education programming course. During the pilot, both quantitative and qualitative data were collected to examine students' perceptions and experiences, addressing the second and third research questions. This phased structure aligns with the core logic of design-oriented research, in which theoretical analysis directly informs the design of an educational solution, and empirical evaluation provides initial evidence of its value and directions for future refinement.

### **3.1 Phase 1 Methodology (Addressing RQ1):**

Our systematic literature review followed a three-step process: First, we adhered to the PRISMA 2020 guidelines. Next, we used a quality appraisal method from Theelen et al. (2019) to refine our selection. Finally, we analyzed the included studies with a coding scheme designed to address our research questions.

#### **3.1.1 Search Strategy**

A systematic search was conducted in the Web of Science (WoS) Core Collection database, which was chosen for its extensive coverage of literature in the social sciences, computer science, and education. The search query used is detailed as follows: (education OR student) AND (“computer science” OR programming) AND “feedback”. The results were limited to articles indexed in the Social Science Citation Index, Science Citation Index Expanded, or Emerging Sources Citation Index.

#### **3.1.2 Inclusion And Exclusion Criteria**

A set of specific inclusion and exclusion criteria was established prior to the screening process to ensure that only directly relevant studies were included in the final analysis. These criteria, detailed in Table 3.1, were applied across two screening phases

Table 3.1 Inclusion and Exclusion Criteria

Criteria Category	Inclusion Criteria (Must meet all)	Exclusion Criteria (Will be excluded if any are met)
Publication Year	Published between January 1, 2005, and August 1, 2025.	Published before 2005.

Table 3.1 (continued)

Criteria Category	Inclusion Criteria (Must meet all)	Exclusion Criteria (Will be excluded if any are met)
Language	Full text written in English.	Not written in English.
Publication Type	Peer-reviewed journal article with an empirical/experimental design.	Book chapters, conference proceedings, dissertations, technical reports, literature reviews, or conceptual papers.
Context	Study conducted in a higher education (university/college) setting.	Study conducted in a K-12, vocational, or non-educational setting.
Content Focus	Must explicitly investigate a feedback mechanism or practice related to programming education.	Lacks a clear feedback component; focuses solely on system architecture without assessing its impact on learning; focuses only on instructor-led teaching without feedback exploration.

### 3.1.3 Identification of Relevant Publications

The study selection followed a structured, multi-phase process, which is visually summarized in the PRISMA flowchart (see Figure 3.1). Initially, all records identified from the database searches were imported into a reference management software, and 124 duplicate records were removed. The screening was then performed independently by two researchers to minimize bias.

In the first phase, the two researchers screened the titles and abstracts of the remaining 622 articles against the inclusion criteria. Articles that clearly did not meet the criteria were excluded. This phase resulted in the exclusion of 438 articles, leaving 184 for full text review.

In the second phase, the full texts of these 184 articles were retrieved and assessed for eligibility by the same two researchers. During this stage, articles were excluded for not meeting the detailed content criteria (e.g., non-experimental design, wrong context, no feedback component). Disagreements between the researchers at any stage were resolved through discussion and consensus; a third senior researcher was available for arbitration if needed. After the full-text screening, 125 articles were excluded, resulting in a final sample of 59 studies deemed eligible for inclusion in this review.

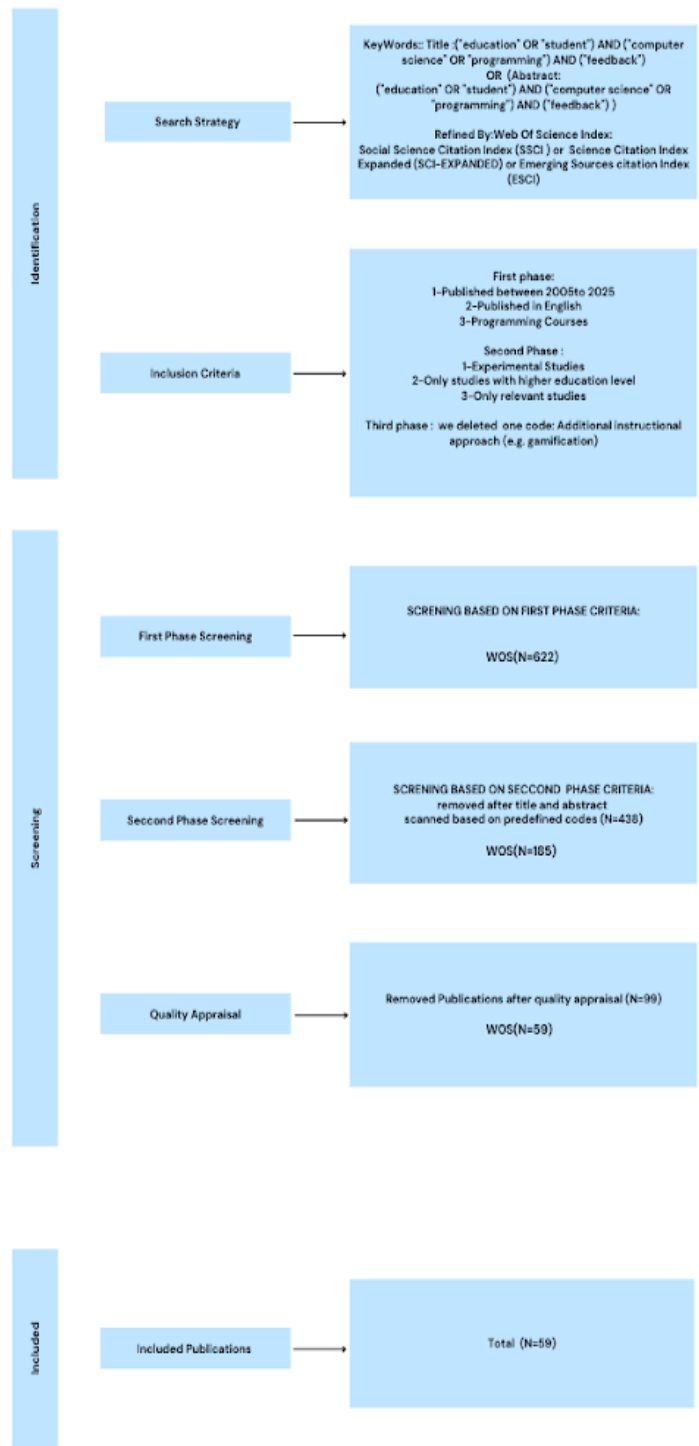


Figure 3.1 The flowchart of the screening and selection process

### 3.1.4 Data Extraction

To synthesize the findings from the included articles, a structured data extraction form was developed to systematically collect and organize relevant information from each of the 59 studies. Two researchers independently extracted the data, and any discrepancies were resolved through discussion to ensure consistency and accuracy. The following categories of information were extracted:

- **Research goals:** The primary research objectives investigated in the study.
- **Theoretical grounding:** The conceptual or theoretical frameworks underpinning the feedback design.
- **Research methodology employed:** The overall research approach (e.g., quantitative, qualitative, mixed-methods) of the study.
- **Data collected:** The specific types of data and instruments used for measurement (e.g., exam scores, assignment grades, code metrics, survey responses, interview transcripts, system interaction logs).
- **Instructor roles:** The nature and extent of the instructor's involvement in the feedback process.
- **Technologies used for feedback provision:** The specific platforms, systems, or tools used to generate and deliver feedback.
- **Feedback enhancement strategies:** Specific techniques employed to improve the feedback's effectiveness.
- **Students' use of feedback:** Evidence or measures of how students engaged with, and used the provided feedback for revision and learning.
- **Research findings:** The main results and outcomes of the study related to the impact of the feedback intervention on student learning, performance, or attitudes.
- **Research implications:** The authors' stated conclusions and their implications for educational practice, future research directions, or technological development.

## **3.2 Phase 2 Methodology (Addressing RQ2 & RQ3):**

### **3.2.1 Research Design**

The second phase employed a descriptive, mixed-methods pilot study to gather initial insights into students' perceptions and experiences of receiving AI-generated feedback through the designed platform rather than evaluating the full range of peer feedback features.

A mixed-methods approach was adopted, combining Likert-scale items with open-ended questions within a single questionnaire. This combination allows quantitative data to capture broad patterns in student perceptions while qualitative responses provide richer, contextual understanding of their experiences (Creswell & Plano Clark, 2007). Given the exploratory nature of the pilot, this integration was considered appropriate for generating complementary insights from a single data collection point.

During the pilot, students used the platform to submit their code and receive dialogue-based feedback generated by the LLM. Students were not informed whether the feedback originated from a human peer or from AI. It is important to note that this phase did not involve a comparison group or random assignment, and therefore does not constitute an experimental or quasi-experimental design. Instead, it functions as a single-group exploratory pilot intended to provide formative evidence on the platform's perceived usefulness, the quality of the feedback experience, and directions for future development. The findings from this phase are thus understood as preliminary and primarily serve to complement the conceptual contribution of the study.

### **3.2.2 Context, Participants and Sampling**

The participants were students enrolled in a Python course offered by the Department of Computer Education and Instructional Technology (CEIT) at Middle East Technical University during the fall semester of 2025. The course spanned 14 weeks and followed a flipped classroom approach. Students were expected to study course materials in advance; during class time, they worked on programming tasks while the instructor served as a guide and facilitator. Lab sessions were held two days after each class, giving students the opportunity to work on more advanced tasks and consolidate their understanding.

This course contained two sections, one taught to second-year CEIT students (n=39), another taught to first-year Statistics students (n=59). In total, 98 students participated in the study. Participants ranged in age from 19 to 25 and included both men (n=48) and women (n=50). No prior knowledge of Python was required for enrollment in the course, and although students' prior programming experience was not formally assessed, it was assumed that participants entered the course at a comparable level given the introductory nature of the curriculum.

The study used convenience sampling, in which participants are selected based on their accessibility to the researcher (Golzar et al., 2022). This approach was appropriate because the participant group was predetermined, all students enrolled in the course were invited to take part in the study as part of a regular class activity.

### **3.2.3 Data Collection Instruments and Process**

The study employed two instruments corresponding to its mixed-methods design: a Likert-scale survey and an open-ended questionnaire.

The Likert-scale survey was designed to assess students' perceptions of the feedback they received, including its perceived usefulness, clarity, and relevance to their learning. The instrument used a five-point scale with two response formats: an

agreement scale (Strongly Disagree to Strongly Agree) and a frequency scale (Never to Always). Items were adapted from previously validated instruments in the peer feedback literature (Tsui & Ng, 2000) with modifications to reflect the specific context of AI-generated feedback in programming education.

The open-ended questionnaire complemented the survey by providing space for students to describe their experiences, elaborate on their perceptions, and reflect on the feedback process in their own words. Together, the two instruments were intended to capture both broad patterns and individual nuances in students' responses.

### **3.2.4 Validity and Reliability**

Several measures were taken to support the validity and reliability of the instruments used in this study. Content validity was supported by aligning all survey items and open-ended questions with the research questions of the study, so that the instrument as a whole covered the key concepts under investigation. Face validity was examined through expert review; the study supervisor reviewed all items for clarity, relevance, and appropriateness.

In addition, the use of both quantitative and qualitative data sources enabled methodological triangulation, which strengthens the overall credibility of the findings by allowing cross-verification across different forms of evidence (Bans Akutey & Tiimub, 2021).

Reliability was assessed using Cronbach's alpha to evaluate the internal consistency of the Likert-scale items. As shown in Table 3.2, 97 of 98 cases were included in the analysis, with one excluded due to missing data. The analysis yielded a Cronbach's alpha of .873 for the 12 item scale (Table 3.3), indicating high internal consistency and confirming that the instrument was suitable for further quantitative analysis.

Table 3.2 Case processing summary for the reliability analysis

Case Status	n	%
Valid cases	97	99.0
Excluded cases	1	1.0
Total	98	100.0

Listwise deletion was applied based on all variables included in the analysis.

Table 3.3 Reliability Statistics (Cronbach's Alpha)

Number of Items	Cronbach's $\alpha$
12	.873

### 3.2.5 Data Analysis

Quantitative data from the Likert-scale survey were analyzed using SPSS. Descriptive statistics, including means, standard deviations, and frequency distributions, were calculated to summarize students' perceptions of the feedback.

Qualitative data from the open-ended questions were analyzed using thematic analysis, following the six phase framework proposed by Braun and Clarke (2006). First, all responses were read multiple times to achieve familiarization with the data. Initial codes were then generated inductively from the responses, without imposing predetermined categories, allowing themes to emerge directly from students' own language and reflections. Related codes were subsequently grouped into candidate themes, which were reviewed and refined to ensure internal coherence and clear distinctions between themes. Finally, themes were defined and named to capture the key patterns in students' experiences and perceptions of AI generated feedback.

To enhance the credibility of the analysis, coding was carried out by one researcher and independently reviewed by the study supervisor, who examined the codes and

themes for consistency and interpretive accuracy. This collaborative review process served as a form of investigator triangulation, reducing the risk of individual bias in the interpretation of qualitative data (Lincoln & Guba, 1985). Where discrepancies arose, they were discussed and resolved through consensus.

### **3.2.6 AI-Based Peer Feedback Platform**

The AI-based peer feedback platform is described in detail at the end of Phase 1 rather than within the Phase 2 research design, as the platform itself is a direct outcome of that phase. Specifically, the platform represents the operationalization of the conceptual framework and pedagogical design principles that emerged from the systematic literature review. Positioning its description within Phase 1 reflects the logic of the design-oriented research approach adopted in this study, in which theoretical analysis gives rise to a concrete educational solution. Phase 2, in turn, focuses on the empirical piloting of that solution and is concerned with how the platform was implemented, experienced, and evaluated in practice.



## CHAPTER 4

### RESULTS AND DISCUSSION

#### 4.1 Results of Systematic Literature Review (Answering RQ1)

##### 4.1.1 Analysis of Research Goals in Reviewed Studies

Research goals of the reviewed studies are summarised in Table 4.1. According to the results, the primary and most dominant research objective is to enhance student learning/engagement in programming with feedback, which was the central goal in 55.9% of the reviewed studies (Kaburlasos et al., 2008; Hulls et al., 2005). This high frequency underscores that the fundamental motivation driving research in this field is the pedagogical improvement of student outcomes, comprehension, and motivation. A secondary, yet significant goal in this category is to assess the effectiveness of automated feedback approaches and platforms, which are represented by 11 studies. This suggests a strong interest in validating the impact and efficacy of automated tools (Alshantiti & Namoun, 2019; Robinson & Carroll, 2016). Following this, the emerging field of artificial intelligence is also gaining considerable traction, with 6 studies dedicated to assessing the effectiveness of generative AI chatbots for feedback in programming (Husain, 2024; Jukiewicz, 2023). The focus on these two areas highlights a clear trend towards investigating scalable, technology-driven solutions. In contrast, other research avenues are less explored. This comparatively low frequency may suggest a shift in research interest away from these established areas toward more novel automated and AI-powered systems.

Table 4.1 Frequency and percentage of research goals

General Goal	Frequency	Percent
Enhancing student learning and engagement in programming with feedback	33	55.9%
Assessing the effectiveness of automated feedback approaches and platforms	11	18.6%
Assessing the effectiveness of generative AI chatbots for feedback in programming	6	10.2%
Assessing the effectiveness of peer feedback in programming	2	3.4%
Assessing the effectiveness of intelligent tutoring systems for feedback in programming	2	3.4%
Identify student performance patterns and predict learning outcomes	2	3.4%
Investigate how students perceive feedback tools	2	3.4%
Visualise student progress and programming behaviour to support learning	1	1.7%

#### **4.1.2 Analysis of Theoretical Grounding of Feedback Practices in Reviewed Studies**

An analysis of the selected literature on feedback in programming education reveals a significant gap in the theoretical grounding of feedback practices: the design and implementation of feedback interventions appear to be largely decoupled from established pedagogical and psychological theories of learning. Out of the 54 studies examined, a vast majority (n=45, or 83.3%) did not explicitly state a theoretical or conceptual framework that informed the design of their feedback interventions. This finding suggests that much of the research in this area is primarily practice driven, with a focus on the immediate effects of feedback rather than on building upon

established educational theories. Only nine studies (16.7%) were grounded in a specific framework. The distribution of these frameworks is presented in Table 4.2.

Table 4.2 Frequency of Theoretical and Conceptual Frameworks

Theoretical or Conceptual Framework	Number of Mentions
Cognitive Apprenticeship	2
Narciss and Huth's (2004) Conceptual Framework	2
Hattie and Timperley's (2007) Feedback Model	1
Cognitive Load Theory	1
Constructive Alignment	1
Social Translucence Framework	1
Technology Mediated Learning (TML) Framework	1
Total Studies with a Framework	9

The distribution of these frameworks, as shown in Table 1, is sparse, with Cognitive Apprenticeship and Narciss and Huth's (2004) conceptual model being the most frequent, each cited twice (Kenny & Pahl, 2005; Hao et al., 2022). Other seminal frameworks, such as Hattie and Timperley's (2007) feedback model, Cognitive Load Theory, and Constructive Alignment, were each employed in only a single study, indicating the absence of a dominant theoretical paradigm within the field (Ott et al., 2016; Wang et al., 2026).

The infrequent application of established theoretical frameworks constitutes a significant limitation within the current body of research. The lack of an underlying theoretical framework in current research constrains the generalizability of findings and hinders the systematic advancement of the field. Therefore, it is essential to

establish a more explicit and robust connection between the design of feedback interventions and educational theory. Such an integration would be instrumental in building a more cohesive, evidence-based understanding of optimal feedback practices.

### 4.1.3 Analysis of Research Methodologies Employed in Reviewed Studies

According to the analysis of research methodologies employed in the reviewed articles (see Table 4.3), inferential methods are the most prevalent, accounting for 41.67% of the studies (Heo, 2005) . Descriptive methodologies also constitute a significant portion of the research, at 23.33% (Kenny & Pahl, 2005) . Mixed methods and advanced analytical approaches are utilized in 18.33% (Pérez-Mercado et al., 2023) and the current research landscape relies heavily on numerical data, often favoring statistical results over deeper descriptive insights. While mixed-methods studies account for 15.00% of the literature, purely qualitative research is notably rare, making up only 1.67% of the total. This imbalance contributes to an atheoretical disposition in the field, which limits the generalizability of findings and hinders our ability to build a comprehensive picture of how learning occurs. To address this, it is essential to forge a more explicit and robust connection between the design of feedback interventions and established educational theories.

Table 4.3 Frequency and percentage of research methodologies

Methodology	Frequency	Percentage
Inferential	25	41.67%
Descriptive	14	23.33%

Table 4.3 (continued)

Methodology	Frequency	Percentage
Mixed Methods	10	18.33%
Advanced Analysis	9	15.00%
Qualitative Analysis	1	1.67%

These results suggest that the field of feedback research is predominantly quantitative in nature. The prominence of inferential statistics indicates that a primary research goal was to test the effectiveness of specific feedback interventions. On the other hand, the reliance on quantitative approaches and the relatively infrequent application of mixed methods (18.33%) suggests that the details of feedback interactions and the specific content of the feedback are commonly discarded. Without this deeper, qualitative layer, research may determine if an intervention works but fails to explain how or why it works. This gap is even more pronounced given the scarcity of purely qualitative methods. Thus, the low frequency of such studies indicates that research in this area does not sufficiently investigate students' learning experiences.

#### **4.1.4 Analysis of Research Data Collected in Reviewed Studies**

The analysis of data collection methods across the reviewed studies reveals the data priorities in the feedback research literature. Table 4.4 presents the results based on two distinct metrics: the total number of mentions for each data type and the percentage of unique articles that collected it. According to results, the most prevalent data collection method was system & interaction logs, utilised in 71.2% of the articles reviewed (Robinson & Carroll, 2019). This was followed closely by the collection of code & submissions (Razali et al., 2021) (64.4%) and grades & scores

(Alshantiti & Namoun, 2019) (62.7%). These figures demonstrate that a majority of studies in this domain rely on tracking student behaviour and performance. Interestingly, when looking at the total volume of mentions, grades & scores (n=60) emerged as the most frequently mentioned category, surpassing system & interaction logs (n=46) and code & submissions (n=46). This suggests that while logging is the most widespread practice, performance metrics are the most discussed data outcome.

Data from surveys & questionnaires were present in 30.5% of articles (Lajis & Baharudin, 2019). Other methods, such as collecting feedback-related data (27.1%) (Kenny & Pahl, 2005), qualitative data (25.4%) , and error & bug data (25.4%)(Kuo et al., 2022) , were each found in about a quarter of the studies. Finally, demographic & background data was the least common category by both measures, appearing in only 6.8% of articles(Azcona et al., 2019), suggesting that student background is not a primary focus in the data collection strategies of the literature analysed.

Table 4.4 Frequency and percentage of data collection approaches

Data Collection	Total Mentions	Frequency (No. of Articles)	Percentage of Articles (%)
Grades & Scores	60	37	62.70%
System & Interaction Logs	46	42	71.20%
Code & Submissions	46	38	64.40%
Surveys & Questionnaires	21	18	30.50%
Feedback-Related Data	18	16	27.10%

Table 4.4 (continued)

Data Collection	Total Mentions	Frequency (No. of Articles)	Percentage of Articles (%)
Qualitative Data	17	15	25.40%
Error & Bug Data	15	15	25.40%
Demographic & Background Data	4	4	6.80%

#### 4.1.5 Analysis of Instructor Roles in Feedback Process in Reviewed Studies

To understand the roles instructors assume when implementing technology enhanced feedback, an analysis of instructors' roles in the reviewed research was performed. According to Table 1, the high frequency of studies where the instructor's role is "not specified" (27.1%) is considered a significant concern (Liénardy et al., 2021). It reinforces the idea that a substantial portion of the literature overlooks or minimises the instructor's role, focusing instead on the technology itself and thereby failing to capture the pedagogical importance of its implementation. Nonetheless, the analysis yielded several key instructional functions. The most frequently identified roles are task design (24.7%) and feedback design (21.2%) (Gálvez et al., 2009; Hulls et al., 2005). This finding suggests that the instructor's primary function is to design the initial learning tasks and the associated feedback processes that the technology will facilitate. However, in many of the analysed articles, their role in feedback design was limited to creating the rubric and providing the desired solution for automated comparison with students' work (Gálvez et al., 2009). This approach positions the instructor as a provider of static evaluative criteria rather than a dynamic facilitator of dialogic, iterative feedback.

Furthermore, the analysis points to a lack of follow-up instructor engagement that complements the feedback activity. The low frequencies for monitoring students (11.8%) and providing supplemental guidance (8.2%) suggest that instructors often play a passive role once the technology is deployed. This suggests a potential over-reliance on the automated capabilities of the feedback systems, accompanied by a corresponding lack of complementary teaching that is crucial for meaningful learning. This finding is further supported by the minimal instructor involvement in platform configuration (7.1%), indicating that instructors rarely adjust the system's feedback behaviour or customize the technology to align with emerging student needs.

Table 4.5 Code frequencies about the instructors' role

Code	Frequency	Percentage
Not specified	23	27.06%
Task design	21	24.71%
Feedback design	18	21.18%
Monitoring students	10	11.76%
Supplemental guidance	7	8.24%
Platform configuration	6	7.06%

#### **4.1.6 Analysis of Technologies Used for Feedback Provision in Reviewed Studies**

Table 4.6 presents the frequency and corresponding percentage of each feedback technology identified in the reviewed literature. The results show that automated feedback delivered via a custom platform is the most prevalent method, representing

a clear majority at 58.49% of all cases (Skalka & Drlík, 2023). The second most common category is feedback from AI-powered chatbots (ChatGPT), which accounts for 11.32% of the instances(Haindl & Weinberger, 2024) . This is followed by automated feedback via online judges, making up 9.43% of the total (Wang et al., 2023) . Other forms of feedback were significantly less common. Teacher, peer, and automated feedback delivered through a Learning Management System (LMS) each constituted 5.66% of the observed methods(Zampirolli et al., 2021). The least frequent types were peer feedback within a custom platform and semi-automated feedback, each at 1.89% (Ott et al., 2016). Among all studies, only 5 of them leveraged a hybrid approach, combining the strengths of human and technology-generated feedback (Hämäläinen et al., 2011; Azcona et al., 2019).

Table 4.6 Frequency and Percentage of Feedback Delivery Types

Feedback Type	Frequency	Percentage
Automated feedback via a custom platform	31	58.49%
ChatBOT (Chat GPT)	6	11.32%
Automated feedback via online judge	5	9.43%
Automated feedback in LMS	3	5.66%
Teacher feedback in a custom platform	3	5.66%
Peer feedback in a custom platform	3	5.66%
Peer feedback in LMS	1	1.89%
Semi-automated feedback via a custom platform	1	1.89%

The findings reveal the prevalence of technologically mediated solutions that prioritise scalability and immediate delivery. This suggests a research focus on developing specialised, standalone tools that can operate without direct human input. However, this strong emphasis on full automation stands in contrast to the integration of human expertise with technology. A critical observation is that only five of the documented approaches considered hybrid models, where human feedback was complemented by automatically generated feedback. This scarcity highlights a significant gap in the literature, pointing towards a preference for replacing human interaction rather than augmenting it. While the pursuit of efficient, automated feedback is valuable, the underexploration of hybrid models suggests a missed opportunity to leverage technology to enhance the motivational and dialogic elements of human-led feedback.

#### **4.1.7 Analysis of Feedback Enhancement Strategies in Reviewed Studies**

To understand the extent to which research studies incorporated specific strategies to enhance the student feedback experience, a specific analysis was conducted. According to the results (see Table 4.7), the majority of studies did not implement an additional pedagogical approach beyond providing and generating accurate feedback on the correctness of the work. That is, out of 59 analysed studies, an overwhelming 76.27% (n=45) did not employ any specific measure to improve the feedback experience other than aiming for accurate feedback (Haldeman et al., 2021). The most common strategy that was implemented was personalised feedback (based on students' knowledge levels), though it was present in only 15.25% (n=9) of the studies (Azcona et al., 2019). Other approaches were exceptionally rare. This distribution suggests that the primary focus of the reviewed literature remains on the functional aspect of feedback accuracy rather than on the experiential and motivational dimensions of its delivery.

Table 4.7 Frequency and Percentage of Feedback Enhancement Strategies

Feedback Approach	Frequency (n)	Percentage (%)
None	45	76.27%
Personalized feedback based on knowledge level	9	15.25%
Gamification	2	3.39%
A virtual character in video format	1	1.69%
Explainable AI to make feedback interpretable	1	1.69%
Trustworthy feedback through control flow automata	1	1.69%

The emergence of personalised feedback as the most common enhancement strategy indicates an awareness of the limitations of a one size fits all model. Nonetheless, these findings suggest that the primary focus in the field has been on the algorithmic accuracy and technical implementation of feedback systems rather than on the human-computer interaction and pedagogical aspects of how students receive and process that information. This correctness-first paradigm, while foundational, treats the learner as a passive recipient of information and overlooks critical factors that influence feedback uptake, such as student motivation, self-efficacy, and cognitive load.

#### **4.1.8 Analysis of Students' use of Feedback for Revision in Reviewed Studies**

The analysis of how students uptake feedback in the existing literature revealed five primary ways in which student engagement with feedback is reported. The most prominent usage identified was direct revision & correction, accounting for 42.4% of the instances (Carbonaro, 2019). This suggests that the literature most frequently captures the instrumental use of feedback, where students apply comments directly to improve their work, correct errors, and resubmit assignments. Interestingly, a combined 40.6% of the manuscripts reported either no student uptake (20.3%) or that the method of uptake was unspecified (20.3%) (Ahmed et al., 2021; Estévez-Ayres et al., 2024; Gupta & Mehrotra, 2022). This is a significant finding, indicating that in a substantial portion of the reviewed literature, either the feedback intervention was ineffective at prompting student action, or the specific nature of the student's engagement was not documented.

Moreover, the theme of information seeking and understanding represented 13.6% of the cases. This category includes more cognitively engaged behaviours, where students use feedback not just for immediate correction but to deepen their comprehension, for instance, by using hints or engaging in self-assessment (Kaburlasos et al., 2008). The relatively low percentage of this category suggests that many studies may not be designed to capture these deeper forms of engagement. It also suggests that if the goal of feedback is long-term learning and conceptual growth, then strategies that explicitly encourage information seeking and self-assessment may be underutilised in the current literature. Finally, peer-mediated uptake was the least common theme, appearing in only 3.4% of the manuscripts (Grey & Gordon, 2023). This finding is particularly notable given the growing emphasis on collaborative learning and peer-to-peer feedback in modern educational settings.

Table 4.8. Frequency and Percentage of Students' Feedback Use

Feedback Use	Frequency	Percentage	Description
Direct Revision & Correction	25	42.40%	Students made direct changes to their work, such as correcting code, revising text, or resubmitting assignments.
No Student Uptake	12	20.30%	The manuscript explicitly stated that students did not use or apply the feedback provided.
Uptake Unspecified	12	20.30%	The manuscript mentioned that students used the feedback but did not specify how they used it.
Information Seeking & Understanding	8	13.60%	Students used the feedback to deepen their understanding, such as using hints, engaging in discussions, or performing self-assessments.
Peer-Mediated Uptake	2	3.40%	Students' use of feedback was mediated through interaction with their peers, such as in peer review activities.

#### **4.1.9 Analysis of Research Findings in Reviewed Studies**

The analysis of the findings from the reviewed literature yielded four primary themes. The frequency distribution of these findings, based on the aggregated count of supporting articles, is presented in Table 4.9. The theme Enhancement of Student Performance & Cognitive Skill Development emerged as the most frequent, with a total of 19 articles. This underscores that the principal goal and most frequently demonstrated result of these systems is a direct improvement in learning. Within this theme, Improved General Academic Performance was the single most prevalent outcome, cited in 11 articles (Fu et al., 2023) . This was followed by the Development of Specific Technical & Programming Skills and Enhanced Understanding & Knowledge Retention, each supported by 4 articles (Ouyang et al., 2024; Carbonaro, 2019) . These results suggest that researchers are successfully validating the broad, tangible benefits of these technologies on core educational objectives.

The second most prominent theme was Efficacy & Attributes of Feedback Systems & Technologies, with an aggregated count of 16 articles. This indicates that the field has matured beyond simply asking if these systems work to investigating how and why they succeed. The Importance of System Design & Features was the most common finding in this category (7 articles), followed closely by The Role of Specific Technologies & Approaches (6 articles) (Razali et al., 2021; Robinson & Carroll, 2016). The High Accuracy & Performance of Automated Systems was also a key finding, supported by 3 articles (Wang et al., 2020). The significant attention paid to system design and specific technologies confirms that the tool's quality and functionality are considered critical to its effectiveness.

The affective and motivational impact on students was also a major focus, as captured by the theme Cultivation of Student Engagement & Positive Affective States, which accounted for 14 articles, highlighting the crucial role of the student experience. Increased Student Engagement was the leading outcome in this category, with 7 articles (Nutbrown & Higgins, 2016) . Enhanced Motivation, Confidence, & Self-Efficacy was also frequently reported (5 articles), along with Improved

Learning Experience & Enjoyment (2 articles) (Fu et al., 2023; Ouyang et al., 2023). By increasing engagement and enhancing motivation, these systems foster a more positive and productive learning environment, which in turn likely contributes to the performance gains seen in the primary theme.

Finally, the theme of Operational Efficiencies & Pedagogical Support for Educators was supported by 10 articles, indicating a focus on the practical benefits for instructors and institutions. Reduced Instructor Workload & Cognitive Load was the most cited benefit in this area (5 articles) (Liénardy et al., 2021). Other important outcomes included Enhanced Scalability & Personalisation of Instruction (3 articles) and the emerging area of Diagnostic & Predictive Capabilities (2 articles) (Ou Yang et al., 2023; Wang et al., 2023). These findings demonstrate that these tools not only save time but also enable more sophisticated and targeted teaching strategies.

Table 4.9. Frequency of Themes and Specific Findings

<b>Theme</b>	<b>Findings</b>	<b>Article Count</b>	<b>Theme Total</b>
1. Enhancement of student performance & cognitive skill development	Improved General Academic Performance	11	19
	Development of Specific Technical & Programming Skills	4	
	Enhanced Understanding & Knowledge Retention	4	
2. Cultivation of Student Engagement & Positive Affective States	Increased Student Engagement	7	14
	Enhanced Motivation, Confidence, & Self-Efficacy	5	
	Improved Learning Experience & Enjoyment	2	

Table 4.9 (continued)

<b>Theme</b>	<b>Findings</b>	<b>Article Count</b>	<b>Theme Total</b>
3. Operational Efficiencies & Pedagogical Support for Educators	Reduced Instructor Workload & Cognitive Load	5	10
	Enhanced Scalability & Personalisation of Instruction	3	
4. Efficacy & Attributes of Feedback Systems & Technologies	Diagnostic & Predictive Capabilities	2	16
	Importance of System Design & Features	7	
	The Role of Specific Technologies & Approaches	6	
	High Accuracy & Performance of Automated Systems	3	

#### **4.1.10 Analysis of Research Implications in Reviewed Studies**

The analysis of feedback research implications yielded 27 distinct items, which were categorised into 6 overarching themes. According to results (see Table 4.10), the most prominent theme was Feedback Content & Quality, accounting for nearly a third of all coded implications (33.33%), followed by Feedback Timing & Frequency (19.61%), and Human-in-the-Loop & Agency (17.65%). The themes of Personalisation & Adaptivity and Feedback Design & Delivery were also significant, each representing 13.73% of the codes.

Table 4.10 Frequency of Themes and Specific Implications

<b>Theme</b>	<b>Frequency</b>	<b>Perc.</b>	<b>Code</b>	<b>Frequency</b>
Feedback Content & Quality	18	35.28	Instructive feedback (explaining concepts)	4
			Specific Feedback	2
			Clear Feedback	2
			Explanatory Feedback	2
			Comprehensive Feedback	1
			Accurate and focused Feedback	1
			Actionable feedback	1
			Guiding Feedback (without direct answers)	1
			Concise feedback	1
			Feedback with concrete Examples	1
Feedback Timing & Frequency	10	19.6	Multiple rounds of feedback & attempts	6
			Immediate/real-time feedback	3
			Delayed feedback	1
Human in the Loop & Agency	9	17.64	Human-AI collaboration	3
			Teacher verification of automated feedback	3

Table 4.10 (continued)

<b>Theme</b>	<b>Frequency</b>	<b>Perc.</b>	<b>Code</b>	<b>Frequency</b>
Human in the Loop & Agency	9	17.64	Self-Assessment	2
			Student Evaluation of Feedback	1
Personalization & Adaptivity	7	13.72	Multi-levelled hint-based feedback	3
			Personalised and adaptive Feedback	3
			Feedback based on performance History	1
Feedback Design & Delivery	7	13.72	Consistent and Fair Feedback	2
			Providing examples for feedback calibration	2
			Rich data for automated feedback generation.	1
			Complex task design	1
			Encouraging tone	1

Within the first theme, feedback content & quality, instructive feedback (7.84%) was the most frequently identified code, suggesting a strong emphasis on feedback that actively teaches and provides guidance (Razali et al., 2021). Other notable codes included specific feedback, clear feedback, and explanatory feedback (each 3.92%), highlighting the need for feedback that is precise, easy to understand, and provides rationale (Fu et al., 2023; Gordillo, 2019; Razali et al., 2021). The most dominant code in the second theme was the feedback timing & frequency, multiple rounds (or iterations) of feedback and student attempts (11.76%). This finding strongly indicates that iterative processes, allowing learners to apply feedback and retry tasks,

are considered a critical component of effective learning design. This was followed by immediate/real-time Feedback (5.88%), which also emphasises the temporal dimension of feedback delivery (Zhong & Zhan, 2024).

Moreover, within the third theme, human in the loop and agency, capturing the roles of humans (both educators and learners) in the feedback process, the codes for human-AI collaboration and teacher verification of feedback were equally prominent (each 5.88%), signalling the importance of maintaining a human element, particularly for oversight and partnership, in automated feedback systems (Jukiewicz, 2024; Ouyang et al., 2024). Furthermore, learner-centric codes such as self-assessment (3.92%) and student evaluation of feedback (1.96%) were identified (Haindl & Weinberger, 2024; Zhong & Zhan, 2024).

Regarding the personalisation & adaptivity theme, the codes' multi-level hint-based feedback and personalised feedback were the most frequent (each 5.88%)(Li & Hsieh, 2019). This suggests a strong trend towards moving away from one-size-fits-all feedback and towards systems that can adjust the level of detail, scaffolding, or content based on the learner's specific needs and performance history. Lastly, the feedback design and delivery theme encompasses the principles and structural elements that create the foundation for an effective feedback system. The most frequent codes within this theme were consistent/fair feedback and providing examples and calibration training (each 3.92%) (Gupta & Mehrotra, 2022; Hämäläinen et al., 2011). This pairing highlights the importance of establishing a reliable and equitable feedback environment where learners understand the evaluation standards through clear examples and structured exercises.

#### **4.1.11 Synthesis of the Literature Review**

The analysis of the reviewed literature reveals a field focused on validating the effectiveness of automated feedback systems in programming education, but this endeavor is frequently pursued in a theoretical vacuum. The research is

predominantly quantitative, using data from system logs and grades to measure performance. While the primary goal is to enhance student learning through scalable, technology-driven solutions, a significant gap exists between the design of these interventions and established educational theories. An overwhelming majority of studies did not cite a theoretical framework, suggesting a practice driven approach that prioritizes immediate technical effects over a systematic advancement of knowledge.

However, this focus on automation and performance metrics has created several significant gaps. First, a critical finding is the overwhelming emphasis on feedback accuracy at the expense of the student's learning experience. Most studies employed no specific pedagogical strategy to make feedback more engaging or motivational. Without grounding in pedagogical frameworks, most studies defaulted to a correctness-first paradigm. This paradigm treats the learner as a passive recipient of information rather than an active participant in their own learning. The research successfully demonstrates whether a tool can identify errors, but it largely fails to investigate how or why students might engage with that information to achieve a deeper understanding. Similarly, the literature largely neglects the qualitative, motivational, and experiential dimensions of learning.

Furthermore, the literature shows a clear preference for replacing human interaction rather than augmenting it. The prevalence of fully automated, custom-built platforms contrasts sharply with the scarcity of hybrid models that combine technological efficiency with human pedagogical expertise. This trend is mirrored in the instructor's role, which is frequently passive and limited to upfront design. Instructors are rarely involved in monitoring student progress or providing supplemental guidance after the technology is deployed, highlighting a missed opportunity to blend automated feedback with crucial, just in time human support. These findings are again considered a consequence of lack of grounding theoretical frameworks, which inherently value expert human guidance. The general neglect of such theories has led to a technology-centric, human-scarce approach where the

instructor's role is often limited to upfront design, missing crucial opportunities for just-in-time human support.

The consequences of this technology-centric, human-scarce approach are evident in the uptake of student feedback. A substantial portion of the literature reported that students either did not use the feedback at all or that their engagement was not documented. When students did act on the feedback, their engagement was typically shallow, focusing on direct revision and correction rather than the more cognitively demanding process of seeking and understanding information.

In essence, the literature review reveals a field that has successfully developed scalable technologies for providing accurate feedback but has yet to fully address the pedagogical complexities of how students learn. This is a direct consequence of the disconnection from established educational theory. The current research trajectory prioritises automation over augmentation and correctness over experience, ultimately limiting the potential for a deeper and more lasting impact on student learning. We argue that establishing a robust connection between practice and theory is an imperative next step for building a more cohesive, evidence-based understanding of optimal feedback practices in programming education.

#### **4.1.12 A Conceptual Framework For AI-Augmented Peer Feedback in Programming Education**

Building on the findings of the reviewed literature, this study proposes a conceptual framework for designing AI-powered peer feedback specifically for programming education. As the literature review demonstrated, majority of the reviewed studies lacked an explicit theoretical or conceptual framework (see Table 4.1). To address this gap directly, the present framework is built on a strong educational foundation, integrating Hattie and Timperley's (2007) feedback model and Shute's (2008) principles of formative feedback. These two theoretical lenses were selected because they complement each other: Hattie and Timperley's model provides a macro-level

structure for the feedback process (Feed-Up, Feedback, and Feed-Forward) and its content (task, process, and self-regulation), while Shute’s principles offer micro level guidance on the characteristics of the feedback itself, such as timeliness, specificity, and tone. This dual-lens approach ensures that the proposed framework is pedagogically sound, directly responding to the literature review’s call for a more explicit and robust connection between the design of feedback interventions and educational theory.

#### **4.1.12.1 Hattie and Timperley Model**

The Hattie and Timperley model posits that an effective feedback process must orient the learner by answering three fundamental questions, which correspond to Feed-Up, Feedback, and Feed-Forward. This structure moves learning beyond a simple, reactive correction of errors and reframes it as a proactive, cyclical journey toward a clear goal.

Where am I going? (Feed-Up): This question addresses the clarity of the learning goals and success criteria. Learners cannot effectively reduce a discrepancy if they do not understand the target. In programming education, this involves a clear problem specification, explicit requirements, and well-defined standards for code quality and functionality.

How am I going? (Feedback): This refers to information about the learner’s current performance in relation to the goal. It answers the question of progress. In the context of programming, this can range from “Your code does not compile” to “Your algorithm is inefficient for large datasets.”

Where to next? (Feed-Forward): This question is very crucial for promoting learning and self-regulation as it focuses on the next steps the learner should take to close the gap between their current state and the goal. Feed-forward information is inherently actionable since, instead of merely pointing out an error, it suggests a path toward improvement.

In the proposed framework, these three questions serve as organizing logic for the entire cycle: Phase 1 (goal setting) operationalizes feed-up, Phase 2 and 3 (submission and peer review) operationalize feedback, and Phase 4 (synthesis and reflection) operationalizes feed-forward.

Additionally, Hattie and Timperley's model distinguishes between four levels at which feedback can be directed. First, feedback about the task (task level) concerns how well a task is being performed or understood, often involving judgments of correctness or incorrectness. While essential, task-level feedback often fails to explain why an answer is incorrect, provides no information on how to improve, and can lead to cognitive overload if a student is presented with a long list of errors in programming. Second, feedback about the process (process level) focuses on the underlying processes or strategies used to complete the task. For programming, this is a more powerful and pedagogically valuable form of feedback as it can address the student's algorithmic thinking, their choice of data structures, their approach to decomposing the problem, and their debugging strategies. Third, feedback about self-regulation (self-regulation level) is aimed at developing the student's metacognitive skills (i.e., ability to monitor, direct, and regulate their own learning). It encourages students to become more autonomous and to develop skills in self-assessment and error detection. Examples in a programming context might include prompts to ask students about the steps they took to test or improve their code. Lastly, feedback about the self (self level) involves personal evaluations or praise directed at the student as a person, such as "You are a smart programmer" or "Well done!". Research consistently shows this to be the least effective and often detrimental form of feedback.

Shute's (2008) broadly describes formative feedback as information given to learners to shape their thoughts and actions, ultimately facilitating better learning. This definition aligns perfectly with the goal of a programming education platform. This study considers several of the feedback principles suggested by Shute (2008) that are more relevant for generating effective formative feedback in programming. The feedback should be non-evaluative and supportive to support learning, not to

pass judgment. Additionally, the tone of feedback should be constructive and encouraging. Second, feedback should be timely as it is most potent when it can be acted upon promptly. Research shows that immediate feedback is particularly beneficial for procedural skills like programming and for preventing the reinforcement of errors (Marwan et al., 2022). Third, feedback must be specific, pointing to particular aspects of the code, and elaborated, explaining why something is correct or incorrect. Finally, although feedback should be informative, it must remain manageable. Providing too much information at once may be counterproductive, as lengthy or overly complex feedback may not be processed by learners and can lead to cognitive overload ( e.g., Narciss & Huth, 2004; Shute, 2008)

#### **4.1.12.2 The Proposed Framework: The Multi-Layered AI-Augmented Peer Feedback (MLA-PF) Cycle**

Building upon the established theoretical foundations and the findings of the literature review, this report proposes the Multi-Layered AI-Augmented Peer Feedback (MLA-PF) Cycle. The framework aims to guide the integration of AI into the peer feedback processes to enhance the experience for all human participants: authors (students submitting work), reviewers (students evaluating submissions), and instructors. It specifically aims to enhance authors' ability to understand and incorporate feedback, reviewers' ability to provide high-quality feedback, and teachers' ability to oversee the entire process and intervene as necessary.

This framework is a comprehensive, five-phase cyclical model designed to structure the entire feedback process in a way that maximizes learning, promotes self regulation, and leverages the unique strengths of both human peers and artificial intelligence. Its design is a direct response to the critical gaps identified in the literature review: the dominance of fully automated systems over hybrid models ( Table 4.6), the passive nature of instructor roles ( Table 4.5), the correctness-first paradigm that overlooks pedagogical quality (Table 4.7), and the shallow engagement of students with the feedback they receive (Table 4.8). Each phase of

the framework is grounded in the theoretical foundations described above and informed by the empirical findings of the reviewed literature.

#### Phase 1: Goal Setting (Feed-up)

The cycle begins by establishing the success criteria for the assignment before any code is written, ensuring that all subsequent feedback is anchored to a shared standard. This phase directly operationalizes Hattie and Timperley's (2007) Feed-Up question ("Where am I going?"), which posits that learners cannot effectively reduce a discrepancy if they do not understand the target.

The instructor initiates the process by defining the high-level parameters of the programming assignment. This includes the problem statement, the core learning objectives (e.g., "to demonstrate understanding of object-oriented inheritance"), and the key dimensions for assessment (e.g., correctness, code style, efficiency, documentation). The instructor also creates a rubric that provides the necessary structure and criteria to ensure the feedback is relevant and comprehensive. The instructor acts as the pedagogical architect, setting the overall direction for the learning activity. This active, upfront involvement directly addresses the literature review's finding that instructors' most frequent roles are limited to task design and feedback design, while monitoring and supplemental guidance are notably less common. By foregrounding the instructor's role as a deliberate designer of the feedback ecosystem, not merely a rubric provider, the framework elevates the instructor from a passive role to an active pedagogical architect.

#### Phase 2: Submission and Automated First-Pass Analysis (Feedback Layer 1)

Once the goals are set, the student author begins working on the assignment. The framework is designed to intervene while the work is still in progress, aligning with Shute's (2008) principle that formative feedback is most effective before a task is finalized and the literature review's emphasis on the importance of timely feedback. The student author submits their initial or in-progress code to the platform. This is not a final submission for a grade, but rather a submission for feedback. Upon

submission, the platform's AI code analyst immediately performs a comprehensive, private analysis of the code. This first layer of feedback is purely technical and encompasses objective metrics, including correctness, code complexity, code duplication, and potential bugs.

A crucial design principle of the MLA-PF framework is that the results of this initial AI analysis are not immediately revealed to the author. Instead, this data is stored by the system as a baseline measurement of the submission's quality. The reason for withholding this assessment information is to mitigate the over-reliance on AI. Providing immediate and direct AI corrections at the start may lead students to engage in a superficial process of fixing the reported issues without necessarily understanding the deeper logical or structural problems in their code, a pattern consistent with the literature review's finding that the dominant mode of feedback uptake is direct revision and correction (42.4%, see Table 4.8), while deeper cognitive engagement such as information seeking and understanding is comparatively rare (13.6%). By holding back this information, the framework prioritizes the richer, more conceptual feedback that will come from human peers in the next phase, deliberately countering the correctness-first paradigm identified in the literature review (see Table 4.7). This design choice also reflects Hattie and Timperley's distinction between task-level feedback (which the AI provides privately) and the higher-order process and self-regulation levels that the framework seeks to promote through peer interaction.

### Phase 3: AI-Coached Peer Review (Feedback Layer 2)

This phase constitutes the core human-to-human interaction within the cycle and provides the second layer of feedback. It is designed to overcome the common pitfalls in peer reviews by integrating AI as a real-time, adaptive pedagogical coach. The author's submission is anonymized to reduce social bias and is then distributed to their peers for review. This phase directly addresses the literature review's finding that feedback types involving human participation, such as teacher and peer feedback, were notably less common than fully automated approaches (see Table

4.6), and that the field has largely focused on replacing human interaction rather than augmenting it with technology.

In this process, AI acts as a pedagogical scaffolding agent for the reviewer, operationalizing Shute's (2008) principles of specificity, elaboration, and constructive tone at the moment of feedback production. As the reviewer provides constructive feedback guided by the instructor's rubric, the AI provides suggestive comments designed to improve the quality of their feedback. A key enhancement in this phase is that the AI's coaching is adaptive. It leverages its private analysis from Phase 2 to tailor its suggestions to the reviewer based on the specific needs of the student author. This adaptive mechanism directly responds to the literature review's finding that personalized feedback based on knowledge level was the most common (yet still rare at 15.25%) feedback enhancement strategy (see Table 4.7), and the implication theme of Personalization & Adaptivity (see Table 4.10).

For an author whose code indicates a lower level of understanding (e.g., frequent syntax errors, fundamental logic flaws), the AI should prompt the reviewer to be more explicit and foundational, guiding them toward Hattie and Timperley's task and process levels. For a reviewer comment like "This function is wrong," the AI might prompt: "The automated analysis suggests the author may be struggling with for-loops. Can you break down why this logic is incorrect and provide a simple example of a correct structure?" For an author whose code is technically correct but inefficient or poorly styled, the AI will guide the reviewer toward more conceptual, process-level feedback. For a comment like "It works," the AI might prompt: "The code is correct, but the automated analysis flagged high complexity. Can you ask the author to consider the efficiency of their algorithm for larger datasets?"

This adaptive intervention serves as a form of just-in-time, differentiated training, teaching students how to give more effective and appropriately leveled feedback while they are in the process of doing it. This directly responds to the literature review's implication that providing examples and calibration training are important components of effective feedback design (see Table 4.10, Feedback Design &

Delivery theme), and addresses the finding that most studies did not implement measures to improve the feedback experience beyond aiming for accuracy (see Table 8).

#### Phase 4: Synthesized Feedback and Guided Reflection (Feed-Forward)

After the peer review period ends, this phase delivers the feedback in a manageable and actionable way, explicitly addressing Hattie and Timperley's Feed-Forward question ("Where to next?") and fostering self-regulation. This phase is grounded in both Shute's (2008) principle that feedback must remain manageable to avoid cognitive overload, and Hattie and Timperley's (2007) self-regulation level of feedback.

First, the AI's role goes beyond simply presenting a raw list of comments from different reviewers, as this can be overwhelming and contradictory. The AI analyzes the content of all peer reviews to identify common themes and points of consensus. It then presents a high-level, synthesized summary to the student author (e.g., "Two reviewers noted that your variable names could be more descriptive. One reviewer suggested your approach might be inefficient for large inputs, while another found your logic easy to follow."). This synthesis is intended to reduce the cognitive load of processing multiple streams of information, and thus help the student to immediately grasp the most critical feedback. This design directly addresses the literature review's finding that feedback uptake is frequently shallow or unspecified (Table 4.8, where a combined 40.6% of studies reported either no student uptake or unspecified uptake), by structuring feedback delivery in a way that facilitates deeper engagement.

Second, a key enhancement in this phase is the introduction of an AI-guided dialogic interaction between the author and their reviewers. The author can ask clarifying questions about the feedback they received. When a reviewer responds, the AI again draws on its initial analysis from Phase 2 to guide the interaction productively. For example, if the author asks, "What do you mean my code isn't efficient?" and the initial analysis indicated a poor understanding of algorithms, the AI will guide the

reviewing peer's response. It might prompt the reviewer by saying: "The author seems to need a more basic explanation. Try using a simple analogy or providing a side-by-side code snippet to illustrate a more efficient approach." This guided dialogue ensures that the conversation is constructive and tailored to the author's level, increasing their understanding and the likelihood of feedback uptake. The dialogic nature of this interaction also addresses the literature review's finding that peer-mediated uptake was the least common form of feedback use (3.4%, Table 4.8), by providing a structured mechanism for productive peer dialogue.

Finally, following the dialogue, the AI provides guided reflection to help the student author figure out what to do. In doing so, it actively prompts them to engage in a metacognitive reflection process, directly targeting Hattie and Timperley's (2007) self-regulation level. The system poses targeted questions based on the feedback received, such as: "Based on the dialogue and your peers' comments, what are the one or two most important changes you plan to make?" This step is crucial as it forces the student to process the external feedback, generate their own internal feedback, and formulate an explicit plan of action (Feed-Forward). This design responds to the literature review's implication that self-assessment is an important but underutilized element of the feedback process (see Table 4.10, Human in the Loop & Agency theme).

#### Phase 5: Revision, Resubmission, and Progress Analytics

The final phase of the cycle closes the loop by focusing on action and making learning visible, thus completing the Feed-Up, Feedback, Feed-Forward cycle described by Hattie and Timperley (2007). To begin with, guided by the synthesized feedback and their own reflection plan, the student author revises their code. They then resubmit the improved version to the platform. Then, the AI re-runs the same technical analysis from Phase 2 on the revised submission and compares the new results to the stored baseline. This comparative analysis allows the platform to generate concrete, data-driven feedback on the author's progress. Accordingly, the platform can provide an encouraging summary of improvements, such as: "Great

work on your revision! You have successfully passed 8 new test cases and resolved 12 style guide issues identified in the previous version.” This reinforces the value of the feedback and revision process and directly operationalizes the literature review’s most frequently cited implication: the importance of multiple rounds (iterations) of feedback and student attempts (see Table 4.10).

The platform provides instructors with an enhanced dashboard that transforms their role from passive observers to active guides. This dashboard gives them a high-level overview of class progress and diagnostic capabilities. It uses AI to pinpoint common errors, identify students who give or receive excellent feedback, and flag learners who are struggling. This allows instructors to offer targeted support to those who need it most. This design directly addresses the literature review’s finding that in 27.1% of the reviewed studies the instructor’s role was not specified, and that monitoring students (11.8%) and providing supplemental guidance (8.2%) were notably infrequent functions (see Table 4.5). By providing actionable analytics, the framework promotes a shift from the passive instructor role documented in the literature to an active, informed facilitator.

The dashboard also helps instructors to verify and manage the quality of feedback. They can quickly spot-check peer comments, approve high-quality feedback with a single click, or step in to add their own insights when necessary. This maintains a crucial human element and control in the feedback process. To further support instructor autonomy, the platform allows educators to easily adjust feedback rubrics, modify the AI’s sensitivity, or customize prompts based on the needs they observe from the dashboard’s analytics. This addresses the literature review’s finding that platform configuration by instructors was very rare (7.1%, see Table 4.5) and responds to the implication that human-AI collaboration and teacher verification of feedback are important components of effective systems (see Table 4.10). This data gives instructors a clear, rich picture of the class’s improvement, moving beyond simply looking at the final quality of work to understanding the progress of each student.

Table 4.11.Roles and Responsibilities within the MLA-PF Cycle

Phase	Student-Author	Student-Reviewer	AI Agent	Instructor
1. Goal Setting (Feed-Up)	Reviews and understands assignment goals and rubric.	-	-	Defines learning objectives and creates the rubric.
2. Submission & Analysis	Submits code-in-progress for feedback.	-	Performs private, baseline analysis of code (correctness, style, quality) and stores the results.	Monitors submission rates and overall class progress.
3. Peer Review (Feedback)	Awaits peer feedback.	Reviews anonymous peer code, provides feedback guided by the rubric.	Provides real-time, non-evaluative prompts to the Student-Reviewer to improve feedback quality, focusing on specificity, constructiveness, and alignment with process-level criteria.	Observes the quality of peer feedback being generated via the dashboard; intervenes if necessary.
4. Synthesis & Reflection (Feed-Forward)	Reviews synthesised feedback, engages with reflection prompts, and creates a revision plan.	-	Synthesises comments from multiple reviewers, presents a summary to inform the dialogue between author and reviewer students. Poses guided reflection questions to enhance the dialogue. Selectively reveals initial AI analysis to confirm peer findings.	Analyses class-wide reflection plans to identify common points of confusion.

Table 4.11 (continued)

Phase	Student-Author	Student-Reviewer	AI Agent	Instructor
5. Revision & Analytics	Revises and resubmits code based on feedback and reflection. Reviews personal progress analytics.	-	Re-analyses the revised code, compares it to the baseline, and generates progress analytics for both the student and the instructor dashboard.	Review progress analytics on the dashboard to identify students who are improving and those who may need direct intervention.

This completes one iteration of the AI-powered peer feedback cycle. For complex assignments, the cycle could be repeated multiple times, allowing for continuous refinement and deeper learning.

#### 4.1.13 Alignment of the Proposed Framework with Research Gaps

The MLA-PF framework was intentionally designed as a direct and purposeful response to the critical gaps, challenges, and implications identified in the comprehensive literature review. The analysis of existing research revealed a field heavily focused on the technical accuracy of fully automated systems, often at the expense of pedagogical theory, instructor involvement, and the student’s own learning experience. It is essential to show the alignment between the identified research findings and the proposed framework. The following table maps each major finding or implication identified in the literature review to the corresponding phase or design principle of the proposed framework.

Table 4.12. Alignment Between the Proposed Framework and the Literature Review.

Research Finding and Identified Gap	How the MLA-PF Framework Addresses This
Lack of theoretical grounding (83.3% of reviewed studies lacked an explicit framework; Table 4.1)	The MLA-PF framework is explicitly grounded in Hattie and Timperley's (2007) feedback model and Shute's (2008) formative feedback principles. Each phase of the framework is mapped to specific theoretical constructs (Feed-Up, Feedback, Feed-Forward, and feedback levels), creating a cohesive, evidence-based model.
Dominance of fully automated systems over hybrid, human-in-the-loop models (58.49% automated via custom platform; Table 4.6)	The framework is fundamentally a hybrid, human-in-the-loop model. AI's role is to augment and support the core human-to-human interaction, not replace it. This is seen in the AI-Coached Peer Review (Phase 3), where AI scaffolds human reviewers, and in the instructor's active role across all phases.

Table 4.12 (continued)

Research Finding and Identified Gap	How the MLA-PF Framework Addresses This
Passive instructor role (27.1% not specified; monitoring 11.8%, supplemental guidance 8.2%, configuration 7.1%; Table 4.5)	The framework redefines the instructor’s role as an active facilitator. The instructor acts as a “pedagogical architect” in Phase 1, monitors and intervenes through the analytics dashboard in Phase 5, verifies feedback quality, and configures the AI’s behavior based on observed class needs.
Correctness-first paradigm (76.27% of studies had no feedback enhancement strategy beyond accuracy; Table 4.7)	The MLA-PF framework deliberately counters this paradigm by withholding the initial AI analysis (Phase 2). It prioritizes the conceptual, process-level, human-driven feedback in Phase 3, uses guided reflection in Phase 4 to actively engage the student in metacognitive processes, and operationalizes Hattie and Timperley’s process and self-regulation feedback levels.
Shallow student feedback uptake (42.4% direct revision only; 40.6% no uptake or unspecified; Table 4.8)	Phase 4 (Synthesized Feedback and Guided Reflection) is designed specifically to foster deeper engagement. The AI synthesis reduces cognitive load (operationalizing Shute’s manageability principle), dialogic interaction supports peer-mediated understanding, and targeted reflection prompts encourage students to create a revision plan, promoting metacognition and feed-forward.

Table 4.12 (continued)

Research Finding and Identified Gap	How the MLA-PF Framework Addresses This
<p>Limited personalization and adaptivity in feedback delivery (only 15.25% of studies used personalized feedback; Table 4.7; Personalization &amp; Adaptivity implication theme, Table 4.10)</p>	<p>The AI’s coaching in Phase 3 is adaptive: it uses the private analysis from Phase 2 to tailor its prompts based on the specific needs of each student author. This enables differentiated feedback guidance without a one-size-fits-all approach.</p>
<p>Importance of iterative feedback cycles (most frequently cited implication: multiple rounds of feedback; Table 4.10)</p>	<p>The framework is designed as a comprehensive cycle. Phase 5 (Revision, Resubmission, and Progress Analytics) closes the loop by tracking improvements against a baseline and sets the stage for further iterations, reinforcing the value of the revision process.</p>
<p>Need for human-AI collaboration and teacher verification (5.88% each, Human-in-the-Loop &amp; Agency theme; Table 4.10)</p>	<p>The instructor dashboard in Phase 5 provides tools for monitoring and verifying the quality of peer feedback. Instructors can spot-check comments, approve high-quality examples, configure AI behavior, or intervene directly when necessary, maintaining a crucial human element of oversight and quality control.</p>

Table 4.12 (continued)

Research Finding and Identified Gap	How the MLA-PF Framework Addresses This
Need for specific, instructive, and elaborated feedback (Feedback Content & Quality, 35.28% of implications; Table 4.10)	Phase 3 (AI-Coached Peer Review) provides just-in-time, scaffolded guidance to reviewers. The AI acts as a pedagogical coach, offering non-evaluative prompts aligned with Shute’s principles of specificity and elaboration, helping students write more specific, helpful, and constructive comments.

#### **4.1.14 Operationalization of the Framework: The AI-Based Peer Feedback Platform**

The conceptual framework and design principles presented above represent the theoretical contribution of this study. However, a framework’s value in design-oriented research ultimately depends on whether it can be translated into a functioning educational tool. The purpose of this section is to describe how the key principles of the MLA-PF framework were operationalized in a concrete platform designed for use in programming education.

Operationalization in this context refers to the process of translating abstract pedagogical constructs into specific interface components, interaction flows, and system behaviors that students and instructors can engage with directly. For example, the framework’s principle that AI should support rather than replace human judgment was realized through dialogue-based feedback interactions in which the system guides peer feedback. Similarly, the feed-up phase was operationalized by presenting students with the assignment rubric and learning objectives within the platform before they begin the feedback process, ensuring that expectations are visible and accessible throughout the task.

It is important to note that the platform, as piloted in Phase 2, did not implement the full scope of the MLA-PF cycle. Given the exploratory nature of the pilot, the implementation focused specifically on the AI-as-peer feedback component, in which students submitted their code and received dialogue-based feedback generated by a large language model. The implementation integrates Gemini 1.5 Flash as a technical layer to augment the peer feedback process, facilitating the automated generation and refinement of formative insights within the MLA-PF framework. Features such as human peer review coaching (Phase 3) and the full instructor analytics dashboard (Phase 5) were not active during the pilot. This scope was to gather initial evidence on the feasibility and perceived value of the platform's core feedback mechanism before expanding to the complete cycle in future iterations. The following subsection describes the platform's user interface and the interaction flow as experienced by students during the pilot.

The platform's landing page is provided in Figure 4.1, which mainly describes the purpose of the platform and lists the core features.

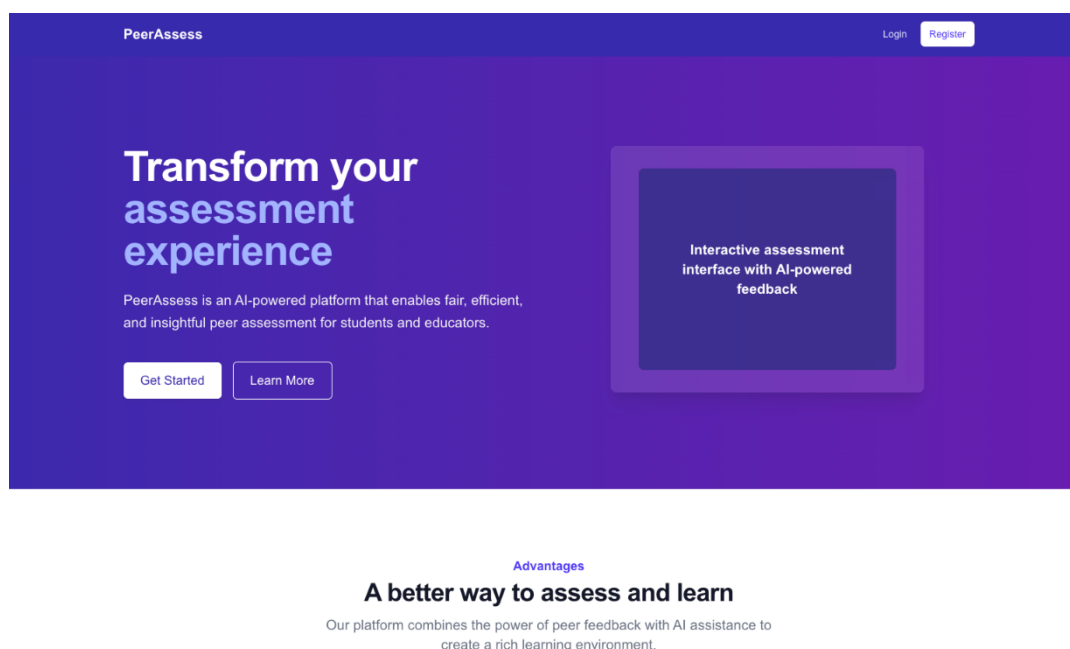


Figure 4.1 Landing page of the Platform

From the landing page, instructors can navigate to the dashboard page (see Figure 4.3) which provides all necessary tools and features to manage course, assignments, rubric, and peer reviews.

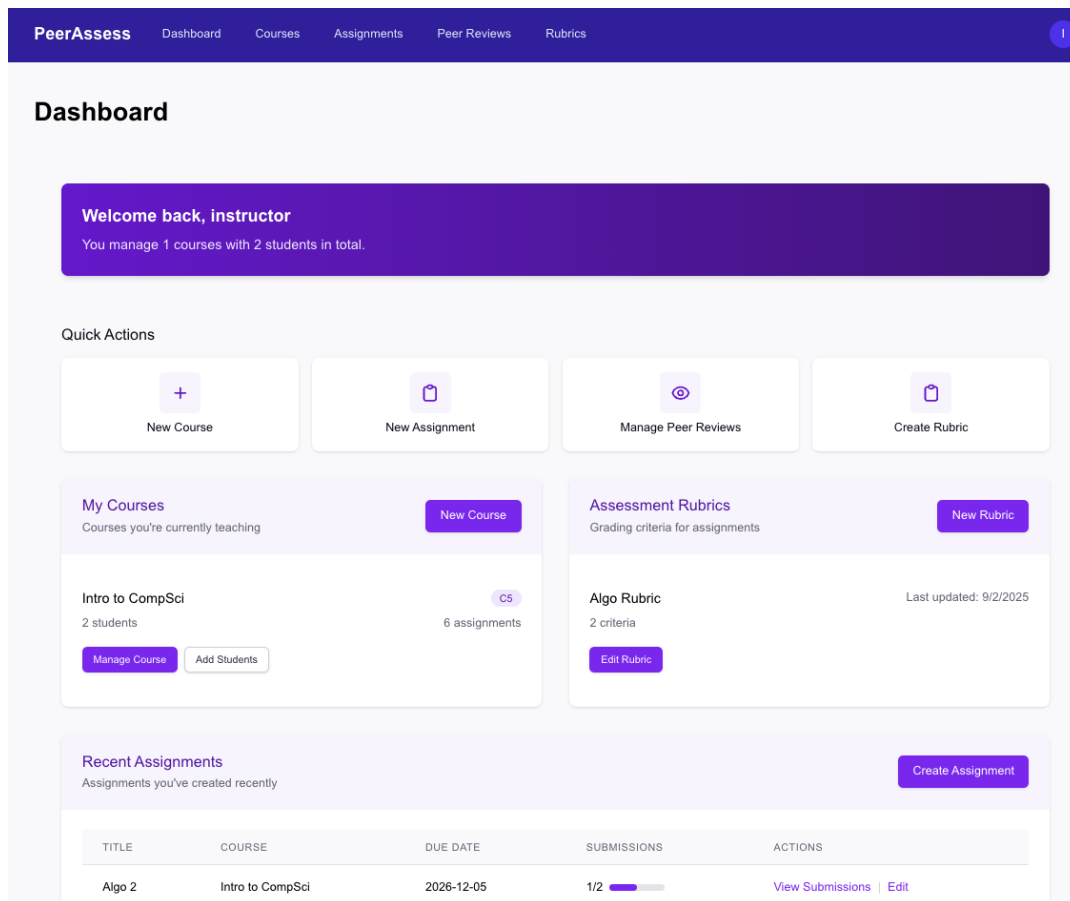


Figure 4.2 Instructor dashboard page

As seen in Figure 4.3, the component for creating a new course is a simple form where the instructor is required to provide the name and description for the course being created.

### Create New Course

Course Name\*  
e.g., Introduction to Computer Science

Description  
Provide a brief description of the course

Cancel Create Course

Figure 4.3. Create New Course

Moreover, instructors can manage the student enrollments once a course is created. Instructors may enroll students individually via email address or utilize the bulk upload feature for larger groups.

## Manage Students

[Back to Course](#)

Course: Intro to CompSci (C5)

### Add Single Student

Add a student by email to join this course. If the student is already registered, they will be added immediately. If not, an invitation email will be sent for them to register and join automatically.

### Bulk Add Students

Upload CSV File

No file chosen

Upload a CSV file with email addresses. The file should have an "email" column or emails will be auto-detected.

Or Enter Emails Manually

Enter email addresses (one per line, or separated by commas) example1@email.com example2@email.com, example3@email.com

Send Invitations (0)

Clear

### Students

Manage students in this course

<p style="font-size: 0.8em; margin: 0;">studentA</p> <p style="font-size: 0.8em; margin: 0;">✉ studentA@gmail.com</p>	<input style="border: 1px solid #ccc; padding: 2px 5px; font-size: 0.8em;" type="button" value="Remove from Course"/>
<p style="font-size: 0.8em; margin: 0;">studentB</p> <p style="font-size: 0.8em; margin: 0;">✉ studentB@hotmail.com</p>	<input style="border: 1px solid #ccc; padding: 2px 5px; font-size: 0.8em;" type="button" value="Remove from Course"/>

Figure 4.4 Managing enrollments

To create an assignment for a course, instructor has to provide the title, due date, rubric and description information in the form displayed in Figure 4.7..

**Create New Assignment** Cancel

Assignment Title \*  
Big O Test

Course \*  
Select a course  
Found 1 course(s) available.

Due Date \*  
dd.mm.yyyy

Assessment Rubric (Optional)  
Select a rubric (optional)  
Choose a rubric to define assessment criteria for this assignment. You can also create or assign a rubric later.

Description  
Describe the assignment requirements, objectives, and any special instructions...

Figure 4.5 Creating new Assignment

Beyond the basic information about the assignments, the platform allows instructors to configure various AI features to shape the AI assistance and behavior during the peer feedback processes. One configuration in this regard corresponds to enabling AI review analysis (see Figure 4.6), referring to the AI providing feedback to the peer feedback. For this configuration, once enabled, instructor can use the default system prompt or write their own prompt to tell AI how to respond when providing feedback to peer feedback. As the second configuration (see Figure 4.7), instructor can enable to AI-generated peer reviews, in which AI behaves like a peer and provides feedback to an assigned student work. Similarly, in this feature instructor can use the default system prompt or write their own instructions.

**AI Review Analysis Configuration**  
 Configure AI prompts for peer review analysis. Students will receive AI-powered suggestions to improve their reviews.

**Enable AI-powered review analysis for this assignment**

**Overall Feedback Analysis Prompt**  
 Configure the instructions for AI analysis of overall feedback. System context (assignment details, scores) is added automatically.

**Please provide constructive suggestions to improve the overall feedback. Focus on making the feedback more helpful, specific, and balanced. Keep your response concise and actionable.**

**Criteria-Specific Feedback Analysis Prompt**  
 Configure the instructions for AI analysis of individual criteria feedback. System context (criterion details, scores) is added automatically.

**Your tasks:**

1. Suggest 1 specific improvement to the feedback for this criterion to the reviewer. The suggestion should be concise and actionable.
2. After the suggestion, provide a single revised version of the feedback as if written by the reviewer, incorporating the improvement. Write it in the reviewer's voice.

**Format your response as:**

1. [suggestion]. Revised Feedback Example: "[your rewritten reviewer feedback here]"

Figure 4.6. Disabling or Enabling AI Features

**Instructor AI Review Generation**  
 Allow instructors to generate AI peer reviews for submissions. Reviews will appear anonymous to students.

**Enable AI peer review generation for instructors**

**Custom AI Peer Review Prompt (Optional)**  
 Customize how the AI generates peer reviews. Leave empty to use the default peer-style prompt. Assignment details, submission content, and rubric criteria are added automatically.  
 Leave empty for default peer-style reviews, or enter custom instructions...

Example: "Focus on encouraging creative aspects and provide specific suggestions for improvement using casual, supportive language."

Figure 4.7. Disabling or Enabling AI Features

For assignments, instructors can create rubric, which is later used to assess peers' work and provide feedback. Within a rubric, instructors can determine criteria, performance levels, and scoring ranges (see Figure 4.11).

## Create New Rubric Cancel

**Rubric Name \***  
e.g., Essay Evaluation Rubric

**Assignments (Optional)**

- Algo Test - Intro to CompSci
- Algo 2 - Intro to CompSci (Already has rubric)
- safesf - Intro to CompSci (Already has rubric)
- safesf - Intro to CompSci (Already has rubric)
- Algo 3 - Intro to CompSci (Already has rubric)

Select the assignments this rubric will be used for. You can select multiple assignments or none.

**⚠️ Assignments that already have a rubric cannot be selected. Each assignment can only have one rubric.**

**Description**  
Describe the purpose of this rubric

Figure 4.8. Creating Rubric

## Assessment Criteria

Define the criteria that will be used to evaluate submissions.

---

**Criterion 1**

**Name \***  
e.g., Content Quality

**Description**  
Describe what this criterion evaluates

**Maximum Points**  **Weight**   
Weight determines the relative importance of this criterion (default: 1.0)

**Performance Levels** Add Level

LEVEL	DESCRIPTION	SCORE	
Level 1	Does not meet expectations	<input type="text" value="3"/>	Remove
Level 2	Partially meets expectations	<input type="text" value="5"/>	Remove
Level 3	Meets expectations	<input type="text" value="8"/>	Remove
Level 4	Exceeds expectations	<input type="text" value="10"/>	Remove

+ Add Criterion

Cancel Create Rubric

Figure 4.9. Assessment Criteria

For each assignment (see Figure 4.10), instructors can configure peer review matchings. As seen in Figure 4-11, they can assign each peer to review manually, or they can do automatic assignment, which randomly matches students to review each other's work.

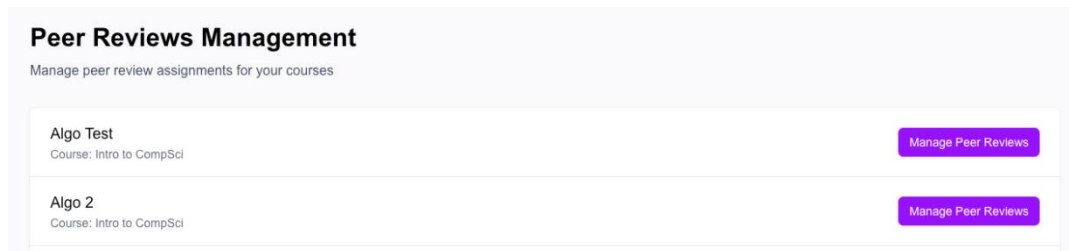


Figure 4.10 Managing Peer Reviews

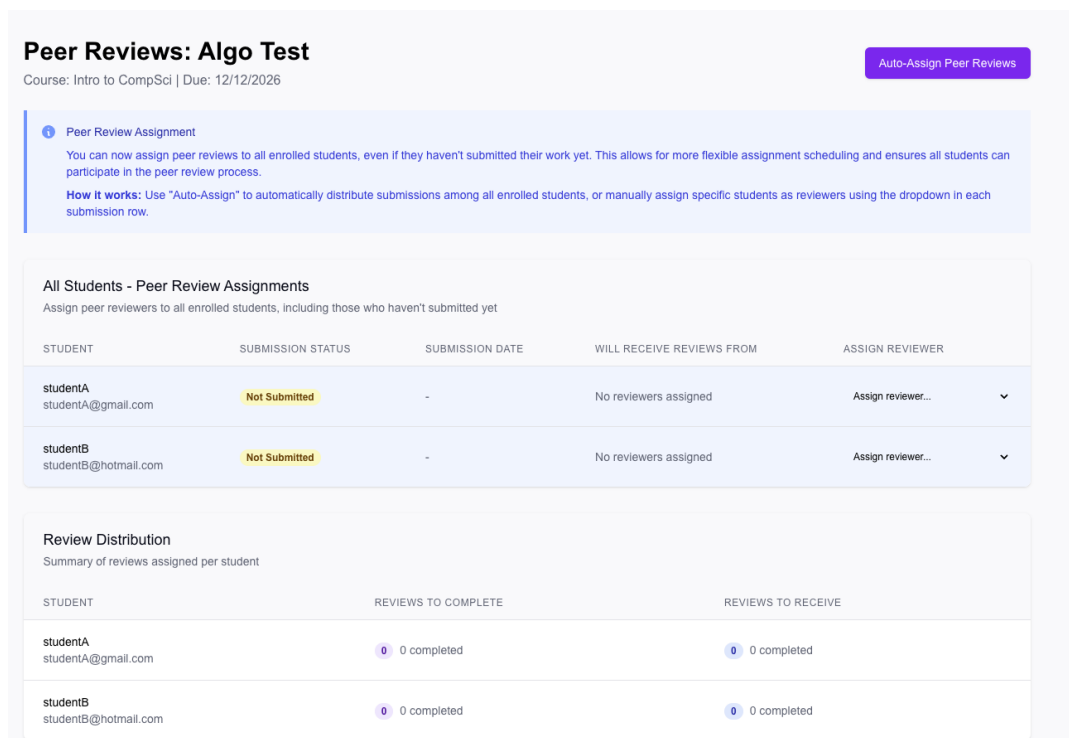


Figure 4.11 Managing Peer Reviews

For a specific assignment, instructors can click “View Submissions” button to view existing student submissions (see Figure 4.12), where they can click AI Review to generate feedback by AI acting as peer.

The screenshot displays the 'test Submissions' page. At the top, there are navigation buttons: 'Back to Dashboard', 'Edit Assignment', and 'Go to Course'. Below this, a 'Submission Status' summary shows 1 student submission, 0 reviewed submissions, and 1 pending review submission. A section titled 'Student Submissions' lists all submissions for this assignment. The table below contains one submission entry.

STUDENT	SUBMISSION	DATE	STATUS	SCORE	ATTACHMENTS	REVIEWS	ACTIONS
studentB muhammad.sinar0@hotmail.com	afscasfas	Aug 20, 2025, 01:58 PM	Submitted	-	No files	-	<a href="#">View</a> <a href="#">Review</a> <a href="#">AI Review</a>

Figure 4.12 Submissions list

Once the feedback is generated by AI, instructor needs to oversight, edit it as needed, and confirm it. Then, the feedback will be accessible to the corresponding student.

### AI Peer Review Preview

Generated peer-style feedback for studentB's submission

---

#### Overall Feedback

Hey there! I checked out your submission for the assignment, and I think there's some potential there. Your content seemed to be a bit brief, but I see where you're going with it. Let me break it down a bit more for you.

#### Detailed Scores

Precision	5/10	50%
So, for precision, I felt like your explanation was a bit vague. It would be great if you could elaborate more on the topic and provide more detailed information. Maybe include some examples or expand on your points to make it clearer.		
Big O	7/10	70%

AI-generated peer-style review using gpt-3.5-turbo • Will appear as anonymous peer feedback to the student

Figure 4.13 Getting AI Review

From the student dashboard (see Figure 4.1), students can access 4 main components: My Course, My Assignment, My Submissions and Peer Reviews

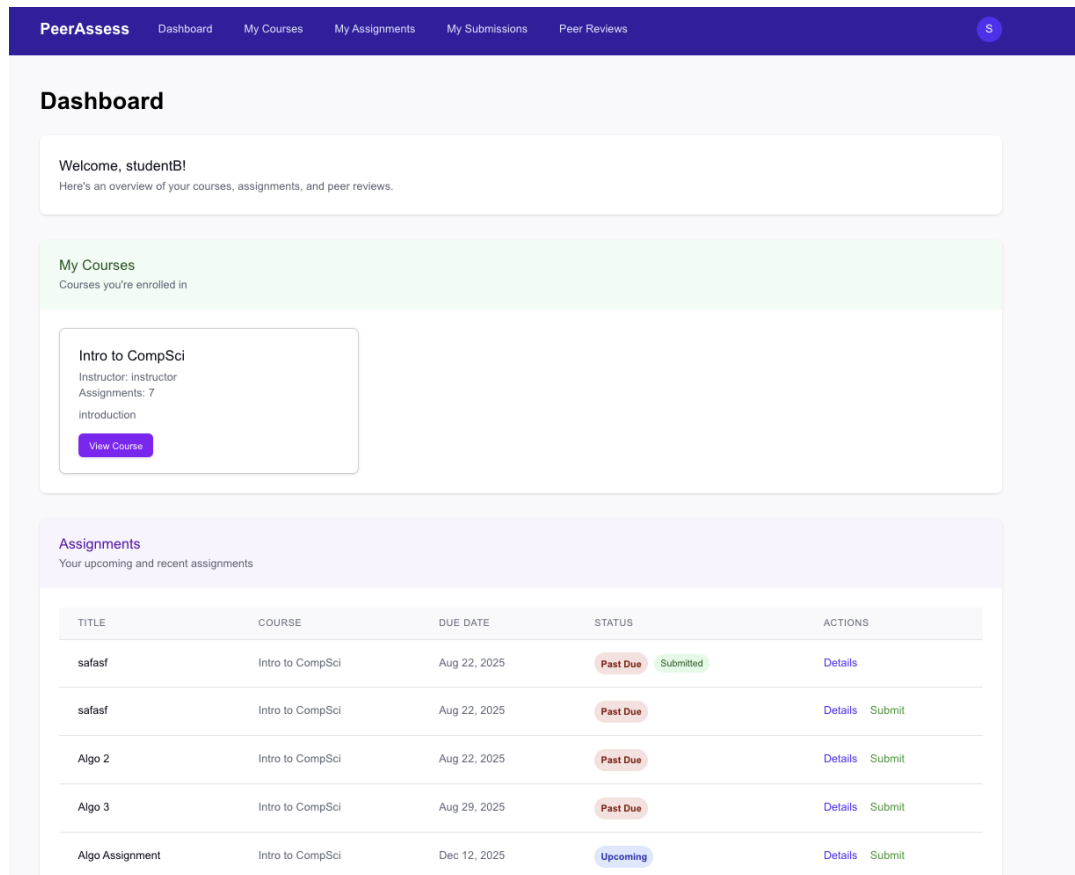


Figure 4.14 Student Page Dashboard

For any available assignment, they can click on Submit button to work on their submissions. They can either provide their submissions either as the text or as a file using the upload feature (see Figure 4.15).

Once the submission is posted, in the background, AI silently evaluates the work based on instructor rubric and assign a score, and based on this score categorize student knowledge level: high, medium, or low. This information about the students'

knowledge levels are stored in the database, and used to customize AI support in peer feedback processes, as suggested in the theoretical framework.

### Assignment Details

Algo test. Be careful.









### Submit Your Work

Please provide a title, content, and any attachments for your submission.


**Submission Title \***  
e.g., Assignment 1 Submission - [Your Name]  
Give your submission a clear title so it's easy to identify

**Submission Content \***

File Edit View Insert Format


↶ ↷ **B** *I*  ▾       *I*<sub>x</sub> 

p

Build with  tinyMCE

Provide a detailed response to the assignment prompt. You can format text, add images, and more.

**Attachments**

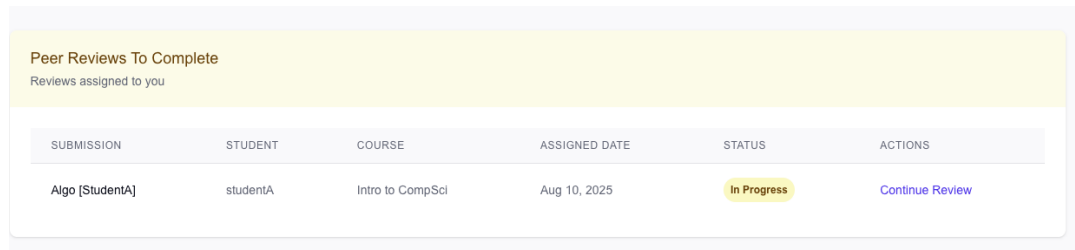


Upload files or drag and drop  
PDF, Word, Excel, PowerPoint, ZIP up to 10MB each

Cancel Submit Assignment

Figure 4.15.Submitting Assignment

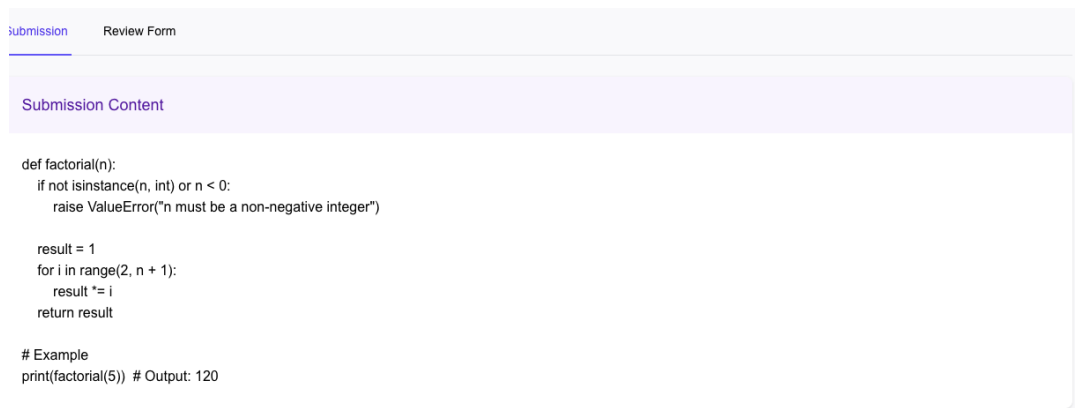
On a separate page, students can view the assigned peer reviews for the selected assignment (see Figure 4.16).



SUBMISSION	STUDENT	COURSE	ASSIGNED DATE	STATUS	ACTIONS
Algo [StudentA]	studentA	Intro to CompSci	Aug 10, 2025	In Progress	<a href="#">Continue Review</a>

Figure 4.16. Assignment Details

On this page, by clicking on “Continue Review”, then can access the peer review page as shown in Figure 4.17. On this page they can display the peer submission and use the Review Form (see Figure 4.18) to assess the work and provide feedback. The review form is based on the rubric created by the instructor.



```
def factorial(n):
    if not isinstance(n, int) or n < 0:
        raise ValueError("n must be a non-negative integer")

    result = 1
    for i in range(2, n + 1):
        result *= i
    return result

# Example
print(factorial(5)) # Output: 120
```

Figure 4.17 Submission Details

The student can review their peers submission, and give proper feedback

## Peer Review: Algo Assignment

By: studentA | Course: Intro to CompSci | Submitted: 8/10/2025

Submission [Review Form](#)

---

### Precision

Performance Level

- 2 points:** Does not meet expectations
- 4 points:** Partially meets expectations
- 8 points:** Meets expectations
- 10 points:** Exceeds expectations

Selected: 0/10 points

Feedback

Provide constructive feedback for this criterion...

---

### Big O

Performance Level

- 2 points:** Does not meet expectations
- 4 points:** Partially meets expectations
- 7 points:** Meets expectations
- 10 points:** Exceeds expectations

Selected: 0/10 points

Feedback

Provide constructive feedback for this criterion...

---

Figure 4.18 Peer review form

Once students complete their review, they can get feedback from AI on their reviews (see Figure 4.19 and 4.23). While by default AI focuses on the quality of feedback and its correctness considering the submission, the system prompt written by instructors play an essential role in how AI generates feedback in this phase. AI behaves adaptively based on the knowledge level of the (author) student. For example, if student is marked with low knowledge level, then AI provide suggestions to be more explicit and detailed.

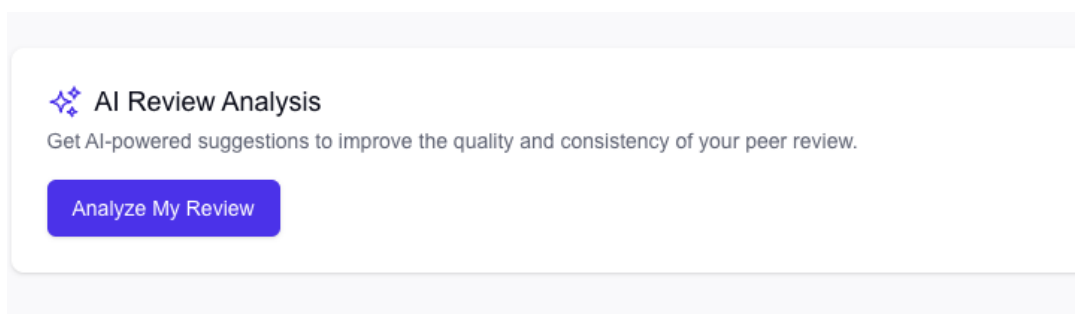


Figure 4.19. AI Review The Analysis

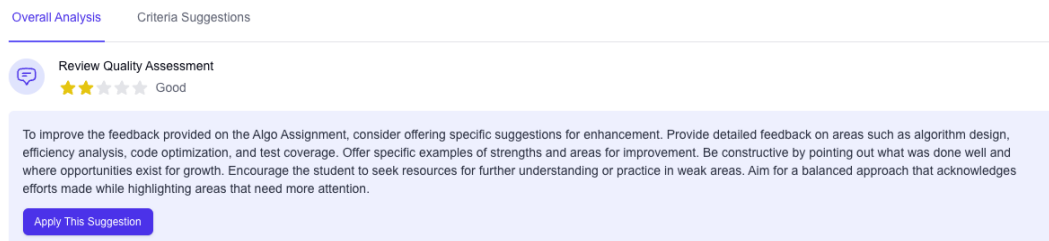


Figure 4.20. Getting Suggestion From Ai

Additionally, as seen in Figure 4.21, AI provides criteria-specific suggestions as well to provide more granular support helping peers to improve their feedback.

**Precision**  
Your score: 4/10

1. Provide specific examples or details to explain why the algorithm is not working correctly. This will help the recipient understand the issue better and make improvements more effectively. Revised Feedback Example: "The algorithm is not working correctly due to inconsistencies in the sorting logic, which is causing unexpected results."

[Apply This Suggestion](#)

**Big O**  
Your score: 4/10

1. Provide specific examples or explanations to support the assessment of "Bad big o".

[Apply This Suggestion](#)

Figure 4.21. Getting Suggestion From AI

On the reviews page (see Figure 4.22), students can access the reviews they provided and those they received as well. They can access peer feedback on their work by clicking on "View Feedback" button as shown.

### My Peer Reviews

View and complete your assigned peer reviews

#### Reviews to Complete

Algo [StudentA]  
Assignment: Algo Assignment  
Course: Intro to CompSci  
Student: studentA  
Assigned: 8/10/2025

[In Progress](#) [Continue Review](#)

#### Reviews Completed (Given Feedback)

No completed reviews yet

#### Feedback Received

Student A Algo 2  
Assignment: Algo 2  
Course: Intro to CompSci  
Reviewer: Anonymous Reviewer  
Completed: 9/2/2025  
Score: 9

[View Feedback](#)

Hey there! I took a look at your Algo 2 test submission. I see you've started on the bad\_permutations function. It's great that you're diving into recursion! However, there are some areas where we can improve to make this function more efficient and readable.

Figure 4.22. Feedback Details

Figure 4.23 presents the interface where students can access their peer feedback. This page provides a structured view of overall scores, general comments, and criterion-specific feedback. Following the system's theoretical framework, the AI can synthesize evaluations from multiple peers into a concise summary. This allows students to efficiently grasp diverse perspectives, thereby facilitating more effective feedback uptake. Note that because the specific example in Figure 4.23 displays feedback from only a single peer, this automated synthesis feature is not currently shown.

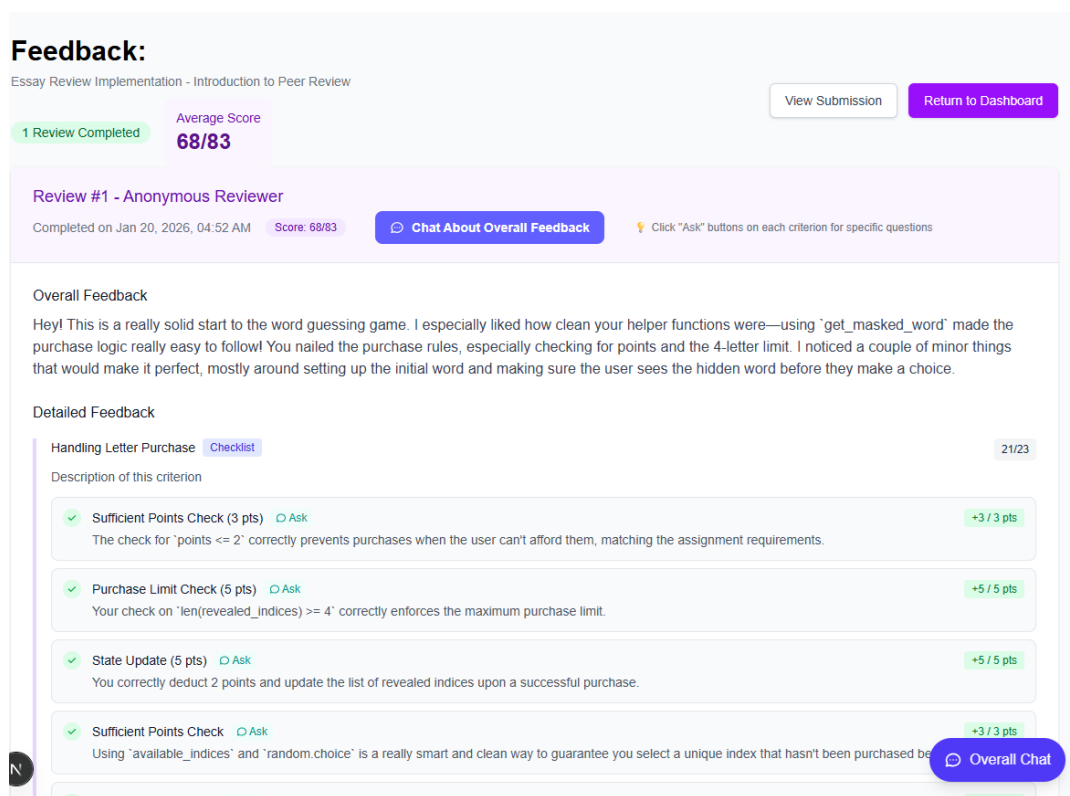


Figure 4.23 Peer feedback page

As illustrated in Figure 4.24, any criterion where a student lost points includes a dedicated chat option for inquiry. This feature enables students to reflect on and clarify feedback to determine their next steps, aligning with the study's grounding

theoretical framework. Additionally, this dialogic interface can be configured for interactions with either the peer reviewer or the AI..

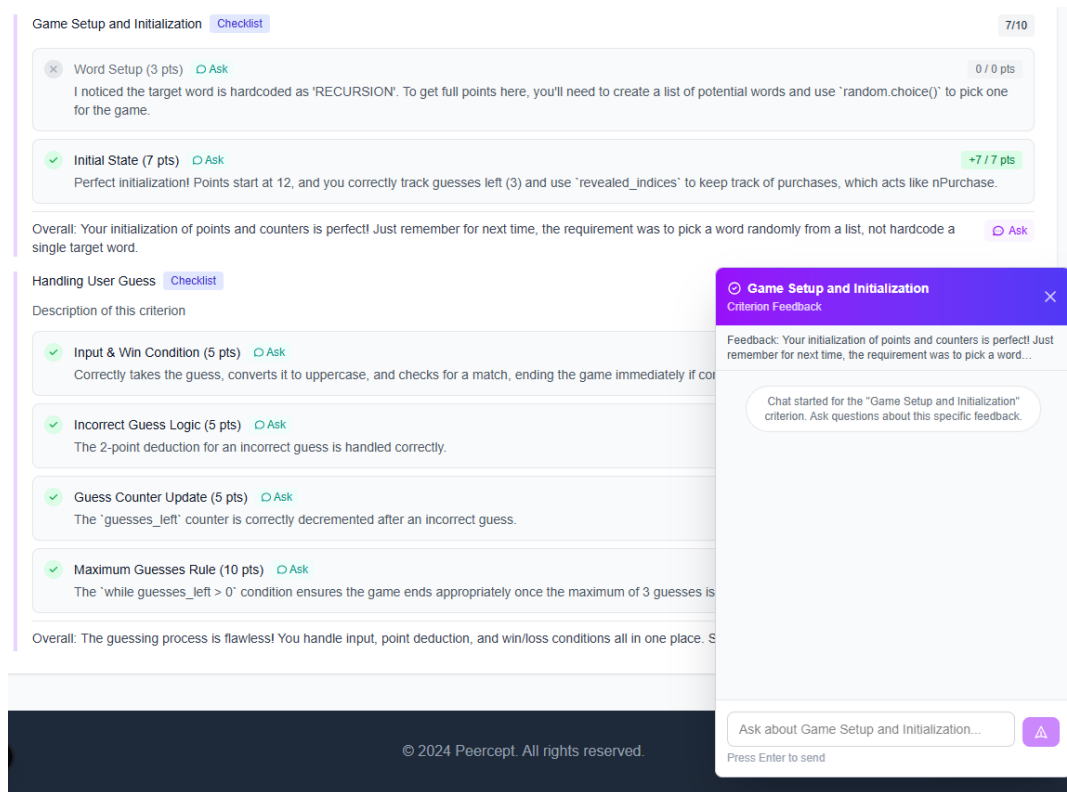


Figure 4.24 Chatbot for criteria-specific feedback

When this dialogic process occurs between human peers, the AI is configured to intervene and facilitate the exchange. This intervention aims to enhance the quality of the interaction, ensuring that the reviewer provides constructive guidance and the author student accurately interprets and applies the feedback to their work. In this process, students' knowledge level is taken into consideration to provide adaptive support. This is illustrated in Figure 4.25.

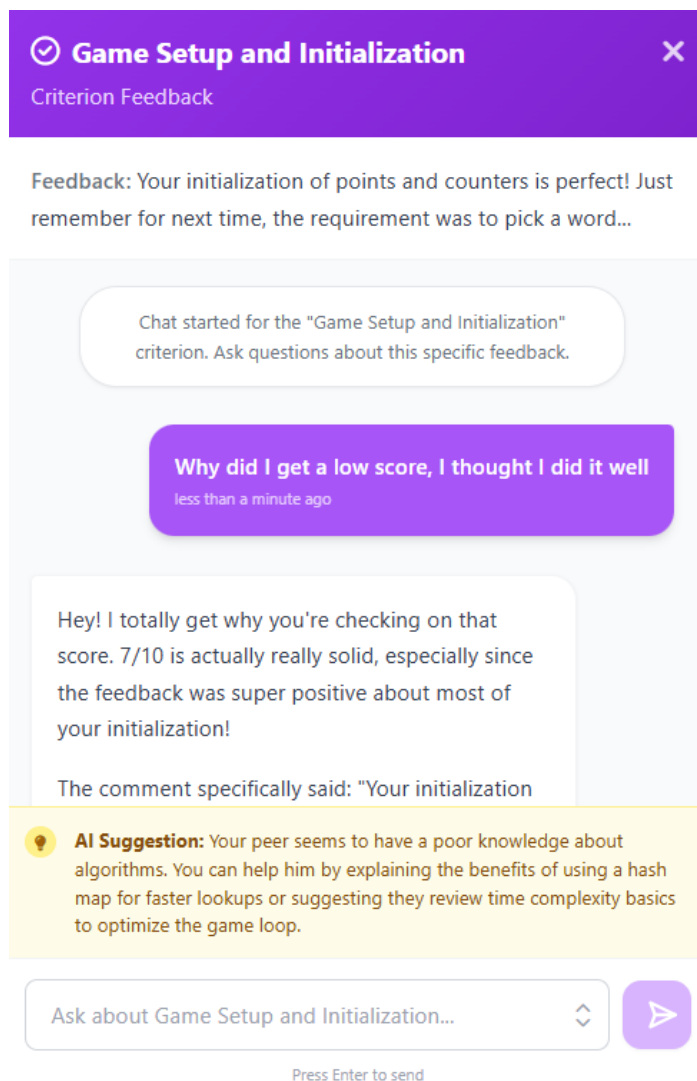


Figure 4.25 Example chatbot interactions for criteria-specific feedback

## 4.2 Results of Pilot Study (Answering RQ2 & RQ3)

This section presents the findings of the pilot study, which investigated students' perceptions and experiences with an AI-powered peer feedback system (Peercept) in a programming education context. The results are organised into two main sections. The first section reports the qualitative findings derived from thematic

analysis of open-ended survey responses, while the second section presents the quantitative results based on descriptive statistics and frequency analysis.

**4.2.1 Result of Thematic Analysis (Qualitative Findings)**

Open-ended survey responses were analysed using thematic analysis to identify recurring patterns in students’ experiences with the AI-based feedback system. The analysis was structured around five key questions, each addressing a different dimension of the feedback experience. For each question, themes and subthemes were identified and are presented in the tables below, accompanied by interpretive discussion.

**4.2.2 Perceived Strengths and Weaknesses of the Feedback Process**

Students were asked to identify what they considered to be the primary strengths of the feedback process within Peercept. The themes that emerged are summarised in

Table 4.13. Thematic Analysis of Perceived Strengths of the Feedback Process

Main Theme	Subtheme	Definition
Skill Development and Support	Improved performance and coding skills	Students reported that the feedback helped improve their overall programming performance
	Enhanced cognitive and critical thinking skills	Not providing direct answers encouraged students to think critically about their mistakes

Table 4.13 (continued)

Main Theme	Subtheme	Definition
High quality and detailed feedback	Detailed, specific, and diverse reviews	Students noted that feedback covered multiple aspects of their code, helping them identify weaknesses
Error Detection and Diagnostic Value	Clear identification of mistakes	Feedback was perceived as accurate in detecting and highlighting problems in the code
Structured and Segmentated Feedback	Section-by-section code feedback	Dividing the code into sections and providing targeted comments for each part was seen as clear and helpful
Chatbot interaction and support	Communication and follow-up Q&A	Students valued the ability to ask follow-up questions through the chatbot during the feedback process

The analysis revealed that students were generally satisfied with the feedback they received. A key strength was that the system refrained from providing complete solutions, which participants believed encouraged deeper engagement with their mistakes and promoted the development of critical thinking skills.. In addition,

students believed that the feedback was sufficiently diverse and covered multiple aspects of their code, helping them identify missing parts and weaknesses.

The clear identification of specific errors was viewed as particularly beneficial, and the segmented structure of the feedback, whereby comments were provided for individual sections of the code rather than the assignment as a whole, was found to enhance comprehension. Finally, the ability to communicate with the chatbot and ask follow-up questions was identified as another important strength of the platform.

Students also identified a number of weaknesses, which are presented in Table 4.14.

Table 4.14. Thematic Analysis of Perceived Weaknesses of the Feedback Process

Main Theme	Subtheme	Definition
Unclear feedback	Time consuming and slow process	Students frequently reported long waiting times before receiving feedback
	Vague feedback with insufficient explanation	Some comments lacked depth and did not explain how to resolve identified issues
	Absence of specific error location	Feedback discussed problems at a section level but did not indicate the exact line of the error

Table 4.14 (continued)

Main Theme	Subtheme	Definition
Grading problem	Strict and inflexible rubric	Students felt the rubric did not accommodate alternative but valid coding approaches
	Perceived unfairness in scoring	Some students objected to scores they received despite having functional code
UI & Usability Issues	Chatbot size and visual design	Students recommended improvements to the chatbot interface, fonts, and background colours
Chatbot Performance	Repetitive and occasionally insufficient responses	The chatbot sometimes repeated the same explanation or failed to detect specific mistakes

The most frequently cited concern was the speed of the feedback process. Many students reported that waiting times were excessively long and that this negatively affected their experience. Additionally, while students acknowledged the quality of

the feedback itself, a number of participants felt that the comments were too general. Specifically, they noted that feedback often identified the presence of a problem without offering sufficient guidance on how to resolve it, and without indicating the exact location (e.g., line number) of the error in the code.

Grading was another area of concern. Several students objected to the scoring system, arguing that the rubric was overly rigid and did not account for alternative coding approaches or different logical structures that still produced correct outputs. Furthermore, the platform’s user interface was criticised for its visual design, including font choices, background colour, and the layout of the chatbot. Although most students found the chatbot to be helpful, some reported that its responses could be repetitive and did not always address their specific questions adequately.

### 4.2.3 Technical Accuracy and Helpfulness of Feedback

Students were asked to evaluate the quality of the feedback, including the chatbot, in terms of technical accuracy and helpfulness. The resulting themes are presented in Table 4.15.

Table 4.15 Quality of the Feedback

Main Theme	Subtheme	Definition
Quality of Technical Accuracy and Helpfulness of Feedback	High technical accuracy	Students considered the feedback to be technically correct and reliable

Table 4.15 (continued)

Main Theme	Subtheme	Definition
Quality of Technical Accuracy and Helpfulness of Feedback	Helpfulness for improving code	Feedback was seen as useful for identifying weak points and improving the quality of submissions
	Repetitive chatbot responses	Some students reported receiving the same responses from the chatbot regardless of the question asked
	Insufficient personalized feedback	Students expressed a desire for more tailored feedback aligned with individual coding styles and preferences
	Insufficient depth of explanation	Feedback was perceived as too general and lacking the level of detail needed for full understanding

The findings indicate that students generally regarded the feedback as technically accurate and reliable. The majority of participants reported that the feedback helped them identify weaknesses in their code and improve the quality of their submissions. However, some students reported that they faced repetitive comments from the chatbot even after asking different questions and that they received the same responses.

Furthermore, while the technical accuracy of the feedback was not in question, students expressed a desire for greater depth and personalisation.. They felt that the feedback tended to be general rather than tailored to the specific context of their code, and that it often stopped at identifying the problem without offering a sufficiently detailed explanation. These findings suggest that although the system’s diagnostic accuracy was high, there remains scope for improvement in the richness and specificity of the feedback provided.

#### 4.2.4 Clarity and Understandability of Feedback

Students were asked to rate how clear and understandable the feedback was, including the chatbot, and to provide examples of unclear or ambiguous comments where applicable. The identified themes are shown in Table 4.16.

Table 4.16. Thematic Analysis of Feedback Clarity and Understandability

Main Theme	Subtheme	Definition
Perceived Clarity and Understandability of Feedback	Clear and well-structured feedback	The segmented format of the feedback made it visually more accessible and easier to follow

Table 4.16 (continued)

Main Theme	Subtheme	Definition
Perceived Clarity and Understandability of Feedback	Need for more detailed explanations	Students reported a lack of explanation as to why their code was marked as incorrect
		Some comments used overly general praise (e.g., “great” or “you are doing well”) without substantive guidance
	Unclear comments	

Students generally found the feedback to be clearly structured, particularly due to the segmented format in which comments were organised by code section. This approach was seen as enhancing the visual clarity and readability of the feedback. Nevertheless, participants identified areas where clarity could be improved. Specifically, several students noted that feedback often flagged errors without explaining the underlying reason, making it difficult for them to fully understand the issue and learn from it.

#### 4.2.5 Changes Made to Code Based on Feedback

Students were asked to describe the specific changes they made to their code as a result of the peer feedback. The themes that emerged are presented in Table 4.17.

Table 4.17. Thematic Analysis of Code Changes Based on Feedback

Main Theme	Subtheme	Definition
Code Completion and Structural Refinement	Completing missing components	Students added parts of the assignment they had overlooked or forgotten to include
	Correcting logical & control-flow errors	Changes included restructuring loops, adjusting return statement placement, and improving programme flow
	Syntax-level corrections	Students corrected syntax errors such as mismatched file names and incorrect operators (e.g., changing = to ==)
	Functional / Feature Enhancements	Some students added new functionality, such as file existence verification, based on feedback suggestions
	Code Structure & Readability Enhancement	Students improved readability by adding comments and reorganising code for smoother progression

The results demonstrate that the feedback prompted a range of meaningful revisions to students' code. The most commonly reported change involved completing missing components that the assignment required but that students had initially overlooked. In addition, students made corrections to logical and control-flow errors, including the repositioning of return statements and the restructuring of loops to improve programme execution.

Syntax level corrections were also frequently mentioned, with students addressing issues such as mismatched file names and incorrect comparison operators. Beyond corrective changes, some students reported making functional enhancements—for example, adding file existence checks that improved the robustness of their code. Finally, several participants improved the overall readability and structure of their code by adding inline comments and reorganising sections for greater clarity. Collectively, these findings suggest that the AI-based feedback system was effective in guiding students toward substantive and diverse improvements in their programming work.

#### 4.2.6 Suggested Improvements to the Feedback Activity

Table 4.18. Thematic Analysis of Suggested Improvements

Main Theme	Subtheme	Definition
Platform Usability & Interface Design	Chatbot size and layout	Students requested a larger chatbot interface to better accommodate code and reflections
	Overall platform interface	Suggestions included improving accessibility, font choices, and adding example buttons

Table 4.18 (continued)

Main Theme	Subtheme	Definition
Speed & Real-Time Feedback	Need for faster and more timely feedback	Students consistently emphasised that reducing response time should be a priority
Communication & Interaction with chatbot	More interactive chatbot communication	Students wanted the ability to interact with the chatbot while making changes to their code in real time
Quality of Explanations & Examples	Need for more explanation not just comments	Students preferred concise, clear explanations over lengthy, repetitive comments
Personalised and Detailed Feedback	More detailed and targeted feedback	Students expressed a preference for shorter, more precise, and individually tailored responses

The analysis of students' suggestions for improvement revealed several consistent themes. First, the platform's user interface was a frequent target of criticism, with students recommending changes to the chatbot's size, the overall layout, font choices, and the visual presentation of feedback. Second, students expressed a strong desire for faster feedback delivery, identifying response time as one of the most critical areas for improvement.

Third, students advocated for a more interactive feedback experience, suggesting that they should be able to update their code and receive real-time responses from the chatbot, rather than submitting the entire code and waiting passively. Fourth, the quality and format of explanations were also raised: students preferred concise and

substantive comments over lengthy or repetitive ones. Finally, the need for greater personalisation was reiterated, with students requesting feedback that was more specifically tailored to their individual code and coding approach, rather than generic observations.

### 4.3 Quantitative Results

This section presents the quantitative findings of the study, including descriptive statistics for the overall AI experience variable and frequency analysis of students' self-reported feedback usage behavior.

#### 4.3.1 Descriptive Statistics

Descriptive statistics were calculated to summarize students' overall experience with the AI-based feedback system.

Table 4.19. Descriptive Statistics for the AI Experience Variable

	N	Minimum	Maximum	Mean	Std.Deviation
AI_Experience_Mean	98	2.42	5.00	3.9334	.49567

Data from 98 participants were included in the analysis. The mean score for the overall AI experience variable was  $M = 3.93$  ( $SD = 0.50$ ), indicating a generally positive evaluation of the system. Scores ranged from a minimum of 2.42 to a maximum of 5.00.

The relatively high mean value suggests that students were, on average, satisfied with their experience of using the AI-based feedback tool. In addition, the low standard deviation indicates a high degree of consistency in students' responses, reflecting a homogeneous perception of the system across the sample. Although a small number of participants reported lower satisfaction scores, the overall results

demonstrate that the AI feedback system was positively received by the majority of the students.

### 4.3.2 Frequency of Feedback Usage

Students were asked to indicate how frequently they took feedback comments, including those generated by the chatbot, into consideration when revising their code. Responses were collected on a five-point Likert scale. Descriptive results are presented in Table 4.19, and the frequency distribution is detailed in Table 4.20.

Table 4.20. Descriptive Statistics for Feedback Usage Frequency

Item	N	M	SD
Frequency of feedback consideration during code revision	98	3.35	0.79

Table 4.21 How often did you take into consideration feedback comments (including the chatbot) when you revised your code?

	Frequency	Percent	Valid Percent	Cunulative Percent
2 (Rarely)	13	13.3	13.3	13.3
3 (Sometimes)	44	44.9	44.9	58.2
4 (Often)	35	35.7	35.7	93.9
5 (Always)	6	6.1	6.1	100.0
Total	98	100.0	100.0	

The mean score was  $M = 3.35$  ( $SD = 0.79$ ), indicating a moderate to high frequency of feedback usage among participants. The largest proportion of respondents (44.9%,

n = 44) selected a score of 3,, suggesting that they sometimes considered the feedback. In addition, 35.7% (n = 35) selected a score of 4, and 6.1% (n = 6) selected 5, indicating that over 41.8% of students reported that they often or always used the feedback when revising their code. Conversely, 13.3% (n = 13) selected a score of 2, suggesting that a smaller proportion of students used the feedback less frequently.

Overall, these findings suggest that the majority of students actively engaged with the feedback provided by the system, with more than four out of ten students frequently incorporating feedback into their code revision process. This result highlights the practical utility of the AI-based feedback system in supporting students' programming revision practices, and suggests that students perceived the feedback as sufficiently valuable to integrate into their workfl



## CHAPTER 5

### DISCUSSION

#### 5.1 Pedagogical Grounding of the MLA-PF Framework (RQ1)

A central contribution of this study is the Multi-Layered AI-Augmented Peer Feedback (MLA-PF) framework, developed through a systematic literature review to address the persistent lack of theoretical grounding in programming feedback systems. The review confirmed that existing automated assessment tools overwhelmingly prioritise technical correctness over pedagogical quality, consistent with critiques by (Keuning et al., 2019). In response, the MLA-PF framework was anchored in Hattie and Timperley's (2007) feedback model and Shute's (2008) formative feedback principles, ensuring that each phase of the cycle is aligned with established educational theory.

A particularly distinctive design choice is the deliberate withholding of the initial AI analysis from the student-author in Phase 2. This directly addresses the risk of over-reliance on automated tools, which can encourage shallow, fix-oriented behaviour rather than deep conceptual engagement (Boud & Molloy, 2013b). By leveraging the AI's diagnostic output into coaching the peer reviewer in Phase 3, the framework prioritises human-mediated, dialogic feedback, an approach advocated by (Winstone & Carless, 2019; Banihashem et al., 2025). Furthermore, the framework's cyclical structure, culminating in guided reflection and progress analytics, aligns with models of self-regulated learning (Nicol & Macfarlane-Dick, 2006), moving beyond the correctness-first paradigm that characterises most programming feedback tools.

## **5.2 Perceived Effectiveness and Usefulness of AI-Assisted Feedback**

The pilot study findings indicate that students generally perceived the AI-assisted feedback as effective and useful. The mean overall AI experience score of 3.93 out of 5.00 (SD = 0.50) reflects a consistently positive reception, while the low standard deviation suggests homogeneous perceptions across the sample. Qualitatively, students highlighted the system's ability to provide detailed, multi-dimensional feedback addressing functionality, structure, and style. The segmented format, in which comments were organised by code section, was particularly valued for its clarity, consistent with Shute's (2008) emphasis on specificity and manageability as characteristics of effective formative feedback.

Notably, students reported that the system's approach of withholding direct solutions encouraged critical thinking about their errors, aligning with the pedagogical intent of the framework and the literature on productive failure (Kapur, 2008). The thematic analysis of code changes corroborates this: students made substantive revisions ranging from correcting logical and control-flow errors to adding new functionality and improving code readability, indicating meaningful engagement rather than superficial corrections.

However, limitations were also identified. Students raised concerns about the depth and personalisation of feedback, noting that comments sometimes identified problems without sufficiently explaining their underlying causes. This tension between diagnostic accuracy and explanatory richness is a recognised challenge in AI-generated feedback (Guo et al., 2024) and suggests that the current implementation has not yet fully achieved the elaborated feedback quality prescribed by Shute's (2008) framework. The chatbot's tendency toward repetitive responses further indicates a need for improved prompt engineering and contextual adaptation.

### **5.3 Students' Experiences with AI-Assisted Peer Feedback**

The qualitative data reveal both enabling and constraining experiential factors. On the positive side, the interactive chatbot component introduced a dialogic dimension often absent from traditional automated systems (Combéfis, 2022). Students appreciated the ability to ask follow-up questions, supporting the premise that AI can facilitate the sustained engagement that (Carless & Boud, 2018) describe as essential for feedback literacy.

Conversely, the most prominent concern was the speed of the feedback process, with long waiting times negatively affecting motivation and workflow. Since timeliness is a core attribute of effective formative feedback (Shute, 2008), this delay undermines a key expected advantage of AI-based systems and represents a priority for technical optimisation. The grading dimension also surfaced as a point of tension; students perceived the rubric as overly rigid, failing to accommodate alternative but valid coding approaches. This reflects the broader challenge of balancing standardised criteria with the inherent diversity of programming solutions (Keuning et al., 2019).

The frequency analysis of feedback usage behaviour offers an encouraging indicator of feedback uptake: over 41% of students reported often or always incorporating feedback into their revisions, while 44.9% did so sometimes. Only 13.3% reported rarely using the feedback. These patterns suggest that, despite its limitations, the system generated feedback most students found sufficiently relevant and actionable, a meaningful outcome given that feedback uptake remains one of the most persistent challenges in higher education (Winstone & Carless, 2019).

## **5.4 Integration of Mixed-Methods Findings**

The findings of this study demonstrate a clear convergence between student perceptions and their observed technical actions, justifying a mixed-methods approach. While the quantitative data revealed a high mean satisfaction score of 3.93, the qualitative feedback helps explain that this was largely due to the system's ability to provide segmented, manageable feedback addressing functionality and style. This indicates that the MLA-PF framework successfully met the students' perceived needs for specific and manageable formative insights, as prescribed by Shute (2008).

Crucially, the study's mixed-methods design allowed for the triangulation of self-reported behavior with qualitative evidence of code revisions. While 41% of students reported incorporating feedback 'often or always,' the thematic analysis of code changes provided a behavioral context for these claims. Because the qualitative analysis showed students making substantive revisions to logic and control flow rather than superficial corrections—the results suggest that the quantitative uptake frequency reflects genuine pedagogical engagement. This 'double-check' between what students reported and how they modified their code indicates that the framework effectively scaffolds deep conceptual engagement, moving beyond the correctness-first paradigm typical of many feedback tools.

## **5.5 Implications and Limitations**

The findings of this study carry several theoretical, practical, and methodological implications. From a theoretical standpoint, the positive reception of the AI-assisted feedback system, combined with the substantive code revisions students made, suggests that GenAI-supported feedback is a viable and promising approach, provided it is embedded within a pedagogically coherent framework. The MLA-PF framework offers one such model, demonstrating that grounding AI assistance in peer feedback in established educational theories such as Hattie and Timperley's (2007) feedback model and Shute's (2008) formative feedback principles can

produce a system that students perceive as both useful and educationally meaningful, while maintaining the human interaction. This contributes to the growing body of literature calling for theoretically informed AI integration in education, rather than technology-driven approaches that lack pedagogical foundations (Banihashem et al., 2026).

From a practical perspective, the study offers several actionable insights for educators and designers seeking to implement AI-augmented peer feedback in programming courses. First, the recurring student requests for greater depth, personalisation, and speed underscore that technical feasibility alone is insufficient. The quality of AI-generated peer feedback must be continually refined through improved prompt engineering, context-aware language models, and iterative user testing informed by student needs. Second, the finding that students valued the system's approach of withholding direct solutions suggests that AI feedback tools should be designed to scaffold learning rather than provide ready-made answers, a principle that aligns with constructivist pedagogies and the concept of productive failure (Kapur, 2008). Third, the concerns raised about rubric rigidity highlight the importance of designing flexible assessment criteria that can accommodate the diversity of valid programming solutions, particularly in introductory courses where multiple algorithmic approaches may be equally correct.

Moreover, the pilot's scope was limited to the AI-as-peer feedback component; the full MLA-PF cycle, including human peer review coaching (Phase 3), AI-guided dialogic interaction (Phase 4), and the instructor analytics dashboard (Phase 5), was not implemented during this study. Future work should operationalise and evaluate the complete framework to determine whether the integration of human peers alongside AI coaching produces qualitatively different, and potentially richer, feedback experiences than AI-generated feedback alone. In particular, the interplay between AI coaching and human peer interaction represents a critical area for future investigation, as it lies at the heart of the framework's hybrid intelligence design.

Several methodological limitations must also be acknowledged. First, this study is exploratory in nature and was conducted with a single cohort of students in one institutional context, which limits the generalisability of the findings. Second, the absence of a control group restricts the ability to draw causal inferences about the system's impact on learning outcomes; the observed code improvements cannot be attributed solely to the AI feedback without comparison to alternative conditions such as instructor feedback or unassisted revision. Third, the quantitative data relied on self-reported measures of feedback usage and experience, which are subject to response bias and may not fully capture the nuances of students' actual engagement with the system. Fourth, the study did not measure learning outcomes through pre- and post-tests or compare students' grades before and after using the platform, which would have provided stronger evidence of the system's educational impact.

Future research should address these limitations by employing experimental or quasi-experimental designs with control groups, utilising larger and more diverse samples across multiple institutions and programming courses, and incorporating objective measures of learning such as code quality metrics and examination performance. Longitudinal studies that track students' development over an entire semester or across multiple feedback cycles would also provide valuable insights into the sustained impact of AI-augmented peer feedback on programming competence and self-regulated learning. Finally, investigating the experiences and perceptions of instructors who use the platform's dashboard and oversight tools would complement the student-centred findings of this study and contribute to a more comprehensive understanding of the framework's value within the broader educational ecosystem.

## REFERENCES

- Ali, F., Ahmed, A., Alipour, M. A., & Terashima-Marin, H. (2025). Adoption of AI-coding assistants in programming education: Exploring trust and learning motivation through an extended technology acceptance model. *Journal of Computers in Education*. <https://doi.org/10.1007/s40692-025-00375-w>
- Baker, R. S., & Hawn, A. (2022). Algorithmic Bias in Education. *International Journal of Artificial Intelligence in Education*, 32(4), 1052–1092. <https://doi.org/10.1007/s40593-021-00285-9>
- Banihashem, S. K., Noroozi, O., Khosravi, H., Schunn, C. D., & Drachsler, H. (2026a). Pedagogical framework for hybrid intelligent feedback. *Innovations in Education and Teaching International*, 63(2), 554–570. <https://doi.org/10.1080/14703297.2025.2499174>
- Banihashem, S. K., Noroozi, O., Khosravi, H., Schunn, C. D., & Drachsler, H. (2026b). Pedagogical framework for hybrid intelligent feedback. *Innovations in Education and Teaching International*, 63(2), 554–570. <https://doi.org/10.1080/14703297.2025.2499174>
- Barus, O. P., Hidayanto, A. N., Handri, E. Y., Sensuse, D. I., & Yaiprasert, C. (2025). Shaping generative AI governance in higher education: Insights from student perception. *International Journal of Educational Research Open*, 8, 100452. <https://doi.org/10.1016/j.ijedro.2025.100452>

- Boud, D., & Molloy, E. (2013a). Rethinking models of feedback for learning: The challenge of design. *Assessment & Evaluation in Higher Education*, 38(6), 698–712. <https://doi.org/10.1080/02602938.2012.691462>
- Boud, D., & Molloy, E. (2013b). Rethinking models of feedback for learning: The challenge of design. *Assessment & Evaluation in Higher Education*, 38(6), 698–712. <https://doi.org/10.1080/02602938.2012.691462>
- Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., ... Amodei, D. (2020). *Language Models are Few-Shot Learners* (Version 4). arXiv. <https://doi.org/10.48550/ARXIV.2005.14165>
- Carless, D., & Boud, D. (2018a). The development of student feedback literacy: Enabling uptake of feedback. *Assessment & Evaluation in Higher Education*, 43(8), 1315–1325. <https://doi.org/10.1080/02602938.2018.1463354>
- Carless, D., & Boud, D. (2018b). The development of student feedback literacy: Enabling uptake of feedback. *Assessment & Evaluation in Higher Education*, 43(8), 1315–1325. <https://doi.org/10.1080/02602938.2018.1463354>
- Cho, K., & MacArthur, C. (2010). Student revision with peer and expert reviewing. *Learning and Instruction*, 20(4), 328–338. <https://doi.org/10.1016/j.learninstruc.2009.08.006>

- Combéfis, S. (2022). Automated Code Assessment for Education: Review, Classification and Perspectives on Techniques and Tools. *Software, 1*(1), 3–30. <https://doi.org/10.3390/software1010002>
- Crow, T., Luxton-Reilly, A., & Wuensche, B. (2018). Intelligent tutoring systems for programming education: A systematic review. *Proceedings of the 20th Australasian Computing Education Conference*, 53–62. <https://doi.org/10.1145/3160489.3160492>
- Dai, W., Lin, J., Jin, H., Li, T., Tsai, Y.-S., Gašević, D., & Chen, G. (2023). Can Large Language Models Provide Feedback to Students? A Case Study on ChatGPT. *2023 IEEE International Conference on Advanced Learning Technologies (ICALT)*, 323–325. <https://doi.org/10.1109/ICALT58122.2023.00100>
- Darvishi, A., Khosravi, H., Sadiq, S., Gašević, D., & Siemens, G. (2024). Impact of AI assistance on student agency. *Computers & Education, 210*, 104967. <https://doi.org/10.1016/j.compedu.2023.104967>
- Douce, C., Livingstone, D., & Orwell, J. (2005). Automatic test-based assessment of programming: A review. *Journal on Educational Resources in Computing, 5*(3), 4. <https://doi.org/10.1145/1163405.1163409>
- Estévez-Ayres, I., Callejo, P., Hombrados-Herrera, M. Á., Alario-Hoyos, C., & Delgado Kloos, C. (2025). Evaluation of LLM Tools for Feedback Generation in a Course on Concurrent Programming. *International Journal of Artificial Intelligence in Education, 35*(2), 774–790. <https://doi.org/10.1007/s40593-024-00406-0>

- Fang, Y., & Zhang, S. (2025). Review of *AI Competency Framework for Teachers*, Fengchun Miao and Mutlu Cukurova, 2024. *Journal of China Computer-Assisted Language Learning*, 5(2), 317–326. <https://doi.org/10.1515/jccall-2025-0018>
- Fu, Q., Zhao, Y., Jia, Z., & Zheng, Y. (2025). Large Language Models (LLMs) in Programming Learning: The Current Research State and Agenda. *IEEE Transactions on Learning Technologies*, 18, 942–961. <https://doi.org/10.1109/TLT.2025.3622043>
- Gálvez, J., Guzmán, E., & Conejo, R. (2009a). A blended E-learning experience in a course of object oriented programming fundamentals. *Knowledge-Based Systems*, 22(4), 279–286. <https://doi.org/10.1016/j.knosys.2009.01.004>
- Gálvez, J., Guzmán, E., & Conejo, R. (2009b). A blended E-learning experience in a course of object oriented programming fundamentals. *Knowledge-Based Systems*, 22(4), 279–286. <https://doi.org/10.1016/j.knosys.2009.01.004>
- Guo, K. (2024). EvaluMate: Using AI to support students' feedback provision in peer assessment for writing. *Assessing Writing*, 61, 100864. <https://doi.org/10.1016/j.asw.2024.100864>
- Gupta Priyanka, P., & Mehrotra, D. (2022). Objective Assessment in Java Programming Language Using Rubrics. *Journal of Information Technology Education: Innovations in Practice*, 21, 155–173. <https://doi.org/10.28945/5040>
- Hattie, J., & Timperley, H. (2007). The Power of Feedback. *Review of Educational Research*, 77(1), 81–112. <https://doi.org/10.3102/003465430298487>

- Hmelo-Silver, C. E., Duncan, R. G., & Chinn, C. A. (2007). Scaffolding and Achievement in Problem-Based and Inquiry Learning: A Response to Kirschner, Sweller, and Clark (2006). *Educational Psychologist*, 42(2), 99–107. <https://doi.org/10.1080/00461520701263368>
- Holmes, W., Bialik, M., & Fadel, C. (2019). *Artificial intelligence in education: Promises and implications for teaching and learning*. Center for Curriculum Redesign.
- Hulls, C. C. W., Neale, A. J., Komalo, B. N., Petrov, V., & Brush, D. J. (2005). Interactive Online Tutorial Assistance for a First Programming Course. *IEEE Transactions on Education*, 48(4), 719–728. <https://doi.org/10.1109/TE.2005.858400>
- Ihantola, P., Ahoniemi, T., Karavirta, V., & Seppälä, O. (2010). Review of recent systems for automatic assessment of programming assignments. *Proceedings of the 10th Koli Calling International Conference on Computing Education Research*, 86–93. <https://doi.org/10.1145/1930464.1930480>
- Kapur, M. (2008). Productive Failure. *Cognition and Instruction*, 26(3), 379–424. <https://doi.org/10.1080/07370000802212669>
- Kasneci, E., Sessler, K., Küchemann, S., Bannert, M., Dementieva, D., Fischer, F., Gasser, U., Groh, G., Günemann, S., Hüllermeier, E., Krusche, S., Kutyniok, G., Michaeli, T., Nerdel, C., Pfeffer, J., Poquet, O., Sailer, M., Schmidt, A., Seidel, T., ... Kasneci, G. (2023). ChatGPT for good? On opportunities and challenges of large language models for education.

*Learning and Individual Differences*, 103, 102274.

<https://doi.org/10.1016/j.lindif.2023.102274>

Kerman, N. T., Banihashem, S. K., Karami, M., Er, E., Van Ginkel, S., & Noroozi, O. (2024). Online peer feedback in higher education: A synthesis of the literature. *Education and Information Technologies*, 29(1), 763–813.

<https://doi.org/10.1007/s10639-023-12273-8>

Keuning, H., Jeurig, J., & Heeren, B. (2018). A Systematic Literature Review of Automated Feedback Generation for Programming Exercises. *ACM Transactions on Computing Education*, 19(1), 1–43.

<https://doi.org/10.1145/3231711>

Keuning, H., Jeurig, J., & Heeren, B. (2019). A Systematic Literature Review of Automated Feedback Generation for Programming Exercises. *ACM Transactions on Computing Education*, 19(1), 1–43.

<https://doi.org/10.1145/3231711>

Liu, N.-F., & Carless, D. (2006). Peer feedback: The learning element of peer assessment. *Teaching in Higher Education*, 11(3), 279–290.

<https://doi.org/10.1080/13562510600680582>

Lodge, J. M., De Barba, P., & Broadbent, J. (2024). Learning with Generative Artificial Intelligence Within a Network of Co-Regulation. *Journal of University Teaching and Learning Practice*, 20(7).

<https://doi.org/10.53761/m2v9an32>

Marwan, S., Akram, B., Barnes, T., & Price, T. W. (2022). Adaptive Immediate Feedback for Block-Based Programming: Design and Evaluation. *IEEE*

*Transactions on Learning Technologies*, 15(3), 406–420.

<https://doi.org/10.1109/TLT.2022.3180984>

Messer, M., Brown, N. C. C., Kölling, M., & Shi, M. (2024). Automated Grading and Feedback Tools for Programming Education: A Systematic Review.

*ACM Transactions on Computing Education*, 24(1), 1–43.

<https://doi.org/10.1145/3636515>

Molenaar, I. (2022). Towards hybrid HUMAN-AI learning technologies. *European*

*Journal of Education*, 57(4), 632–645. <https://doi.org/10.1111/ejed.12527>

Narciss, S., & Huth, K. (n.d.). *How to design informative tutoring feedback for multi-media learning*.

Nicol, D. (2010). From monologue to dialogue: Improving written feedback

processes in mass higher education. *Assessment & Evaluation in Higher*

*Education*, 35(5), 501–517. <https://doi.org/10.1080/02602931003786559>

Nicol, D. J., & Macfarlane-Dick, D. (2006). Formative assessment and self-

regulated learning: A model and seven principles of good feedback practice.

*Studies in Higher Education*, 31(2), 199–218.

<https://doi.org/10.1080/03075070600572090>

Ott, C., Robins, A., & Shephard, K. (2016). Translating Principles of Effective

Feedback for Students into the CS1 Context. *ACM Transactions on*

*Computing Education*, 16(1), 1–27. <https://doi.org/10.1145/2737596>

Ouyang, F., Guo, M., Zhang, N., Bai, X., & Jiao, P. (2024). Comparing the Effects of Instructor Manual Feedback and ChatGPT Intelligent Feedback on

Collaborative Programming in China's Higher Education. *IEEE*

*Transactions on Learning Technologies*, 17, 2173–2185.

<https://doi.org/10.1109/TLT.2024.3486749>

Paiva, J. C., Leal, J. P., & Figueira, Á. (2022). Automated Assessment in Computer Science Education: A State-of-the-Art Review. *ACM Transactions on Computing Education*, 22(3), 1–40. <https://doi.org/10.1145/3513140>

Panadero, E., Brown, G. T. L., & Strijbos, J.-W. (2016). The Future of Student Self-Assessment: A Review of Known Unknowns and Potential Directions. *Educational Psychology Review*, 28(4), 803–830.

<https://doi.org/10.1007/s10648-015-9350-2>

Pintrich, P. R. (2000). The Role of Goal Orientation in Self-Regulated Learning. In *Handbook of Self-Regulation* (pp. 451–502). Elsevier.

<https://doi.org/10.1016/B978-012109890-2/50043-3>

Prather, J., Reeves, B., Leinonen, J., MacNeil, S., Randrianasolo, A. S., Becker, B., Kimmel, B., Wright, J., & Briggs, B. (2024). *The Widening Gap: The Benefits and Harms of Generative AI for Novice Programmers* (Version 1). arXiv. <https://doi.org/10.48550/ARXIV.2405.17739>

Qian, Y., & Lehman, J. (2017). Students' Misconceptions and Other Difficulties in Introductory Programming: A Literature Review. *ACM Transactions on Computing Education*, 18(1), 1–24. <https://doi.org/10.1145/3077618>

Sadler, D. R. (1989). Formative assessment and the design of instructional systems. *Instructional Science*, 18(2), 119–144. <https://doi.org/10.1007/BF00117714>

- Selwyn, N. (2024). On the Limits of Artificial Intelligence (AI) in Education. *Nordisk Tidsskrift for Pedagogikk Og Kritik*, 10(1).  
<https://doi.org/10.23865/ntpk.v10.6062>
- Shute, V. J. (2008a). Focus on Formative Feedback. *Review of Educational Research*, 78(1), 153–189. <https://doi.org/10.3102/0034654307313795>
- Shute, V. J. (2008b). Focus on Formative Feedback. *Review of Educational Research*, 78(1), 153–189. <https://doi.org/10.3102/0034654307313795>
- Sweller, J. (1988). Cognitive Load During Problem Solving: Effects on Learning. *Cognitive Science*, 12(2), 257–285.  
[https://doi.org/10.1207/s15516709cog1202\\_4](https://doi.org/10.1207/s15516709cog1202_4)
- Topping, K. J. (2005). Trends in Peer Learning. *Educational Psychology*, 25(6), 631–645. <https://doi.org/10.1080/01443410500345172>
- Topping, K. J., Gehringer, E., Khosravi, H., Gudipati, S., Jadhav, K., & Susarla, S. (2025). Enhancing peer assessment with artificial intelligence. *International Journal of Educational Technology in Higher Education*, 22(1), 3.  
<https://doi.org/10.1186/s41239-024-00501-1>
- Wang, T., Zhou, N., & Chen, Z. (2024). *Enhancing Computer Programming Education with LLMs: A Study on Effective Prompt Engineering for Python Code Generation* (Version 1). arXiv.  
<https://doi.org/10.48550/ARXIV.2407.05437>
- Winstone, N., & Carless, D. (2019a). *Designing Effective Feedback Processes in Higher Education: A Learning-Focused Approach* (1st ed.). Routledge.  
<https://doi.org/10.4324/9781351115940>

- Winstone, N., & Carless, D. (2019b). *Designing Effective Feedback Processes in Higher Education: A Learning-Focused Approach* (1st ed.). Routledge.  
<https://doi.org/10.4324/9781351115940>
- Wood, D., Bruner, J. S., & Ross, G. (1976). THE ROLE OF TUTORING IN PROBLEM SOLVING\*. *Journal of Child Psychology and Psychiatry*, 17(2), 89–100. <https://doi.org/10.1111/j.1469-7610.1976.tb00381.x>
- Zhai, X. (2022). ChatGPT User Experience: Implications for Education. *SSRN Electronic Journal*. <https://doi.org/10.2139/ssrn.4312418>
- Zimmerman, B. J. (2000). Attaining Self-Regulation. In *Handbook of Self-Regulation* (pp. 13–39). Elsevier. <https://doi.org/10.1016/B978-012109890-2/50031-7>
- Reeves, T. C. (2006). Design research from a technology perspective. In J. van den Akker, K. Gravemeijer, S. McKenney, & N. Nieveen (Eds.), *Educational design research* (pp. 52–66). Routledge.  
<https://doi.org/10.4324/9780203088364-13>

## APPENDICES

### A. Student Questionnaire

How often did you take into consideration <sup>\*</sup> feedback comments (including the chatbot) when you revised your code?

Never

Rarely

Sometimes

Often

Always

[Sonraki](#) [Formu temizle](#)

Share your experience with feedback received (including chatbot).

I liked the review session. \*

- Strongly Agree
- Agree
- Neutral
- Disagree
- Strongly Disagree

I found the feedback comments (including the chatbot) in the review session *useful*. \*

- Strongly Agree
- Agree
- Neutral
- Disagree
- Strongly Disagree



The feedback comments (including the chatbot) helped me to enrich the functionality of my program.

- Strongly Agree
- Agree
- Neutral
- Disagree
- Strongly Disagree



Feedback comments (including the chatbot) helped me to improve the organization of my code.

- Strongly Agree
- Agree
- Neutral
- Disagree
- Strongly Disagree



Feedback comments (including the chatbot) helped me to improve the style (including syntax and conventions) of my code.

- Strongly Agree
- Agree
- Neutral
- Disagree
- Strongly Disagree



I benefited from the feedback comments (including the chatbot) .

- Strongly Agree
- Agree
- Neutral
- Disagree
- Strongly Disagree



How often did you take into consideration <sup>\*</sup> feedback comments (including the chatbot) when you revised your code?

- Never
- Rarely
- Sometimes
- Often
- Always

Sonraki

Formu temizle

Please respond to the following items to indicate how much the feedback supported your learning and improvement.

My code became better after revisions. \*

- Strongly Agree
- Agree
- Neutral
- Disagree
- Strongly disagree

After the revision, the functionality of my code became richer. \*

- Strongly Agree
- Agree
- Neutral
- Disagree
- Strongly disagree



Revisions helped me improve my code. \*

- Strongly Agree
- Agree
- Neutral
- Disagree
- Strongly disagree

I hope that my teacher will continue to use \*  
this **feedback approach** in the programming  
course next year.

- Strongly Agree
- Agree
- Neutral
- Disagree
- Strongly disagree

[Geri](#)

[Sonraki](#)

[Formu temizle](#)

After the revision, the organization of my code became better. \*

- Strongly Agree
- Agree
- Neutral
- Disagree
- Strongly disagree

After the revision, the style (including syntax and conventions) of my code improved. \*

- Strongly Agree
- Agree
- Neutral
- Disagree
- Strongly disagree

Revisions helped me improve my code. \*

- ? Strongly Agree

**In this section, you will be asked to share your experience with receiving and using AI-driven feedback and peer feedback. We want to understand what you think and feel about both methods and whether you find them useful. There are no right or wrong answers in this survey, so please feel free to speak openly about your experiences when answering these questions.**

What do you consider to be the primary **strengths** and **weaknesses** of the feedback process within Peercept? \*

Yanıtınız

How would you evaluate the **quality** of the feedback (including the chatbot) in terms of technical accuracy and helpfulness? \*

Yanıtınız

How **clear** and **understandable** was the feedback you received (including the chatbot)? Please provide examples of unclear or ambiguous comments if applicable. ? \*

How **clear** and **understandable** was the feedback you received (including the chatbot)? Please provide examples of unclear or ambiguous comments if applicable. \*

Yanıtınız

Describe specific changes you made to your code based on the peer feedback. If you chose not to make changes, please explain why.

Yanıtınız

If you could change one aspect of the peer feedback activity (either the platform or the process), what would it be? \*

Yanıtınız

Geri

Gönder

Formu temizle