

How Do Students Perceive Their Teammates? A Qualitative Study of Peer Evaluation in SE Course Projects

Sahithya Gandu
Colorado State University
Fort Collins, USA
sahithya.gandu@colostate.edu

Anurag Vemula
Colorado State University
Fort Collins, USA
anurag.vemula@colostate.edu

Satyanarayanachowdary
Kadiyala
Colorado State University
Fort Collins, USA
satyanarayanachowdary.kadiyala@colostate.edu

Márcia Moraes
Colorado State University
Fort Collins, USA
marcia.moraes@colostate.edu

Sudipto Ghosh
Colorado State University
Fort Collins, USA
sudipto.ghosh@colostate.edu

Fabio Santos
Colorado State University
Fort Collins, USA
fabio.deabreusantos@colostate.edu

Bianca Trinkenreich
Colorado State University
Fort Collins, USA
bianca.trinkenreich@colostate.edu

Abstract

Context: Collaborative work is central to software engineering (SE) education, yet students often encounter unequal participation that shapes their perceptions of fairness and reliability. Prior research has classified teammates as Couch Potatoes (minimal contributors), Hitchhikers (credit-sharers with little effort), Lone Wolves (independent workers), and Good Teammates (active, equitable collaborators). **Goal:** We investigated how students describe these four teammate types' behaviors in SE course projects to uncover the perceived characteristics defining each type. **Method:** We qualitatively analyzed 570 peer evaluations from a senior-level SE project course, focusing on open-text justifications of teammate labels. **Results:** Our findings refine existing definitions by grounding them in students' language and capturing both negative and positive aspects. Good Teammates were described as proactive, responsive, skilled, punctual, and helpful. Couch Potatoes were seen as consistently disengaged. Hitchhikers, while largely negative, occasionally sought help or contributed, suggesting a spectrum rather than a binary role. Lone Wolves paired isolation and dominance with valuable, high-volume contributions. **Conclusion:** These insights offer a more nuanced, empirically grounded view of teammate behaviors, and a reusable codebook of 28 behavioral characteristics organized into five high-level categories, guiding instructors in designing peer evaluation rubrics and interventions that foster balanced participation.

Keywords

Teamwork, Good Teammate, Couch Potato, Hitchhiker, Lone Wolf

ACM Reference Format:

Sahithya Gandu, Anurag Vemula, Satyanarayanachowdary Kadiyala, Márcia Moraes, Sudipto Ghosh, Fabio Santos, and Bianca Trinkenreich. 2026. How Do Students Perceive Their Teammates? A Qualitative Study of Peer Evaluation in SE Course Projects. In *34th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (FSE Companion '26)*, July 05–09, 2026, Montreal, QC, Canada. ACM, New York, NY, USA, 11 pages. <https://doi.org/10.1145/3803437.3805781>

1 Introduction

Software development is rarely a solitary activity; developers work in teams where coordination, communication, and shared decision-making are essential for success [25]. In Software Engineering (SE) classes, team-based projects provide students opportunities to develop both technical competence and teamwork skills necessary for professional practice [15].

Despite these benefits, teamwork in student projects frequently suffers from unequal participation. Some students contribute very little or avoid responsibility, leaving peers to complete most of the work, while others work almost entirely alone, potentially redoing teammates' contributions rather than integrating them. Both extremes can undermine team cohesion, suppress shared ownership, and diminish the learning value of team projects [14, 15]. Such imbalances can erode trust, create conflicts, reduce shared ownership of outcomes, and ultimately diminish the learning value of the project experience. Conversely, teams benefit from peers who motivate others, coordinate work, and help maintain momentum toward collective goals. Prior research has shown that these participation profiles influence workload distribution, communication frequency, and overall team performance [2, 14, 20–22].

Previous research has characterized four common teammate types: *Couch Potatoes* contribute minimally, often relying on others to carry the workload [20]; *Hitchhikers* also contribute little but take advantage of the team's effort to benefit without sharing responsibility [20]; *Lone Wolves* work independently, sometimes



This work is licensed under a Creative Commons Attribution 4.0 International License. *FSE Companion '26, Montreal, QC, Canada*
© 2026 Copyright held by the owner/author(s).
ACM ISBN /2026/07
<https://doi.org/10.1145/3803437.3805781>

struggling to integrate with the team [2]; and *Good Teammates* contribute actively and foster a collaborative environment [22].

In this paper, we investigate how students evaluate one another during a team-based software engineering project by analyzing their open-text justifications for assigning these teammate labels. Our goal is to characterize the perceived attributes of each type and uncover how these perceptions reflect the social dynamics that shape learning experiences in SE teams.

Research Question. *What defining characteristics do students associate with couch potatoes, hitchhikers, lone wolves, and good teammates in SE course projects?*

Our analysis of 570 peer evaluations revealed a rich and nuanced picture of how students perceive their teammates' contributions in team-based SE projects. We found 28 behavioral characteristics and grouped them into five behavioral categories: COMMUNICATION, ENGAGEMENT, TECHNICAL QUALITY, TIMING, and COLLABORATION, capturing both positive and negative patterns of teamwork. Our analysis refines and expands prior conceptualizations of teammate types [21] by grounding them in students' own language from a modern software engineering context. Good Teammates emerged as exemplars of collaboration, characterized by proactive initiative, responsiveness, punctuality, technical expertise, and helpfulness, behaviors that sustained team momentum and mirrored prior work highlighting the value of team citizenship. Couch Potatoes, similar to Oakley et al. [21]'s description of non-contributing members, were consistently portrayed as passively disengaged, with peers reporting repeated patterns of minimal work and lack of initiative across sprints. Hitchhikers, while traditionally framed as free-riders [21], were here described with greater nuance: students noted strategic minimalism (e.g., merging just a few PRs or completing one or two issues) but occasionally acknowledged genuine attempts to contribute or ask for help, suggesting a spectrum rather than a binary classification. Lone Wolves were perceived as both a risk and an asset: they produced large amounts of work. Still, they did so in isolation, sometimes dominating decisions and creating integration challenges, yet their output was valued as essential for project completion. By capturing both negative and positive aspects of typically problematic types, our results offer a more granular and empirically grounded picture of teammate behaviors, highlighting the trade-offs teams navigate between productivity and collaboration.

This paper makes the following contributions:

- Empirical characterization of teammate types through qualitative analysis of how students justify labeling peers as Good Teammate, Couch Potato, Hitchhiker, or Lone Wolf across multiple project sprints in a SE team-project.
- Codebook of 28 behavioral characteristics organized into five high-level categories, offering a reusable lens for future research.
- Insights for educational practice about concrete, observable signals of disengagement, inequity, and leadership behaviors, which can inform more nuanced peer-evaluation rubrics, early-warning systems, and pedagogical interventions that promote balanced participation.

We present the context in which this research was conducted in Section 2. We introduce our research design in Section 3 and the study results in Section 4. Section 5 discusses the results and

implications of our results, followed by related work in Section 6, limitations, and conclusions in Sections 7 and 8.

2 Course Context

The study was performed using data from the Spring 2024 semester of a senior-level computer science course focused on Software Testing at <omitted due to blind review>, which had 24 teams comprised of 118 students (89 in-person and 29 online).

The lectures focus on the fundamentals of systematic testing using various formalisms such as input space partitioning, graph coverage, mutation analysis, state-based testing, and testing workflows. The project provides hands-on training in test-driven development with various testing tools while using models to support various testing and implementation activities in a team of five students.

2.0.1 Course Project. In Sprint 1 (P1), students used input space partitioning to create tests, implement core business logic, and improve code coverage through graph-based coverage principles. In Sprint 2 (P2), they enhanced the P1 test suite's mutation score obtained using Pitest [5], applied static analysis tools (PMD¹ and SpotBugs²), refactored code, and used automatic test generation tools (EvoSuite [8] and Randoop [23]). They also implemented a REST controller and mocked a database. In Sprint 3 (P3), students developed the front-end with JavaScript and React and wrote comprehensive test cases using Fittness and Selenium.

2.0.2 Team Creation. Students in this course were split into teams to work together on a SE project throughout the 16-week semester broken into three 5-week sprints. Teams consisted of about five students each.

Rather than randomly assigning students to teams, the instructor intentionally formed diverse and balanced groups. These teams were designed to reflect a mix of genders, attendance model (in-person or online), computer science skills (based on accumulated course credits), and prior experience (assessed through a questionnaire on internships, coursework, and familiarity with specific languages, tools, technologies, and processes). To give students some control over their team dynamics, each student was allowed to select one partner to be in the same group, enabling them to work with a known and trusted collaborator. Given the inclusion of online students—who often have different schedules and time zones compared to in-person students—teams were limited to no more than two remote participants. Furthermore, based on research showing that women tend to feel more motivated and confident when paired with peers of the same gender [26], teams were structured to include at least two women, ensuring that women had the opportunity for peer parity.

2.0.3 Teammate Types. In addition to the initial survey used to assess students' experiences for the team formation, the instructor conducted three online surveys for peer evaluation, one at the end of each sprint. The class started with 124 students, and after six students dropped out, it ended with 118 students. Not all students answered all the surveys, and some students changed to another group. Based on previous research on peer evaluation through teammate types [20], the questionnaire had a question about how

¹https://docs.pmd-code.org/latest/pmd_rules_java.html

²<https://spotbugs.readthedocs.io/en/latest/bugDescriptions.html>

the student perceived themselves and each of their teammates' contribution levels, with four options to select: Good Teammate, Lone Wolf, Couch Potato, or Hitchhiker, and an optional open-text field to justify the evaluation. The instructor provided an in-class explanation of teammate types to ensure clarity.

3 Research Design

In this section, we outline our methodology for addressing the research question proposed in Section 1.

3.1 Dataset

For each of the three sprints, the instructor collected the peer evaluations that included a classification of each teammate and the person themselves as Couch Potato, Hitchhiker, Lone Wolf, or Good Teammate, along with a justification for the selection. Not all the students answered the survey and from those who did, not everyone justified. We removed the justifications that had no meaningful text (e.g. “does not apply”, “not applicable”) and ended up with 570 evaluations to analyze, which we present in Table 1.

Table 1: Distribution of Justifications Analyzed per Teammate Type per Sprint

	Sprint 1	Sprint 2	Sprint 3
Teammate Types	214 justifications	178 justifications	178 justifications
Good Teammate	150	112	137
Couch Potato	27	39	19
Hitchhiker	27	13	15
Lone Wolf	10	14	7

3.2 Data Analysis

To answer our research question, we qualitatively analyzed the open-ended peer evaluation justifications collected over three sprints of a team-based software engineering course project.

Three authors conducted a collaborative qualitative analysis of the open-ended responses over a two-month period, meeting weekly throughout the process. Following an inductive approach, they applied open coding [17] to identify and organize students' stated justifications for their peer evaluations. Throughout the analysis, codes and emerging behavioral categories were iteratively discussed and refined. The resulting codebook and category structure were then reviewed with a fourth author with extensive experience in qualitative research. Disagreements were resolved through discussion until consensus was reached, following a negotiated agreement approach [10] approach. A fourth author, who has a high degree of experience in qualitative analysis studies, reviewed the results. The outcome was a set of higher-level categories as cataloged in our codebook [9].

The study was approved by the Institutional Review Board of the university. All data were deidentified so that each student was assigned a random number associated with their team. The data was collected in three sprints. We represent the students' IDs as S1 to S124.

4 Study Results

We identified five categories of peer-evaluation characteristics (Engagement, Communication, Team Dynamics, Technical Quality, and Timing) depicted in Fig. 1. Inside each category, negatively perceived characteristics are presented in blue and positively perceived ones are presented in yellow. Each characteristic code indicates, inside it, the teammate type(s) in our dataset for which students provided justifications reflecting that behavior. Table 1 reports the number of students whose evaluations aligned with each teammate type.

In the following sections, we elaborate on these findings, discussing each category of peer-evaluation behavioral characteristics in detail.

4.1 Communication

Effective communication among team and group members is essential to a collaborative SE effort [27]. Our analysis identified three communication-related behavior patterns, including two negative patterns (Lack of Communication and Poor Responsiveness) and one positive pattern (Responsive Communication).

Table 2: Communication behavioral category of teammates

Code	Representative Examples
Lack of Communication	“Never really communicated (S66)”; “Didn’t communicate [with the] team at all (S66)”
Poor Responsiveness	“Viewed and ignored communication attempts (S78)”; “Would not respond for 3–4 days (S119)”
Responsive Communication	“Having a responsive communication (S5)”; “Organiz[ing] team communication (S5)”

Lack of Communication. Was mentioned by students when teammates demonstrated little to no interaction across the project’s duration, leaving others without necessary updates or awareness of their progress. This behavior was found in *Couch Potato* and *Lone Wolf* evaluations.

For *Couch Potatoes*, lack of communication happened when the teammate was away from the project, “barely communicat[ing] during sprint (S4, S37, S91). Others emphasized more severe disengagement, noting that the teammate “never really communicated (S66)” or “didn’t communicate [with the] team at all” (S66). Several students pointed out a complete absence of interaction on collaboration platforms, reporting that they were “completely silent [on] [Microsoft] Teams and GitHub” (S67) and that there was an “almost complete lack [of] communication” (S27) “with teammates” (S27). This lack of participation also manifested in visible inactivity, as peers noted that they “didn’t see [those teammates] type anything [in] chats” (S89) or that they were “not even online” (S89).

For *Lone Wolves*, lack of communication happened when teammates were working on the project, but in isolation, “without much discussion with the rest of the team” (S32), were “inconsistent with communicating” (S67), or “didn’t communicate with team very much on what [they were] working on” (S32). Other comments highlighted that teammates communicated outcomes after implementation, such as “[not being] in contact with the team for most of the sprint” (S32) or not using the official channel (S32).

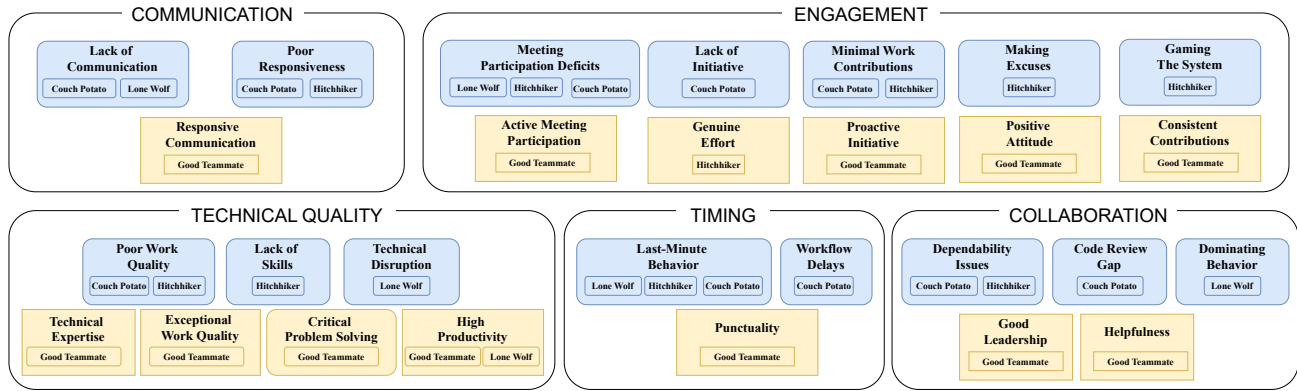


Figure 1: The behavioral characteristics of teammates in SE group-based class project as reported by students in our study. Boxes represent the teammate types from whose evaluations the communication codes emerged. Blue and rounded denotes negative communication aspects, whereas yellow and non-rounded denotes positive ones.

Poor Responsiveness. Was mentioned by students when teammates failed to acknowledge or reply to communication attempts, creating coordination gaps and disrupting project workflows. Unlike Lack of Communication, where students rarely initiate interactions, POOR RESPONSIVENESS was perceived when messages are seen but ignored, leaving teammates without timely feedback. This behavior was found in *Couch Potato*'s and *Hitchhiker*'s evaluations.

For *Couch Potatoes*, Poor Responsiveness reflected passive disengagement. Students described peers who “viewed and ignored communication attempts” (S78), or students who “would not respond for 3–4 days” (S119), overall “frequently [not] respond[ing] messages” (S119), or communicates only “after being pinged” (S27). They were considered as being “hard to contact” (S119) and “never respond[ing] a scheduling demo” (S89), which is related to the deliverable of the project.

For *Hitchhikers*, Poor Responsiveness suggested Hitchhikers used to delay or avoid task commitments while maintaining plausible deniability. Students noted that these teammates “had no response until the last day” (S108) or had “low communication until the weekend before the sprint was due” (S9).

Responsive Communication. Was mentioned by students when teammates provided timely and appropriate responses to communication attempts, acknowledging messages and keeping discussions moving. This behavior creates reliability in team coordination, ensuring that information requests receive prompt feedback and preventing delays. This behavior was found in *Good Teammate* evaluations. Students described peers as “having a responsive communication” (S5) when they did an active chat discussing, also giving fast responses, sharing notes with teammates, and acting as a leader and “organiz[ing] team communication” (S5).

4.2 Engagement

In software engineering projects, engagement reflects whether teammates consistently participate, contribute, and take initiative, which are critical behaviors that signal commitment to collective goals [27]. Our analysis identified ten engagement-related behavior

patterns, including five negatives (Meeting Participation Deficits, Lack of Initiative, Minimal Work Contribution, Making Excuses, and Gaming the System) and five positives (Active Meeting Participation, Genuine Effort, Proactive Initiative, Positive Attitude, and Consistent Contributions). We summarize our findings organized in these higher-level categories in Table 3.

Table 3: Engagement Behavioral Codes

Code	Representative Examples
Meeting Participation Deficits	“Rarely showed up to meetings (S10)”; “Practically absent (S5)”
Lack of Initiative	“Didn’t take initiative (S17)”; “Did not seem interested taking issues (S42)”
Minimal Work Contribution	“Minimal work (S4)”; “Contributed little (S25, S33)”
Making Excuses	“Always [having a] poor excuse for not contributing (S44)”; “Pattern of avoidance and excuse-making (S100)”
Gaming the System	“Aware [they] can only get docked twenty points (S21)”; “Closing all existing issues and creating them from scratch (S9)”
Genuine Effort	“Did try (S45)”; “Tried hard to make an impact (S48)”
Proactive Initiative	“First [ones who] started working (S32)”; “Immediate sprint start (S107, S120)”
Positive Attitude	“Well-liked positive morale (S48)”; “Always happy [to] help review (S119)”
Consistent Contributions	“Contributed all semester (S23)”; “Only consistent worker (S106)”
Active Meeting Participation	“Attend every meeting (S9, S31)”; “Shared notes every meeting (S48)”

Meeting participation deficits. The students mentioned when their teammates did not attend or attended some meetings, but did not participate meaningfully in discussion or planning. This pattern includes situations where presence was noted but contributions were minimal or nonexistent, leading to missed opportunities for coordination and joint decision-making. This behavior was found in *Lone Wolf*, *Couch Potato*, and *Hitchhiker* evaluations.

For *Lone Wolves*, students described teammates who “rarely showed up to meetings” (S10) or not in a “consistent” way (S10). Others observed peers who did the work but not met the team (S19) nor show up to classes (S32).

For *Couch Potatoes*, the meeting participation deficits were observed as “often quiet in meetings” (S31), (S26), or “miss[ing] meetings” (S42, S4, S37, S51, S79).

For *Hitchhikers*, teammates observed meeting participation deficits when they were “practically absent” (S5), “always super late for meetings” (S44), or exhibiting a behavior of “ghost[ing]” (S93, S120) the other teammates, almost seeming the student “may have dropped the course” (S120).

Lack of Initiative. Was mentioned when teammates failed to proactively take ownership without explicit direction. This behavior was found in *Couch Potato* evaluations. Students noted that peers “didn’t take initiative” (S17) and “did not seem interested taking issues” (S42), not “help[ing] [the] project at all” (S119).

Minimal Work Contribution. Was mentioned when teammates completed just enough work to remain part of the team but contributed significantly less than their peers. This behavior was found in *Couch Potato*’s and *Hitchhiker*’s evaluations.

For *Couch Potatoes*, minimal work contribution was perceived as sustained disengagement, with peers emphasizing the very low volume of work produced across multiple sprints. Students described these teammates as doing always “minimal work” (S4), often “contribut[ing] little” (S25, S33). Some justifications highlighted almost complete absence, saying the teammate did almost (S106) or “absolutely nothing” during the entire semester (S119).

For *Hitchhikers*, minimal work contribution was framed not just as low output but as selective engagement that appeared intentional. Students noted that these peers “completed one or two issues” (S5), “merge[d] a few pull requests” (S31), suggesting a pattern of pretending participation and doing the bare minimum to maintain credit, as the grade is calculated using pull request and merge volumes.

Making Excuses. Was mentioned by students when teammates repeatedly justified their low contributions or missed deadlines without taking accountability. This behavior was found in *Hitchhiker* evaluations.

Students described those teammates as “always [having a] poor excuse for not contributing” (S44), sometimes tied to personal commitments, such as the “excuse of being busy with fraternity” (S43). Some students noted a “pattern of avoidance and excuse-making” (S100), where absence was used to evade responsibility as “using absence as blanket excuse” (S100) and even shifting work to others by “suggesting that others complete the work instead” (S100).

Gaming the System. Was mentioned by students when teammates deliberately exploited grading rules, penalties, or contribution tracking mechanisms to reduce their workload while still maintaining team membership. This behavior was found in *Hitchhiker*’s evaluations.

Students described peers who were “aware [they] can only get docked twenty points” (S21) and even “willing to take that penalty” (S21) rather than contributing equitably. Others documented cases where teammates chose to “take the less 10 point adjustment without doing too much work” (S5) or who admitted to “game the system”

(S9). In some cases, the students observed manipulation of the contribution tracking, such as “closing all existing issues and creating them from scratch” (S9) to distort participation metrics.

Genuine Effort. Was mentioned by students when teammates made authentic attempts to contribute despite overall reduced engagement levels. This behavior was found in *Hitchhiker* evaluations. Students mentioned cases where peers did not abandon the project, instead, they “did try” (S45), “tried hard to make an impact” (S48), and “asked for help when needed” (S42). Others emphasized follow-through when the teammate “ensured taken tasks got done within reasonable time” (S42) and recognized smaller contributions, noting that they “tried to give support” (S5) to others.

Proactive Initiative. Was mentioned by students when teammates proactively began tasks, commits, or sprint work without waiting for direction, often helping set the pace for the entire team. This behavior was found in *Good Teammate* evaluations.

Students praised peers who showed early and consistent initiative, noting their “initiative help the team” (S32). Others highlighted that these teammates were often the “first [ones who] started working” (S32), setting the pace for the group. Students also emphasized specific early contributions, such as giving an “immediate sprint start” (S107, S120) and completing “early setup tasks” (S107). Initiative was also recognized in concrete actions like solving issues (S5), creating new components (S42), developing UI features (S103), and even supporting collaboration by helping to “figure [the] first merge” (S77).

Positive Attitude. Was mentioned by students when teammates maintained a constructive, encouraging, and supportive approach to collaboration, which elevated team morale and helped the group work through challenges. This behavior was found in *Good Teammate* evaluations.

Students emphasized the positive influence these teammates had on group dynamics, describing them as providing “well-liked positive morale” (S48), being an “inspiration [about how they] perform” (S36) as they were true “team players” (S93). This positivity was explained as practical collaboration behaviors, with peers noting that they were always “happy [to] help review” (S119).

Consistent contributions. Were perceived when teammates consistently demonstrated dedication, as someone who “contributed all semester” (S23), being many times the “only consistent worker” (S106). Students noticed those teammates brought a solid and consistent project work throughout (S47, S21). This behavior was also noted as “consistently not procrastinating [tasks]” (S116). This behavior was found in *Good Teammate* evaluations.

Active Meeting Participation. Was mentioned by students when teammates consistently attended team meetings and contributed meaningfully to discussions and problem-solving. The opposite of Meeting Participation Deficits, which reflect low engagement despite presence, this pattern captures both attendance and substantive participation. This behavior was found in *Good Teammate* evaluations. Students described peers who “attend every meeting” (S9, S31) or “almost every meeting” (S31). Besides being there, these teammates participated, “communicated and shared notes every meeting” (S48).

4.3 Technical Quality

. High-quality software artifacts reduce errors and rework [27]. Our analysis identified seven technical quality-related behavior patterns, including three negative patterns (Poor Work Quality, Lack of Skills, and Technical Disruption) and four positive patterns (Technical Expertise, Exceptional Work Quality, Critical Problem Solving, and High Productivity). We summarize our findings organized in these higher-level categories in Table 4.

Table 4: Technical Quality Behavioral Codes

Code	Representative Examples
Poor Work Quality	"Had to rewrite (S4)"; "PRs [that] were garbage (S65)"
Lack of Skills	"Complete lack of understanding (S67)"; "Lack of fundamental programming skills (S100)"
Technical Disruption	"Ignore others' PRs and put [their] own on top of them to create conflict (S41)"; "Put in so many PRs that others couldn't even work (S41)"
Technical Expertise	"Excellent coder (S1)"; "Ability to handle complex tasks (S67, S48)"
Exceptional Work Quality	"Exceptional quality work (S25)"; "Good clean code (S65)"
Critical Problem Solving	"Fixed many issues (S25)"; "Solving critical problems [of] design (S48)"
High Productivity	"Daily multiple things (S31)"; "Lot [of] code contributions (S89)"

Poor Work Quality. Was mentioned by students when teammates delivered code or artifacts that were careless, incorrect, or created additional rework for the team. This behavior was found in both *Couch Potato* and *Hitchhiker* evaluations, with distinct patterns in how it was perceived.

For *Couch Potatoes*, students described minimal effort and inattentive contributions that burdened others with extra fixes. This behavior was mentioned as doing "sloppy job" (S1) which other teammates "had to rewrite" (S4). Students also noted that these teammates would "randomly push random pull requests" (S66), forcing others to "fix few commits" (S64). Their work was frequently described as "barely helpful" (S119) or limited to "only fix small bugs" (S27). Other examples included contributions done not [in a] meaningful fashion" (S119), performing "tasks not at all required" (S118), and cases where they "did not help implementation or testing" (S119), leaving major responsibilities to the rest of the team.

In *Hitchhiker* evaluations, Poor Work Quality reflected a more deliberate disregard for quality practices, interpreted as intentional effort minimization. Students described teammates who "implemented all methods in one day incorrectly" (S51) and "pushed directly to main" (S51), bypassing collaborative safeguards. Their work was labeled as "PRs [that] were garbage" (S65) or "pushing goofy code" (S65). Other examples included code that "messed up most of the code [that was] added" (S51), which was submitted "without testing anything properly" (S51) and teammates who "keeps messing up the pull requests" (S51).

Lack of Skills. Represents a teammate's fundamental inability to execute required technical tasks effectively, regardless of effort investment. This behavioral pattern differs from **POOR WORK QUALITY** by reflecting capability deficits rather than inattentiveness or lack of care. Lack of Skills was identified in *Hitchhiker*'s evaluations.

Students mentioned those teammates had a "complete lack of understanding" (S67) about the project repository (S36) and demonstrate "lack of fundamental programming skills" (S100), also some level of "lack of computer science knowledge" (S100), "frequently claim[ing] ignorance of subject matter" (S100), and not being able to "understand how to complete [the] sprint" (S9).

Technical Disruption. Was mentioned by students when a teammate's technical contributions interfered with project workflows, created integration problems, or conflicted with the team's established development approach. This behavior differs from *Lack of Skills* in that the teammate may have technical capability, but their implementation choices disrupted coordination and required additional work from others to resolve.

Technical Disruption was identified in *Lone Wolf* evaluations. Students mentioned they caused many merge conflicts (S41) as they "ignore others' PRs and put [their] own on top of them to create conflict" (S41). One student remembered a case when the teammate "put in so many PRs that others couldn't even work" (S41).

Technical Expertise. Was mentioned by students when teammates demonstrated advanced technical knowledge, solved complex problems, and provided guidance that elevated the team's overall capability. This behavior contrasts with *Lack of Skills*, as it reflects competence and contrasts with *Technical Disruption* by showing expertise applied collaboratively rather than in isolation.

Students identified Technical Expertise in *Good Teammate* evaluations, and described them as "excellent coder" (S1), someone who "writes great code" (S65), having "excellent technical skills" (S48). Others highlighted the "depth of understanding" (S48), "being knowledgeable [in] react javascript" (S67), and their "ability to handle complex tasks" (S67, S48).

Exceptional Work Quality. Represents teammates' consistent delivery of high-standard technical solutions that exceed minimum requirements and demonstrate attention to detail, proper implementation practices, and integration considerations. Exceptional Work Quality was identified in *Good Teammate* evaluations, highlighting "exceptional quality work" (S25), "solid important code" (S65), and laying the "foundational project success" (S48). Others emphasized maintainability and style, describing code that was "always organized" (S30) and "good clean code" (S65).

Students also appreciated quality of testing and reliability, noting the "good tests [that were] reliable" (S100) and the "attentive accurate work" (S57). They praised teammates who consistently upheld high standards, such as those producing "always quality work" (S69), maintaining a "solid work throughout" (S21).

Critical Problem Solving. Represents teammates' ability to identify, analyze, and resolve complex technical challenges that impede project progress. This behavioral pattern differs from general technical competence by focusing specifically on diagnostic and resolution capabilities when facing hard or new problems. Critical Problem Solving was identified in *Good Teammate* evaluations. Students noted that these teammates "fixed many issues" (S25) and provided "important help [with] challenging" problems (S65), (S20) critical to project success, for example "solving critical problems [of] design" (S48).

High Productivity. Represents teammates' ability to complete significant amounts of technical work efficiently and effectively. This behavioral pattern builds on *Technical Expertise* and *Exceptional Work Quality*, highlighting not just competence and care but also the pace of contributions. Unlike those categories, however, High Productivity appears in both *Good Teammate* and *Lone Wolf* evaluations.

In *Good Teammate* evaluations, students described peers as delivering "daily multiple things" (S31), and even completing "20 PRs bugs features" (S69). Others called them as the ones bringing "most work output" (S83), the "biggest contributor[s]" (S5), recognized their "lot [of] code contributions" (S89), and praised a "hero" behavior of having "carried team workload" (S107).

In *Lone Wolf* evaluations, students recognized similarly high levels of output but paired with weak collaboration or integration challenges. They described peers as someone who "did significantly more work than anyone else" (S17), doing an "incredible amount of work" (S36, in a "tendency to complete many tasks in rapid succession" (S36) and "submitt[ing] the most pull requests" (S32). However, students also mentioned their tendency to act independently, as they were doing "all the beginning stuff before others had chance to look." (S21)

4.4 Timing

Well-planned schedules and synchronized activities are vital for project success, making contribution timing critical to avoid bottlenecks and keep progress coordinated [27]. Our analysis identified three timing-related behavior patterns, including two negative patterns (Last-Minute Behavior, Workflow Delays) and one positive pattern (Punctuality). Together, these patterns capture how students evaluate their peers' ability to manage work pacing, align with project schedules, and coordinate contributions in a way that supports smooth team progress.

Table 5: Timing Behavioral Codes

Code	Representative Examples
Last-Minute Behavior	"Waited last day [to] push PRs (S59)"; "Did not do any work until last day (S105)"
Workflow Delays	"Waiting for them [to] finish tasks (S1)"; "Frequently stalled (S44)"
Punctuality	"Regular great work (S114)"

Last-Minute Behavior. Encompasses a teammate's tendency to delay task initiation or completion until deadlines are imminent, compressing work schedules and disrupting coordinated progress. This pattern differs from legitimate timeline adjustments because delays stem from avoidance or disengagement rather than technical complexity or external constraints. Last-Minute Behavior was reported in *Couch Potato*, *Hitchhiker*, and *Lone Wolf* evaluations.

For *Couch Potatoes*, Last-Minute Behavior reflected passive time management and procrastination. Students noted peers who "got started [sprint 1] leftover issues late" (S1, S2), "waited last day [to] push PRs" (S59), "waited until end last week" (S70), or even "didn't start anything until last couple days" (S3–S74), leaving teammates with limited time to integrate their work.

In *Hitchhiker* patterns, Last-Minute Behavior often appeared deliberate, as a way to minimize sustained engagement while still meeting minimal requirements. Students described peers who "started trying two days before assignment due" (S64), "messed on day assignment was due" (S80), "finally completed something at very end" (S1), or "did not do any work until last day" (S105), sometimes resulting in what others called a "huge mess procrastinating all work till last day" (S49).

Among *Lone Wolves*, Last-Minute Behavior reflected independent scheduling rather than disengagement. Students reported teammates who "shows up at the last minute to patch up tasks" (S6), "waited until close to the due date to complete much of [their] work" (S7), indicating a preference for working alone and on their own timeline, even if this created integration pressure for the team.

Workflow Delays. Differ from *Last-Minute Behavior* by focusing not only on when an individual begins their own work, but on how those delays ripple through the team's workflow and stall collective progress. While *Procrastination* captures a teammate's late start or last-minute contributions, *Workflow Delays* emphasize the broader coordination impact, situations where slow or inconsistent contributions create bottlenecks, block dependent tasks, and force others to reschedule or duplicate effort. Workflow Delays emerged in *Couch Potato* evaluations.

Students reported "waiting for them [to] finish tasks" (S1), being "[put] in waiting position" (S2) causing the team to be "frequently stalled" (S44) and sometimes needing to "double working on things" (S28) due to lack of timely updates.

Punctuality. Represents teammates' consistent adherence to agreed timelines and predictable work rhythms that enable smooth coordination. Unlike *Last-Minute Behavior* and *Workflow Delays*, which disrupt planning and force others to compress or reschedule their work, Punctuality reflects contributions delivered on time and in a manner that supports the team's progress.

Punctuality emerged from *Good Teammate* evaluations. They mentioned teammates produced "regular great work" (S114, S116, S117), "complete work [on] time" (S48, S39)), hence being "very reliable" (S114).

4.5 Collaboration

Successful software engineering relies on collective decision-making and coordinated teamwork, making collaboration behaviors central to maintaining cohesion and shared progress [27]. Our analysis identified five collaboration-related behavior patterns, including three negative patterns (Dependability Issues, Code Review Gaps, and Dominating Behavior) and two positive patterns (Leadership and Helpfulness). Together, these patterns capture how students perceive their peers' engagement in shared responsibilities, decision-making, and mutual support throughout the project lifecycle.

Dependability Issues. Capture situations where teammates failed to meet commitments, follow through on tasks, or maintain consistent participation, forcing others to compensate. Dependability Issues emerged from *Couch Potato* and *Hitchhiker* evaluations.

For *Couch Potatoes*, Dependability Issues were associated with missing contributions and inconsistent participation that left others to complete their work. Students described having to "pick up the

Table 6: Collaboration Behavioral Codes

Code	Representative Examples
Dependability Issues	"Pick up the slack (S35)"; "Made it harder to succeed as a team (S49)"
Code Review Gaps	"Significantly fewer pull requests (S77)"; "Did not help review (S2)"
Dominating Behavior	"Extremely controlling (S39)"; "Like talking to a brick wall (S18)"
Leadership	"Handle team communication (S4)"; "Leading [the] meetings [to keep the conversation] on track (S23)"
Helpfulness	"Always willing [to] help (S68)"; "Offering help (S56)"

slack" (S35) or even "carry the team" (S107). Others pointed out broken commitments, saying the teammate "did not deliver what was promised" (S3–S115) or "never followed through" (S64). In some cases, students reported that these teammates would "disappear for weeks" (S74) or "not work consistently" (S44, S117), leaving major portions of the project unfinished until others intervened.

For *Hitchhikers*, Dependability Issues reflected a pattern of deliberate avoidance and broken promises that shifted responsibility onto others. Students described teammates who "needed [others] to pick up [their work]" (S80) or were "riding on [the] assumption that [the] team would pick up his slack" (S1). Several noted how these behaviors directly harmed the team, saying they "put [the team] behind at the end" (S117), made [the team] do more tasks than needed" (S95), and overall "made it harder to succeed as a team" (S49). Students also emphasized unreliability in commitments: teammates would "claim issues then get busy" (S117), later on "ask[ing] someone else to take over" (S117), or "fall[ing] through on many plans for issue completion [they themselves] proposed" (S1). These patterns were seen as "unproductive and detrimental to [the] team's progress" (S101), leaving peers feeling the distribution of work was unfair and unsustainable.

Code Review Gaps. Represent teammates' failure to engage effectively in the collaborative coding workflow, including both the submission of pull requests and the review of peers' contributions. This behavioral pattern goes beyond general low engagement by targeting the practices that ensure smooth integration and maintain code quality. Code Review Gaps were found in *Couch Potato* evaluations.

Several students reported that some peers made "significantly fewer pull requests" (S26, S115, S30, S117, S90) and just a "few commits" (S90). Others pointed out long periods of inactivity, saying there were "no commits/PRs until 24 [hours were] left" (S27), or the entire time (S115, S117). In addition to the lack of submissions, students emphasized failures in collaborative quality control, noting that these teammates "did not help review" (S2) or gave "no feedback during merge requests" (S2), which left code review responsibilities concentrated among more engaged teammates.

Dominating Behavior. Represents teammates' excessive control over decision-making, work distribution, or technical direction without properly involving others in discussion or reaching consensus. Dominating Behavior was identified exclusively in *Lone Wolf* evaluations.

Students characterized these teammates as "extremely controlling" (S39) and "difficult to work with" (S18), describing they would not "care to listen to anything [others] say" (S18) and acted as though the "best course of action is to just do what [they] say" (S18). They emphasized an authoritarian dynamic, reporting that these teammates were "unwilling to change [their] mind without specific direction of the Teaching Assistant" (S18) and that interacting with them felt "like talking to a brick wall" (S18). Students noted that these teammates acted as "[they were] the boss instead of a colleague" (S18), would "hardly let any team member do things their own way" (S39), and insisted that, the project "had to be [their] coding style and [their] testing styles" (S39). This control often excluded other perspectives, as they "rarely listen[ed] to input from anyone else" (S83) and, as one student expressed, "did not give anyone else a chance" (S18).

Leadership. Represents teammates' ability to guide the team through proactive coordination, decision-making support, and organizational efforts that facilitate collaborative success. Leadership emerged from *Good Teammate* evaluations.

Students mentioned the ability to "handle team communication" (S4), "track everything needed" (S33), and "ensure [the team was] on pace" (S33), "motivat[ing] team progress" (S65), and "guid[ing] sprint" (S77) deliverables. Many noted that these teammates took on meeting facilitation and organizational responsibilities (S107, S116, S27), also "leading [the] meetings [to keep the conversation] on track" (S23). Others highlighted their broader leadership role, describing them as "natural" (S39), "essential" (S42, S43) and "definite team leaders" (S33).

Helpfulness. Represents teammates' willingness to provide assistance, guidance, and support to others, both technically and collaboratively. Helpfulness emerged from *Good Teammate's* evaluations.

Students highlighted how these teammates "helped teammates [with] problems" (S5), were "always willing [to] help" everyone (S68, S52, S114, S52), including the teammates who were facing more difficulties (S63, S52). Students remembered these teammates as dedicating substantial time to teammates (S39) and proactively "offering help" (S56, S54).

Our study refines established teammate categories by grounding them in students' own language and highlighting both positive and negative aspects of each type. Good Teammates consistently showed initiative, responsiveness, technical skill, punctuality, and helpfulness, sustaining collaboration. Couch Potatoes were described as passively disengaged, contributing little across sprints. Hitchhikers showed strategic minimalism, doing just enough to receive credit. Lone Wolves contributed substantial work but in isolation, sometimes dominating decisions and complicating integration.

5 Discussion

In this section, we discuss our results in the context of the literature and suggest interventions for instructors.

5.1 Communication

Our study identified different communication behaviors when evaluating their teammates. Students described positive behaviors such as prompt responsiveness to messages, regular updates in agreed-upon channels (e.g., Microsoft Teams³), active participation in meetings, and closing the loop by summarizing outcomes. They also reported negative behaviors such as silence, delayed replies, and lack of meeting engagement, often linking these to teammate types like Couch Potato or Lone Wolf. These findings highlight that students view communication behaviors as strong signals of reliability and collaboration quality.

Since the official communication channel for the class we studied is already established (Microsoft Teams), instructors can focus on improving how teams use it. Prior research on collaborative and hybrid teamwork showed that training students in explicit communication norms (such as expected response windows, tagging teammates for urgent items, and summarizing key decisions) reduces coordination breakdowns and increases participation [19]. Instructors can encourage teams to (1) agree on turnaround times for responses, (2) rotate responsibility for posting meeting summaries or action items, and (3) use message threads consistently to keep discussions organized.

5.2 Engagement

Our study found that students justified teammate classifications by referring to differences in engagement, whether teammates (in)consistently contributed, participated (or not) in shared work, and showed (or not) initiative. Teammates labeled as Good Teammate were often described as “*always involved*” or “*actively helping*,” whereas those labeled Couch Potato or Hitchhiker were criticized for missing deadlines, contributing only minimal effort, or failing to show up to meetings.

Previous research on teamwork in SE courses showed that social loafing is a persistent challenge and that peer assessment can mitigate it by making contributions visible [21]. Prior work in psychology showed that engagement predicts both individual learning and team effectiveness [1], and our findings support these results: students are clearly sensitive to differences in effort and use engagement cues as a basis for team judgments.

To foster engagement in a context where formal roles are absent and students perform similar tasks (e.g., implementing features, solving and approving PRs), instructors can support the emergence of shared leadership rather than assigning fixed roles. Our results show that students valued peers who guided communication, kept meetings on track, and motivated team progress, but complained when leadership was either missing, so leaving the team disorganized—or dominating, shutting out others’ contributions.

Instructors can scaffold healthy leadership by coaching students on consensus-building (e.g., rotating who summarizes sprint status), modeling inclusive decision-making, and fostering psychological safety so students can challenge unilateral decisions. Recognizing and providing feedback on coordination tasks—updating boards, reminding peers of deadlines, facilitating discussions—helps legitimize this *invisible work* and distribute it more fairly, reducing frustration about absent or over-controlling leaders.

³<https://www.microsoft.com/en-us/microsoft-teams/group-chat-software>

5.3 Technical Quality

Good Teammates from our sample were praised by their teammates for producing correct, well-structured solutions and clear documentation, while Couch Potatoes were criticized for submitting incomplete, buggy, or poorly documented work that created rework for others. These results suggest that students do not assess contribution volume alone but also use quality as a fairness and reliability signal.

Previous research on code review in professional SE showed that review conversations focus heavily on maintainability, clarity, and long-term quality, not just defect detection [24]. Our findings support this pattern in the classroom context: students spontaneously raised concerns about rework and code cleanliness, suggesting that they, too, see technical quality as a determinant of trust and fairness. Educational research in broader teaching contexts showed that making evaluation criteria explicit through rubrics helps students understand what constitutes high-quality work and improves their performance [3]. Our findings support these results, as students often justified peer types by referring to correctness, completeness, and rework effort.

Instructors can help students improve technical quality by making quality expectations explicit with coding standards (e.g. Code Climate [4]), definition-of-done checklists, and rubric items for correctness, style, and documentation.

5.4 Timing

Our analysis shows that students are highly sensitive to when their peers complete work. Last-Minute Behavior and Workflow Delays were cited as disruptive not only because of the technical impact (e.g., integration pressure, duplicated effort) but also because they signaled low commitment and forced other team members into crisis mode.

These results align with prior research demonstrating that temporal coordination is a key predictor of team success and satisfaction in project-based learning contexts [15, 21]. Conversely, Punctuality emerged as a strong positive cue: students associated on-time, steady work rhythms with reliability, dependability, and psychological safety.

Instructors can help teams define clear internal deadlines during sprint planning. Research on student project management shows that when teams agree on intermediate dates together, they are more likely to adhere to them [11, 18].

5.5 Collaboration

Good Teammates were well-evaluated for helping others, sharing knowledge, and contributing to decision-making, while Couch Potatoes and Lone Wolves were criticized for isolating themselves, failing to coordinate, or dominating conversations in ways that excluded others (Table 6). These findings show that students view collaboration not merely as co-presence but as active participation in collective planning, shared problem solving, and maintaining group cohesion.

Prior work on collaborative learning has shown that teams with a balanced distribution of roles perform better and report higher perceived effectiveness [16]. In SE education, studies have found that effective collaboration depends on both social integration and

shared technical responsibility—teams that fail to integrate contributions early suffer from bottlenecks and rework, while teams with equitable participation report higher satisfaction and better learning outcomes [13, 15]. Research on psychological safety also highlights that teams where members feel safe to ask questions and challenge decisions without fear are more innovative and productive [6]. Our results reinforce these findings: students rewarded peers who supported others, facilitated shared decision-making, while penalizing those who dominated conversations or failed to contribute to reviews, signaling that collaboration quality strongly shapes their perception of fairness and trust.

These insights suggest that instructors can do more than simply form teams. They can actively scaffold collaboration across the semester. Using short surveys to capture students' time availability, collaboration style, and grade ambition allows teams to be formed with fewer schedule mismatches and expectation conflicts [7].

Once teams are created, a structured kickoff session can guide them to define decision-making processes, code review expectations, and communication practices, with written agreements providing a shared reference point when conflicts arise. Collaboration can be further strengthened by encouraging rotation of facilitation, documentation, and integration roles across sprints so that invisible work is distributed fairly and no single student dominates coordination. Short reflections at the end of each sprint, such as asking students what helped or hindered collaboration, create opportunities for teams to self-diagnose and for instructors to intervene early when patterns of exclusion or conflict emerge [13].

6 Related Work

Prior work in SE education has examined teamwork from multiple perspectives, including peer evaluation, communication practices, and coordination. Research on peer assessment shows that frequent feedback helps instructors identify emerging issues and reduce social loafing by making contributions visible [13, 21]. However, these studies primarily focus on when and how often feedback is collected, offering limited insight into how students reason about their evaluations.

Communication and participation have also been identified as key predictors of success in SE project courses. Studies of remote and hybrid collaboration highlight that structured communication norms and equitable participation improve satisfaction and project outcomes [15, 19]. More recently, Luther et al. [14] analyzed communication patterns associated with common teammate types using Ordered Network Analysis, showing systematic differences in frequency and content of interactions. While this work characterizes how much and what kind of communication occurs, it does not explain the behavioral reasoning students use when assigning teammate labels.

The Comprehensive Assessment for Team-Member Effectiveness (CATME) [12] provides a widely adopted quantitative framework for peer evaluation by operationalizing teamwork through standardized dimensions such as contribution, interaction, and responsibility. Our work complements and operationalizes CATME by explicating the behavioral reasoning students use when assigning these evaluations. Through qualitative analysis of open-text justifications, we identify 28 concrete behavioral characteristics,

spanning communication practices, engagement patterns, technical quality, timing, and collaboration, that help explain why students assign particular ratings or teammate labels. For example, CATME's contribution dimension is reflected in behaviors such as consistent task completion, meeting deadlines, and proactive technical ownership, while the interaction dimension maps to behaviors including timely responses, constructive feedback, and coordination during integration phases. At the same time, our results surface nuances not explicitly captured by CATME scales, such as *strategic minimal participation* among Hitchhikers or *the dual productive-disruptive role* of Lone Wolves.

By grounding CATME dimensions in students' own language and observable behaviors, our findings can inform richer feedback, support interpretation of numeric scores, and enable instructors to design targeted interventions that address specific teamwork breakdowns rather than relying solely on aggregate ratings.

7 Threats to Validity

There are some limitations related to our research results.

Internal validity. One potential threat is researcher bias during the qualitative coding process. To mitigate this, three authors independently performed open coding, iteratively refining the codebook through negotiated agreement sessions until consensus was reached. A fourth author, with extensive qualitative research experience, reviewed all codes and categories for consistency. Another internal threat is that students may have been influenced by social desirability bias when labeling teammates (e.g., avoiding negative labels to preserve relationships).

External Validity. The study was conducted in a single senior-level SE course at one institution, limits generalizability to other courses, disciplines, or educational contexts. However, our course structure—semester-long projects, sprint-based iterations, and peer evaluation cycles—is common in SE education, increasing the likelihood that our findings transfer to similar project-based learning settings. Future work should replicate the study across multiple institutions, course levels, and cultural contexts to strengthen external validity.

Construct Validity: Our operationalization of teammate types relied on students' subjective classification and their open-text justifications, which may reflect perceptions rather than objective measures of contribution. However, this approach aligns with our goal of understanding students' interpretations of peer behavior.

Reliability and Data Consistency To ensure analytic rigor, we followed established qualitative methods, including open coding, iterative refinement, and negotiated agreement. We also maintained an audit trail and included a replication package containing our codebook and evaluation questions to support reproducibility and independent verification of our results.

Theoretical saturation. A potential limitation in qualitative studies regards reaching theoretical saturation. While we analyzed a substantial dataset (570 peer evaluations with high response rates across three sprints), we cannot claim complete theoretical saturation. Nonetheless, after multiple coding sessions, new justifications were consistently mapped to existing categories, suggesting that we captured the most salient behaviors relevant to teammate classification in this context.

8 Conclusion

This paper presented an empirical study of how students judge their teammates in a project-based SE course, drawing on three consecutive sprints of peer evaluations.

Our qualitative analysis of 570 peer evaluations revealed that students rely on rich behavioral reasoning when classifying teammates, referencing engagement, communication, timing, technical quality, and collaboration rather than simply assigning a type.

These findings indicate that labels such as GOOD TEAMMATE or COUCH POTATO compress diverse signals into a single judgment, which may obscure the actionable behaviors students wish to see reinforced or improved.

Our results suggest that peer-evaluation rubrics should evolve beyond global labels toward nuanced, behaviorally anchored questions that prompt students to rate dimensions like responsiveness, reliability, quality of work, and collaborative contributions. Such instruments would produce more reliable, formative feedback and provide instructors with clearer insight into which aspects of teamwork require coaching.

Instructors can further scaffold teamwork by modeling inclusive communication practices, rotating coordination roles, and offering targeted support to students struggling with quality or timeliness.

Future work should replicate this study across institutions, course designs, and cultural contexts, as well as examine how refined peer-evaluation instruments influence both student learning outcomes and team dynamics. Integrating behavior-specific feedback with activity traces (e.g., commits, pull requests, issue participation) can provide instructors with timely indicators of team health. Future research should also investigate how to better distinguish between teammate types that are often grouped together in practice, particularly Hitchhikers and Couch Potatoes. While both are perceived as low contributors, our findings suggest that Couch Potatoes are characterized by passive disengagement, whereas Hitchhikers exhibit more strategic minimal participation aimed at preserving credit with limited effort. Understanding these distinctions could enable the design of more targeted interventions, such as engagement-focused support for disengaged students and accountability mechanisms for strategic free-riding behaviors.

Finally, future work could examine the longitudinal stability of teammate types across multiple projects and courses, exploring whether these behaviors reflect temporary situational factors or persistent collaboration patterns. By making peer evaluations more granular and actionable, educators can better support the development of teamwork skills that are critical for professional SE practice.

References

- [1] Arnold B Bakker, Simon L Albrecht, and Michael P Leiter. 2011. Key questions regarding work engagement. *European journal of work and organizational psychology* 20, 1 (2011), 4–28.
- [2] Terri Feldman Barr, Andrea L. Dixon, and Jule B. Gassenheimer. 2005. Exploring the “Lone Wolf” Phenomenon in Student Teams. *Journal of Marketing Education* 27, 1 (2005), 81–90. <https://doi.org/10.1177/0273475304273459> arXiv:<https://doi.org/10.1177/0273475304273459>
- [3] Susan M Brookhart. 2018. Appropriate criteria: Key to effective rubrics. In *Frontiers in education*, Vol. 3. Frontiers Media SA, 22.
- [4] CodeClimate. 2019. expertiza/expertiza - Code Climate. <https://codeclimate.com/github/expertiza/expertiza>. [Online; accessed 26-September-2025].
- [5] Henry Coles, Thomas Laurent, Christopher Henard, Mike Papadakis, and Anthony Ventresque. 2016. Pit: a practical mutation testing tool for java. In *Proceedings of the 25th international symposium on software testing and analysis*. 449–452.
- [6] Amy Edmondson. 1999. Psychological safety and learning behavior in work teams. *Administrative science quarterly* 44, 2 (1999), 350–383.
- [7] Fabian Fagerholm, Stanislav Chren, and Lassi Haaranen. 2025. Student Perceptions of Group Formation in Software Engineering Courses. In *2025 IEEE/ACM 37th International Conference on Software Engineering Education and Training (CSE&T)*. IEEE, 135–145.
- [8] Gordon Fraser and Andrea Arcuri. 2011. Evosuite: automatic test suite generation for object-oriented software. In *Proceedings of the 19th ACM SIGSOFT symposium and the 13th European conference on Foundations of software engineering*. 416–419.
- [9] Sahithya Gandu, Anurag Vemula, Satyanarayanachowdary Kadiyala, Marcia Moraes, Sudipto Gosch, Fabio Santos, and Bianca Trinkenreich. 2025. Replication Package. <https://figshare.com/s/970811e588627328a1ac>.
- [10] D Garrison, Martha Cleveland-Innes, Marguerite Koole, and James Kappelman. 2006. Revisiting methodological issues in transcript analysis: Negotiated coding and reliability. *The Internet and Higher Education* 9, 1 (2006), 1–8.
- [11] Josette MP Gevers, Wendelien van Eerde, and Christel G Rutte. 2009. Team self-regulation and meeting deadlines in project teams: Antecedents and effects of temporal consensus. *European Journal of Work and Organizational Psychology* 18, 3 (2009), 295–321.
- [12] Misty L Loughry, Matthew W Ohland, and D DeWayne Moore. 2007. Development of a theory-based assessment of team member effectiveness. *Educational and psychological measurement* 67, 3 (2007), 505–524.
- [13] Yu Lu and André Van Der Hoek. 2025. Visualizing Frequent Peer Assessment Results to Uncover Problems in Team Dynamics. In *2025 IEEE/ACM 37th International Conference on Software Engineering Education and Training (CSE&T)*. IEEE, 365–375.
- [14] Yanye Luther, Lindsey Nielsen, Logan Cadman, Marcia Moraes, Sudipto Ghosh, and Bianca Trinkenreich. 2025. Analyzing the Communication Patterns of Different Teammate Types in a Software Engineering Course Project. In *33rd ACM International Conference on the Foundations of Software Engineering (FSE Companion '25)*, June 23–28, 2025, Trondheim, Norway (FSE 2025). ACM, Trondheim, Norway. <http://doi.acm.org/10.1145/3696630.3727232>
- [15] Linda Marshall, Vreda Pieterse, Lisa Thompson, and Dina M Venter. 2016. Exploration of participation in student software engineering teams. *ACM Transactions on Computing Education (TOCE)* 16, 2 (2016), 1–38.
- [16] Nicoleta Meslec and Petru Lucian Curşeu. 2015. Are balanced groups better? Belbin roles in collaborative learning groups. *Learning and Individual Differences* 39 (2015), 81–88.
- [17] Matthew B Miles and A Michael Huberman. 1994. *Qualitative data analysis: An expanded sourcebook*. sage.
- [18] Marcello Missiroli, Daniel Russo, and Paolo Ciancarini. 2016. Learning agile software development in high school: an investigation. In *Proceedings of the 38th international conference on software engineering companion*. 293–302.
- [19] Makayla Moster, Denae Ford, and Paige Rodeghero. 2021. “Is My Mic On?” Preparing SE Students for Collaborative Remote Work and Hybrid Team Communication. In *2021 IEEE/ACM 43rd International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET)*. IEEE, 89–94.
- [20] Barbara Oakley. 2003. Coping with hitchhikers and couch potatoes on teams [Student’s Corner]. *IEEE Engineering in Medicine and Biology Magazine* 22, 5 (2003), 9–10. <https://doi.org/10.1109/MEMB.2003.9625655>
- [21] Barbara Oakley, Richard M Felder, Rebecca Brent, and Imad Elhajj. 2004. Turning student groups into effective teams. *Journal of student centered learning* 2, 1 (2004), 9–34.
- [22] Christopher Ong and Kevin McGee. 2012. Good team-mates do more than help the team win: design factors that impact player concern about team-mate experience. In *Proceedings of the 24th Australian Computer-Human Interaction Conference (Melbourne, Australia) (OzCHI '12)*. Association for Computing Machinery, New York, NY, USA, 440–448. <https://doi.org/10.1145/2414536.2414605>
- [23] Carlos Pacheco and Michael D Ernst. 2007. Randoop: feedback-directed random testing for Java. In *Companion to the 22nd ACM SIGPLAN conference on Object-oriented programming systems and applications companion*. 815–816.
- [24] Peter C Rigby and Christian Bird. 2013. Convergent contemporary software peer review practices. In *Proceedings of the 2013 9th joint meeting on foundations of software engineering*. 202–212.
- [25] Margaret-Anne Storey, Neil A Ernst, Courtney Williams, and Eirini Kalliamvakou. 2020. The who, what, how of software engineering research: a socio-technical framework. *Empirical Software Engineering* 25 (2020), 4097–4129.
- [26] Bianca Trinkenreich, Ricardo Britto, Marco A Gerosa, and Igor Steinmacher. 2022. An empirical investigation on the challenges faced by women in the software industry: A case study. In *Proceedings of the 2022 ACM/IEEE 44th International Conference on Software Engineering: Software Engineering in Society*. 24–35.
- [27] Hironori Washizaki and Joanna Isabelle Olszewska. 2024. Guide to the Software Engineering Body of Knowledge v4.0. (2024).