

IntentLint: Supporting Intent Scaffolding and Prompt-time Linting in Human-AI Collaborative Data Analysis

Felicia Li Feng
School of Computer Science,
University of Waterloo
f4feng@uwaterloo.ca

Jian Zhao
School of Computer Science,
University of Waterloo
jianzhao@uwaterloo.ca

Anamaria Crisan
School of Computer Science,
University of Waterloo
ana.crisan@uwaterloo.ca

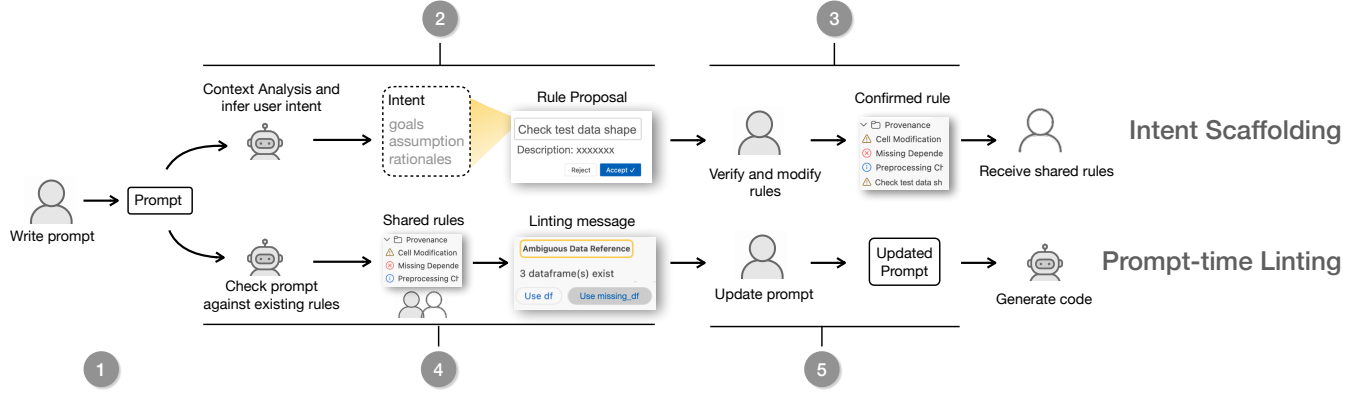


Figure 1: Overview of how our mechanisms work. When Devin submits a prompt (1), the rule-proposal agent analyzes the shared context and prompt, infers user intent, and proposes a rule for Devin to review, which is a coordination norm capturing Devin’s intent (2). Once Devin verifies the rule proposal, it becomes a confirmed rule and is shared with other teammates, such as Maya (3). In parallel with (2), the rule-checking agent checks the current prompt against all existing rules. If the rule-checking agent detects potential conflicts with collaborators’ work, it will call the linting agent to generate a linting message that highlights prior analytic intent, relevant context, and suggested prompt revisions (4). Guided by this linting feedback, Devin updates the prompt for final code generation (5).

Abstract

In human-AI collaborative data analysis, as analyses rapidly evolve, the artifacts meant to capture shared understanding often become incomplete or difficult to interpret, leading to undocumented assumptions, cross-user misaligned intent, context-poor prompts, and unwanted agent behaviors. To address these challenges, we introduce a rule-based coordination layer with two interaction mechanisms, intent scaffolding and prompt-time linting, that make analytic intent explicit and actionable during human-AI collaborative data analysis. We implement them in **INTENTLINT**, a proof-of-concept system that infers analytic intent from shared notebooks, represents it as structured, editable rules, and checks users’ prompts against shared rules. **INTENTLINT** helps analysts externalize and refine their intent and proactively checks prompts for potential conflicts. A study with 16 data analysts shows that IntentLint improves awareness of collaborators’ intent and encourages reflection on analytic strategies, and provides design implications for supporting more aligned and transparent human-AI collaborative data analysis.



This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License.

UIST '26, Detroit, MI, USA

© 2026 Copyright held by the owner/author(s).

ACM ISBN 979-8-4007-2856-3/2026/11

<https://doi.org/10.1145/3830398.3830509>

CCS Concepts

• **Human-centered computing** → **User interface programming.**

1 Introduction

Data analysis is fundamentally collaborative: analysts must coordinate and maintain a shared understanding of what has been done, why it was done, and what should happen next [16, 65, 88]. In practice, this shared understanding is often fragile because the artifacts meant to capture it, such as computational notebooks and external documents, often become incomplete, outdated, or difficult to interpret as analyses evolve [43, 66, 79]. Even when they are of high quality, collaborators must still expend significant effort to interpret them and infer collaborators’ intent behind prior decisions [80, 83]. As large language models (LLMs) are increasingly used to support data analysis, they have shown promise in assisting with code generation, explanation, and conversational interaction with data [52, 72]. These systems suggest that LLMs could help reduce the burden of interpreting collaborators’ work by synthesizing documentation, summarizing analyses, and generating explanations within ongoing workflows.

However, these advances do not fully address the demands of coordination and maintaining shared understanding in human-AI collaborative analysis. As LLMs become active participants in analytic workflows, coordination is required not only among human

collaborators, but also between people and AI agents, and across multiple AI agents [12]. This shift introduces new challenges, including supporting shared mental models, enabling inspection of agent reasoning, and maintaining coherence across agents [86]. Yet existing approaches primarily support artifact production and retrospective documentation [72, 79, 93], offering limited support for these coordination challenges in human-AI collaborative analysis.

To better understand these challenges, we conducted a formative study with five data analysts performing data analysis tasks. We found that analysts struggled to externalize analytic intent in ways legible to both collaborators and AI agents; undocumented assumptions propagated silently into AI-generated code; and conflicts arising from misaligned decisions were rarely surfaced before causing downstream inconsistencies. These breakdowns suggest a need for mechanisms that (1) make intent explicit and shareable, and (2) proactively detect misalignment during interaction.

We introduce a rule-based coordination layer for AI-assisted collaborative notebooks with two interaction mechanisms, intent scaffolding and prompt-time linting, to encode intent as persistent shared rules and enable early issue detection. Together, these mechanisms form a feedback loop between intent articulation and action: making reasoning explicit and reintroducing it at critical moments to guide ongoing analysis. We demonstrate them through a proof-of-concept system, *INTENTLINT*. *INTENTLINT* analyzes computational notebooks to extract context about code dependencies, dataflow, and prior analytic results. From this context and user prompts, it infers likely goals, assumptions, and decision rationales, presenting them as structured, editable rules for analysts’ review. When a user issues a new prompt, *INTENTLINT* checks it against active rules and flags potential conflicts before code generation, helping steer agent behavior toward outputs consistent with the team’s analytic goals.

We evaluated *INTENTLINT* with 16 participants with data analysis backgrounds. Results show that *INTENTLINT* helped participants better understand collaborators’ activities and intent, reflect deeper on their own analytical choices, and identify misalignments earlier. Analytic intent scaffolding encouraged participants to make their strategies explicit and revisit them over time, while prompt-time linting surfaced conflicts between current prompts and previously established goals or constraints. Together, these results suggest that intent scaffolding and prompt-time linting can strengthen shared understanding in human-AI collaborative data analysis.

Our research provides the following contributions:

- **A formative study** that identifies breakdowns in analytic intent and knowledge sharing across multi-human, multi-agent data science workflows, revealing design opportunities for interaction techniques that support collaboration and alignment.
- **A proof-of-concept system** that operationalizes analytic intent scaffolding and prompt-time linting, enabling analysts and AI agents to align on goals, assumptions, and decisions in data analysis.
- **A user study** that demonstrates how real-time scaffolding and linting improve usability, collaboration, and understanding of shared analytic intent, and provides actionable design implications for future human-AI collaborative systems.

Despite advances in AI-assisted analysis, our research shows that collaborative data analysis workflows often break down when analytic intent is implicit or when coordination is unclear. We demonstrate how structuring collaboration through intent scaffolding and prompt-time linting enables analysts and AI agents to build on prior analyses and coordinate more effectively.

2 Related Work

We review prior work on collaborative data analysis and AI-assisted workflows, focusing on shared understanding and intent alignment.

2.1 Collaborative Data Analysis

Maintaining common ground is a fundamental challenge in collaborative data analysis, as intermediate artifacts are often exploratory and difficult to interpret [4, 31]. Documentation plays a key role in sustaining shared understanding [76], and prior systems have explored varied designs to support this. Systems like Callisto, Verdant, and KTGraph support sensemaking by linking discussions to artifacts, visualizing analysis histories, and externalizing reasoning through annotations and provenance tracking [39, 80, 90]. Other work focuses on cross-role knowledge transfer, such as ZIva, which facilitates communication of domain expertise to data scientists [57]. However, these systems primarily focus on human-human collaboration. The introduction of LLMs fundamentally complicates knowledge sharing, as systems must also account for how human knowledge is communicated to AI agents.

Recent work explores knowledge sharing in LLM-mediated environments. Tools like CoAlcoder, SemanticCommit, and CoPrompt leverage context, detect semantic conflicts, and support reuse of prompting strategies, while studies note that prompts are often not preserved, limiting reproducibility and collaboration [20, 21, 28, 75]. Other systems, such as Respark and PolicyPad, focus on transferring human knowledge to guide LLM behavior [18, 73]. However, these systems either overlook the collaborative dynamics of multi-human settings or treat knowledge mainly as constraints for AI agents rather than shared artifacts for all participants. Building on this gap, our work explores knowledge sharing across multi-human and agent workflows, where analytical knowledge is jointly constructed, negotiated, and reused.

2.2 Intent alignment in AI-assisted data analysis

LLM agents with different roles have been applied across the data science workflow [63], from translating natural-language queries into visualizations (e.g., Chat2VIS, ChartGPT) to supporting higher-level, concept-driven analysis authoring (e.g., Data Formulator) and decomposing complex analytical intents into structured subtasks (e.g., DocWrangler, LightVA) [52, 63, 72, 81, 91]. Beyond individual interactions, recent systems investigate LLM-driven analysis workflows. InsightPilot autonomously generates sequences of analysis steps and insights [51], while DataNarrative employs multi-agent architectures to generate and verify analytical narratives [33]. Data Director and similar systems extend this paradigm to multimodal outputs, such as automated data videos [69]. However, these approaches often limit user control and provide little support for expressing user intent throughout the workflow.

Category	Challenges in human collaborative data analysis	Challenges in human-AI team collaboration	Reference
Code management & technical debt	C1: Non-executed code cells with only code comments result in confusion C2: Modification of code cells used by others (unstable cell dependencies) C3: Unintentional function and variable redefinitions	(confuse agents about the cell's effect and lead to inappropriate code generation) Agent modifies code without explicit notification to humans Agent redefine variables without explicit notification to humans	[59] [50, 59] [59, 77]
Analytical rigor & methodology	C4: Difficulty interpreting algorithmic outputs (e.g., correlation coefficients) C5: Subjective pattern identification without documented criteria C6: Unclear methodology for pattern validation	Distinguish human work and AI generation Agent infer pattern definition without surfacing uncertainty Agent infer methodology without surfacing uncertainty	[58, 64] [2, 3, 37, 44] [35, 45, 87, 92]
Data governance & version control	C7: Assuming data structure without verification or documentation C8: In-place data mutations without version tracking C9: Neglected data cleaning steps leading to unreproducible or misleading results C10: Ambiguous data source	Reliability in understanding large datasets without explicit guidance Confuse agents about dataset state, causing inconsistent analyses and conflicts Struggle to interpret if issues are from data or preprocessing Agent infer data source without surfacing uncertainty	[9, 36, 37] [30] [11] [36, 78]
Documentation & reproducibility	C11: Undocumented failed analysis attempts or exploratory paths C12: Lack of documentation for discretionary pre-processing decisions C13: Lack of documentation for data transformations C14: Difficulty understanding how results (charts, tables) were produced C15: Lack of documentation for statistical assumptions and methodology	Cause agents to repeat failed analyses or miss context More effort spent reading generated explanations that offer little value Agent infer data source without surfacing uncertainty Agent provides potentially incorrect inferences Agents apply inconsistent methods and generate conflicting conclusions	[40, 41, 88] [54, 58] [36, 67] [30, 41] [36, 67]

Table 1: Challenges in collaborative data analysis identified from 25 prior studies

Other work focuses on improving user awareness and interaction with AI-generated outputs. Prior systems support intent alignment by enabling users to steer and verify AI analyses through editable plans and intermediate representations [38, 84], improving alignment between user prompts and generated scripts [94], and interactively resolving ambiguous user intent [22, 53]. Related systems also improve verification by exposing richer data context, such as Xavier, which integrates dataset semantics into code suggestions [93], and Amplio, which helps analysts explore underrepresented regions of datasets [85]. At a broader level, studies of agent configuration practices show that developers tend to encode functional instructions but rarely specify higher-level constraints such as intent, preferences, or quality criteria [8, 34]. Benchmarks such as BLADE show that LLM agents struggle with open-ended analytical tasks, producing shallow and low-diversity analyses [25]. While these approaches establish effective mechanisms for intent alignment in single-user, single-session analysis, our work addresses the additional coordination challenges of maintaining shared, evolving intent across multiple humans and AI agents.

Emerging research highlights the importance of making AI reasoning and coordination more transparent. Systems such as Cocoa demonstrate that shared, inspectable plans improve human-AI collaboration [19], while collaborative environments integrating visible agent roles and actions reshape coordination practices [47]. Similarly, Narrative Scaffolding emphasizes externalizing reasoning processes to support more robust and defensible analysis [32]. However, intent is still largely treated as a static input or plan rather than a shared, evolving construct. Our work builds on this gap by supporting intent alignment as an ongoing, collaborative process across both human and AI participants.

3 Formative Study

We conducted a formative study to better understand the challenges in collaborative data analysis involving multiple people and an AI coding assistant. Grounded in issues identified in prior literature (Table 1), we employ a Jupyter notebook as a technology probe to situate, re-examine, and extend these challenges in the context of contemporary AI code-assistant tools. Our study focused on asynchronous workflows, where analysts must interpret changes after they occur, whether made by themselves, other people, or AI assistants [74]. This setting presents realistic and unique challenges for coordination and maintaining shared knowledge.

3.1 Technology Probe Development

The study of collaborative data analysis, both with and without automated support (e.g., AutoML [66, 82] or AI assistants [51, 52]), is well established. Rather than re-eliciting known challenges through self-report, which risks abstract and incomplete accounts, we build on prior work by employing a computational notebook as a technology probe that instantiates these challenges as concrete scenarios, enabling participants to engage with them in realistic workflows.

3.1.1 Defining Collaborative Data Analysis Challenges. As part of our formative study, we constructed a set of 15 challenges in collaborative data analysis from 25 state-of-the-art papers (Table 1). We focused on issues related to knowledge and context sharing among people. Papers were selected from HCI, CSCW, and software engineering based on their influence and relevance to computational notebooks - the workhorse of data analysis. Using a light-weight open coding, we identified recurring problem descriptions and design observations, and iteratively consolidated them into 15 challenges organized into four categories. As we analyzed additional papers, few new themes emerged, suggesting sufficient coverage for our purposes. This synthesis is not intended to be exhaustive, but rather to provide a representative baseline that grounds our formative study; participants were able to critique, extend, and introduce additional challenges during the study.

3.1.2 Construction a Computational Notebook Technology Probe. Each scenario was implemented as a realistic but synthetic code segment within a single Jupyter notebook, grounded in the Ames Housing dataset [1] to maintain a consistent analytical context. The notebook was pre-executed to ensure uniform presentation, with each scenario depicting a plausible failure state, such as a silently overwritten variable or an undocumented preprocessing step. On the study, Participants could freely edit and re-execute cells. The full notebook is included in the supplementary materials.

3.2 Procedure, Participants, & Data Collection

Participants provided consent at the beginning of each session. Each participant received a unique instance of the computational notebook probe. After taking a few minutes to familiarize themselves with the notebook content, they were then guided through 15 brief scenarios illustrating the challenges summarized in Table 1. In each scenario, participants were shown the actions of two other human users (User A and User B) and asked to reflect on their understanding of these actions from the perspective of each user, gaps and challenges in understanding the user's actions, and how

they could be supported by AI assistants or system affordances. We encouraged participants to think-aloud [48] to capture their thought processes. Participants were also asked to reflect on their own experiences encountering these challenges and shared expectations for how LLMs might assist in addressing them. At the end of the session, participants could further elaborate on the challenges presented or propose additional ones. Sessions lasted between 60 and 80 minutes. The study was approved by our institution’s Ethics Board. Participants were remunerated \$30/hr via a digital gift card, with rates adjusted according to the time spent.

3.2.1 Participant Recruitment & Demographics. We recruited five participants (2 males, 3 females; ages 23 - 26) through purposive sampling [17], targeting data professionals who routinely collaborate with others in analysis workflows and use AI coding assistants. Participants were screened on the basis of years of professional data analysis experience, frequency of conducting data analyses, and self-reported familiarity with AI coding assistants (interpreted broadly as using formal coding agents to chatbot interactions asking for coding advice, etc.). All recruited participants had more than three years of data analysis experience, and regularly used LLM-driven tools (See appendix for full demographic details).

3.2.2 Data Collection and Analysis. We collected modifications that participants made to each notebook instance, recordings of their sessions, and any notes made by the study administrator. Audio was transcribed and analyzed using an inductive thematic analysis, seeking to surface insights that informed design goals, rather than strict inter-coder reliability [5, 71].

3.3 Findings

Data analysts collaborating on multi-author notebooks face coordination, context, and reasoning challenges that are amplified when AI assistants are involved. Although some challenges resemble those arising from human collaborators, the absence of social dynamics in AI changes how participants experience and respond to these interactions. Participants reflected on the persistence of these challenges over their careers and their expectations toward how AI agents should respond to support ongoing, asynchronous collaboration between people.

3.3.1 Persistent challenges with collaborative data analysis. Across participants, we observed persistent challenges in collaborative data analysis that span the categories identified in prior work (C1–C15).

Code management & technical debt (C1–C3). Asynchronous updates by people or coding agents contributed to challenges in managing changes in the notebook, particularly when there were execution errors (C1). Participants reported that this problem was exacerbated by coding agents because, unlike humans, it was often difficult to seek clarification on their intentions behind changes and resulting errors. Of particular concern was when coding agents acted on their own accord, by overwriting user code to introduce unstable dependencies (C2), modifying functions and variables (C3), and otherwise failing to consider human objectives in the existing code block. While both human and AI-driven modifications can introduce technical debt, participants reflect that AI technical debt, particularly verifying its outputs, was more disruptive and required active mitigation.

Analytical rigor & methodology (C4–C6). Code-management and issues of technical debt ultimately contributed to questions about the overall rigor of the data analysis. Separating between human-generated and AI-generated code was especially concerning, with participants raising the propensity of coding agents to hallucinate, or confidently provide inappropriate suggestions (e.g., algorithmic choice). As P1 mentioned, *“AI generations [sic] cannot be perfect, so this uncertainty needs to be portrayed to the user.”* This often made it difficult for participants to understand the outputs of analysis algorithms (C4), the logic of the overall analysis (C6), and the rationale for the choices that have been made (C5). Participants reflected that when working with people, discussion played a key role in resolving these ambiguities. However, the same nuanced discussion was more limited with coding agents. Additionally, P2 expressed a preference for AI to proactively ask clarification questions rather than producing inferences based on incomplete information.

Data governance & version control (C7–C10). As datasets grow in size and complexity, participants expressed doubt that coding agents could (1) adequately understand these data (C7), particularly in pre-processing steps, and (2) reliably record changes to the data state (C8) along with the methodological evolution of the analysis workflow. They were especially concerned with AI’s ability to reliably detect issues in the workflows (C9, C10), noting that *“human collaborators can usually infer this is probably a mistake, but AI may assume humans always make the right decisions.”* [P1]. Participants also expressed differing preferences regarding how AI should intervene when issues are detected. Some preferred that AI provide suggestions along with rationales for code fixes, while others favored automated fixes for straightforward issues.

3.3.2 Documentation & reproducibility (C11–C15). Finally, participants highlighted that LLMs are limited in supporting documentation and reproducibility. Data analysis is iterative updating to new findings and observations [40]. This makes documenting assumptions particularly important (C11–C13). These add to technical debt by adding to the burden of *“reading tons of AI generation”* [P5]. However, these challenges went both ways. P1 noted that when collaborators do not explicitly document their reasoning and decisions, AI suggestions about which directions to follow can create additional risks. Moreover, participants expressed skepticism that coding agents, unlike people, could reliably interpret intermediate results (C14) or evolving analysis logic (C15). At the same time, externalizing higher-order analytical thinking and reflection, whether by humans or AI, introduces significant cognitive overhead.

3.3.3 New challenges in HAI collaboration data analysis. Participants also expressed new challenges for how they wished to work collaboratively with fellow analysts and AI-agents. These reflections are rooted in those defined in Table 1, and represent how collaborative human-AI data analysis should evolve.

Participants repeatedly highlighted the importance of shared contextual information in collaborative notebooks, including: factual notebook details, stated and inferred analytic intent of users and AI assistants, and suggested next steps for analysis. While prior research highlights over-reliance on confident AI outputs, our study contextualizes this concern within shared notebooks where participants emphasized that facts should be presented before inferred content and clearly distinguished (P2). Preferences varied in the

amount of context shown: some participants wanted a detailed view, while others favored a lighter presentation focused on facts and relevant background. Most participants also wanted options suggesting solutions or prompt improvements, both to reduce effort and to verify whether the model understood user intent correctly (P5). To avoid disruptions, participants voiced that timing was important, particularly for AI agents: real-time suggestions were helpful for immediate tasks, but non-urgent guidance should be surfaced at natural milestones, such as when committing changes (P3), to avoid disrupting exploratory work. Subtle visual cues that integrate with existing workflows were also critical for effective presentation.

3.4 Design Guidelines

The challenges of our formative study motivate the following design guidelines for aligning shared human and AI knowledge and context in collaborative data analysis.

- **DG1: Externalize analytic intent into a shared human–AI representation.** Many challenges arise when analytic intent remains implicit (C4-7,10-15). Coding conventions address some issues, but most are context-dependent, requiring semantic understanding of workflows and collaborators’ intentions. Systems should externalize analytic intent into a shared representation that both humans and LLMs can operate on, supporting constraints from static rules to context-aware checks.
- **DG2: Reduce cognitive burden through context-aware guidance.** Coordination in data analysis requires analysts to track the notebook state and prior decisions as they progress (C1-3,7-9,14). Systems should reduce this burden by providing context-aware guidance, surfacing relevant context at the point of action so users do not have to retrieve prior context themselves.
- **DG3: Support iterative refinement of collaboration norms.** As analysis unfolds, collaboration norms, such as how data should be handled and which assumptions should guide the work, often need to change as well (C2,5-9,12,13,15). Systems should support iterative refinement of these norms so shared expectations can evolve with the analysis.

4 INTENTLINT

We developed INTENTLINT, a VSCode extension that helps data analysts maintain shared knowledge among humans and AI coding agents in collaborative data analysis. Guided by findings from our formative study, we propose and operationalize a rule-based coordination layer for AI-assisted collaborative notebooks, supported by two novel interaction mechanisms: **analytic intent scaffolding** and **prompt-time linting**. These mechanisms are enabled through shared rules, linked to collaborative challenges, which facilitate coordination and alignment across human and AI collaborators (DG3). The interface allows users to propose, review, and manage rules (Figure 2) and connects with LLM-based tools such as GitHub Copilot and Claude Code.

4.1 Core Interaction Mechanisms

A key finding from our formative study was the burden of enforcing data, code, and analysis norms across people and AI coding agents, leading to significant overhead and technical debt in managing

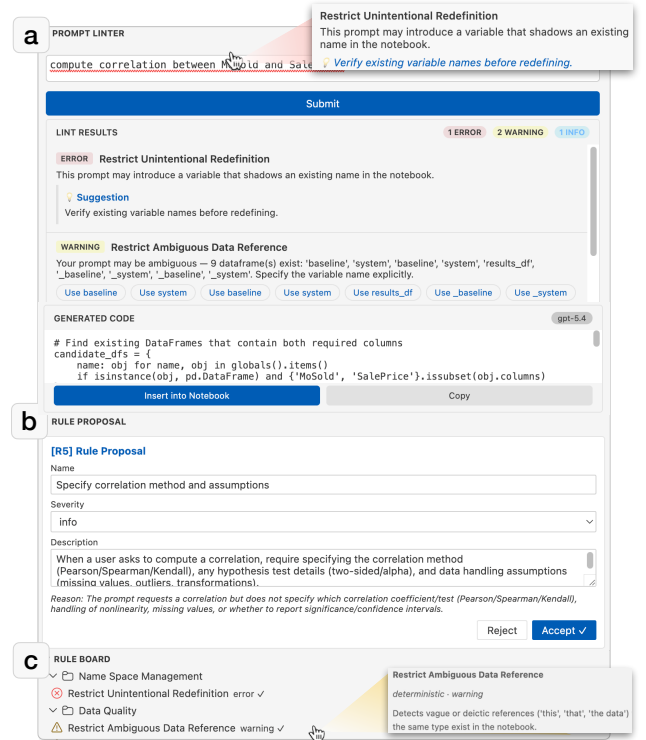


Figure 2: The INTENTLINT interface features three panels: Prompt Linter(a), Rule Proposals(b), and Rule Board(c). The prompt linter displays linting results for the current user prompt, along with explanations and suggested improvements. The Rule Proposals panel allows users to review and edit proposed rules before adding them to the system. The Rule Board displays all active and inactive rules, which users can view and manage.

evolving analyses. Participants valued documentation for orientation, but found it effortful to produce. To address these gaps, we proposed two interaction mechanisms:

Analytic Intent Scaffolding helps analysts externalize goals, assumptions, and decision rationales throughout data analysis (DG1). Rather than relying on people or AI to infer intent implicitly, users iteratively clarify and refine their objectives as the analysis evolves. We argue that structured representations, rather than free-form comments or documentation, make analytic intent explicit, persistent, and verifiable. These representations can be generated by people or AI coding agents, reducing cognitive burden for the former and making assumptions of the latter transparent. By surfacing goals, assumptions, and decisions clearly, they create a shared reference that can be revisited and built upon over time, supporting coordination, verification, and flexible engagement in collaborative analysis.

Prompt-time linting uses externalized analytic intent to provide timely, context-aware feedback during analysis (DG2). Comparing actions against shared expectations helps ensure that human and AI collaborators remain aligned with goals and assumptions. This approach brings relevant context into the moment of action,

Category	Default Rule	Resolved challenges
Namespace & Definition Management	R1: Restrict unintentional variable and function redefinitions R2: Require explicit data reference	C3 C7, C10
Dependency & Side Effect Tracking	R3: Warn about cell modification cascade R4: Require dependent cells to update R6: Check in-place mutation versions	C2 C2 C8
Knowledge & Context Preservation	R5: Require algorithmic/statistical test specification comprehension R7: Flag abandoned/alternative paths or operations R8: Flag the impact of preprocessing on target data R9: Flag pattern/threshold definition for pattern comprehension and validation	C4, C12, C14, C15 C11 C9, C12, C13 C5, C6
State & Execution Consistency	R10: Restrict executing commented out cells	C1

Table 2: Default rules instantiated in INTENTLINT and informed by the formative study challenges. Rules can be extended, modified, or removed by users, and new rules can be added or dynamically proposed by INTENTLINT. Rules persist in a notebook’s workspace, allowing them to be asynchronously enforced over multiple analysis sessions and analysts.

reducing the need for analysts to manually track prior decisions. When ambiguities or potential conflicts arise, it should surface feedback that encourages reflection on how current actions relate to prior work. Considering alternative resolutions promotes accurate, aligned, and informed decision-making without requiring analysts to track prior decisions manually.

Together, these mechanisms form a feedback loop between intent articulation and action: Analytic Intent Scaffolding makes reasoning explicit and shareable, while Prompt-Time Linting reintroduces that intent at critical moments to guide ongoing analysis.

4.2 Rules: soft constraints in collaboration

To operationalize **Analytic Intent Scaffolding** and **Prompt-time Linting** mechanisms, INTENTLINT implements shared rules that encode expectations about data, code, and analysis behavior. We chose rules as the core abstraction because they externalize tacit coordination norms into explicit, machine-readable forms without requiring formal specifications. Acting as boundary objects [70], they also provide concrete artifacts for negotiation: accepting, rejecting, or editing a rule becomes a coordination act that builds common ground (DG3). By linking rules to human and AI actions, the system can surface timely, context-aware guidance (DG2), detect potential conflicts, and maintain alignment across collaborators.

4.2.1 Rule Types and Collaborative Analysis Challenges. Based on the prior literature and formative study findings, we derived a default set of rules that address the collaborative analysis challenges. These are shown in Table 2, along with the challenges they aim to address. Users can modify, remove, or add to this core rule base (§4.3), and our user study directly assesses their utility as defaults.

Each session is instantiated by the default set of rules. These rules can be modified, removed, or deactivated during a session, and new rules can also be proposed. Rules are contained as part of the notebook’s rule board (Figure 2), and persist after an analysis session to be used by other people or AI agents at a later time.

4.2.2 Rule Structure. Rules follow a standardized template that is interpretable by both people and AI coding agents. This template provides a scaffold for expressing analytic intent, capturing assumptions, expected actions, and contextual dependencies in a structured form that can guide reflection and coordination across collaborators. An example template is below:

```
1 {
2   name: "Warn about Cell Modification Cascade"
```

```
3   author: {User A}, // system or user ID
4   level: 'info' | 'warn' | 'error',
5   description: "Detects when a user's prompt is...",
6   trigger condition: "The user's prompt...",
7   context: "Commented cell xx...",
8 }
```

The components of the rule template are as follows:

- A **rule name** describing the action that should be enforced.
- The **author** who constructed the rule, which can be a human or AI collaboration. This information is important for distinguishing between the actions of people and AI coding agents (DG1).
- A **severity level**, that encodes enforcement strength. Based on our formative study, we defined three levels: *information*, *warning*, and *error*, which respectively surface relevant collaborator context, flag potentially important issues, and indicate likely violations that should be resolved before proceeding. The level also determines how disruptive a triggered rule is, allowing intervention intensity to match the coordination need (DG2).
- A **Description** makes the rule’s reasoning transparent and verifiable. An LLM agent is designed to understand and trigger the rules based on the natural language descriptions (DG1).
- **Trigger conditions** that determine when a rule activates: rules fire only when relevant, not constantly. It distinguishes structurally detectable patterns (e.g., variable names, column references) from semantically rich conditions that require LLM interpretation, enabling the range of constraint types.
- **Context** that functions as a grounding field which records what notebook state (outputs, comments, prior code) caused the rule to be proposed. During early prototyping, we found that surfacing specific types of context (e.g., dataflow and code dependencies) improved the accuracy of LLM-based rule checking (DG2).

4.3 Enforcing Rules via Prompt-Time Linting

INTENTLINT provides an interface for prompt-based interaction, similar to emerging AI-assisted coding tools, allowing users to modify computational notebook cells with AI support. When users input a prompt, INTENTLINT automatically determines which rules should be applied, providing real-time linting information (DG2). In Figure 2, we show an example of this when a participant prompts an AI assistant to “add correlation analysis of all the variables”. INTENTLINT warns that this prompt provides an ambiguous data reference (C10) and suggests variables from the analysis to revise

the prompt to provide more information. In addition, INTENTLINT dynamically proposes a new rule, one specific to correlation analysis, that is informed by the triggering of the default and more generic R5 ('Require algorithmic/statistical test specification'; Table 2); this new rule can be rejected or accepted by the user, growing the rule base without imposing additional overhead (DG3).

Figure 1 illustrates this workflow. Bob enters a prompt, and INTENTLINT generates a structured rule that captures his analytic intent. Bob can review and accept this rule, allowing it to persist across analysis sessions as part of the shared context (DG1). At the same time, INTENTLINT uses prompt-time linting to surface previously articulated intent from the whole team, including Bob, Alice, and other team members, warning Bob when his action may conflict with prior assumptions or decisions. Bob can choose to proceed, revise his prompt, or contribute new rules of his own, collectively shaping the shared analytical norms (DG3).

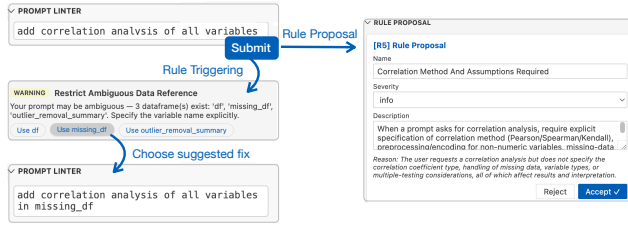


Figure 3: Rule-based Interactions: rule triggering and rule proposal. Upon prompt submission, INTENTLINT checks the prompt with shared rules and the current notebook context. A linting message with a suggested fix is displayed if a rule is triggered (left). A new rule is proposed based on the default rule and the current prompt (right).

4.3.1 Prompt Time Linting Workflow. INTENTLINT uses a multi-agent back-end to dynamically apply, propose, trigger, and manage rules. Multiple agents operate in parallel, but simultaneously return their results to the main INTENTLINT interface. In the appendix, we provide an architectural diagram as well as low-level technical details of this implementation. Here, we focus on conveying the high-level steps of this workflow.

Rule Triggering. When a user submits a prompt, INTENTLINT evaluates it against the current notebook state and a set of existing rules (DG2). A back-end ‘rule-checking’ agent interprets each rule’s trigger condition to determine whether it applies to the prompt. Because these conditions are context-dependent, the agent dynamically assesses the alignment between the prompt and relevant rules. Multiple rules may be triggered and are aggregated in the *Prompt Linter* panel, which displays the rule name, author, severity, and a contextual description of potential conflicts (Figure 3).

Suggested Prompt Updates. When a rule is triggered, INTENTLINT uses its knowledge of the current notebook state, including code cell contents, cell indices, execution state, and execution order in addition to prompt and rules, to propose prompt revisions (DG2). This is achieved through the ‘linting’ agent that reconciles this information and proposes prompt modifications. For example, in Figure 3, INTENTLINT proposes specific variables from the notebook that could help reduce the ambiguity in the user’s original prompt.

The suggested revision is surfaced automatically along with the linting message. Users can accept or decline these changes.

Propose new rules. Finally, INTENTLINT can propose new rules based on those that are triggered, enabling more granular linting over time. A rule-proposal agent suggests candidate rules derived from existing ones, pre-populated with relevant context (DG1). Users can review, refine, and add these rules to the rule board, allowing them to be applied in future interactions. This action is shown in the right-hand panel of Figure 3. The addition or modification of rules is propagated in real time to all collaborators, human and AI, ensuring that the shared rule set remains consistent.

Rule Management. As rule sets expand, INTENTLINT can support personalized enforcement of different rules. Team members can directly modify each rule as the team’s analytical conventions evolve (DG3). However, individual collaborators can also activate or deactivate any rule in their own workspace, allowing them to suppress rules that are irrelevant to their current analysis without affecting other collaborators. This distinction between team-level rule editing and individual-level activation supports shared conventions while preserving each collaborator’s autonomy.

4.4 Implementation

The primary INTENTLINT client-side interface is implemented in Typescript. The backend is implemented using Python, using FastAPI to coordinate with the client-side UI, and facilitates rule checking and prompt-time linting as well as manages the shared rule set. Additional details are in the appendix.

5 User Study

We conducted a within-subjects study to evaluate the usability of INTENTLINT, by answering the following questions: (RQ1) How do participants perceive the effectiveness of rule proposing for scaffolding their intent expression? (RQ2) How do participants perceive prompt-time linting in supporting collaboration and exploration? (RQ3) Does the system with the rule-based coordination layer change participants’ workflow? Our focus is not to evaluate if IntentLint produces better analytical outcomes, but whether it improves the collaborative analysis experience.

The study was approved by a University Ethics Board. Sessions were an hour long. Participants were compensated at a rate of \$30/hr, with pay adjusted for sessions that exceeded this time.

5.1 Procedure, Participants, and Data Analysis

5.1.1 Study Conditions & Tasks. Our study included two conditions: a baseline and INTENTLINT. The baseline condition used GitHub Copilot (agent mode). Each condition was associated with a different task, instantiated through separate pre-populated computational notebooks designed to surface the collaborative challenges. These tasks and notebooks were co-designed with two experienced data analysts to simulate representative analysis workflows. The notebooks contain expert-authored inline comments and markdown cells, which were controlled across conditions. The order of conditions and tasks was counterbalanced in the study.

Task 1: Assess whether sale timing affects price by visualizing median sale price by month and year, and testing for differences across

2006–2010. Analyze the luxury segment by comparing Fireplaces and GarageArea, evaluating alternative thresholds (e.g., top 20%), and updating preprocessing to remove near-zero variance features.

Task 2: Examine salary differences between attrition and non-attrition groups using visualization and statistical analysis. Complete preprocessing with one-hot encoding of key categorical variables, then analyze attrition patterns among high-performing employees across JobRole categories.

5.1.2 Study Procedure. Participants first provided informed consent, including permission for audio, video, and screen recording. They were then shown a demonstration of INTENTLINT and completed a set of practice questions to familiarize themselves with the interface. Participants were randomly assigned to begin with either the baseline or INTENTLINT condition. For each condition, they were given a pre-populated notebook corresponding to Task 1 or Task 2 and asked to build on existing work by completing additional analysis steps that extended prior results. Participants were encouraged to think aloud throughout the session. After 20 minutes, they switched to the other condition and task and repeated the process. At the end of the study, participants took part in a semi-structured interview, reflecting on how rule proposals and prompt-time linting influenced their approach to documenting analytic intent and building on collaborators’ work, as well as their expectations for integrating the system into their workflow and suggestions for improvement.

5.1.3 Participant Recruitment and Demographics. We recruited 16 participants (8 males, 8 females; ages 20–38) from a local university through mailing list, social media, and word-of-mouth. Potential participants were asked if they had prior experience with collaborative data analysis and AI tools such as GitHub Copilot and ChatGPT; they had to answer “yes” to both questions to participate. On a 5-point scale from 1–“a few times a year” to 4–“daily”, they reported their frequency of use of AI tools for data analysis as $Mdn = 3.5$ (See appendix for full demographic details).

5.1.4 Data Collection and Analysis. Both the baseline and INTENTLINT automatically logged various types of events based on participants’ interactions during the study. At the conclusion of each condition, participants were asked to fill out a survey of ten questions (Figure 5) on a seven-point Likert Scale (1= Strongly Disagree, 7=Strongly Agree) about their experience, the standard system usability scale, and the NASA-TLX. We summarize these results as descriptive statistics and compute statistically significant differences using a Wilcoxon signed-rank test. Session audio was transcribed and analyzed using a reflexive thematic analysis [6]. The goal of our analysis was to summarize overall participant impressions and how these support the quantitative findings.

5.2 Findings

5.2.1 Rule-checking reliability. Because IntentLint’s rule-checking interprets natural-language trigger conditions with LLM rather than applying deterministic pattern matching, we assessed how reliably it fires relative to human judgment. Following prior work that validates LLM-based classification in interactive systems by comparing system–human agreement against human–human agreement[46], two human coders independently labeled 860 prompt-rule pairs

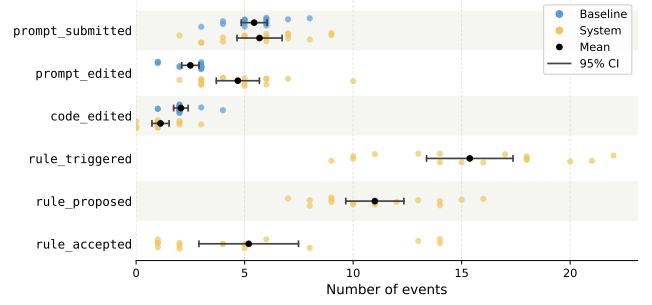


Figure 4: Distribution of system log events for 16 participants comparing the Baseline System and our system

from the user study log, judging for each pair whether the rule should fire given the prompt and notebook context. Inter-coder agreement was almost perfect (Cohen’s $\kappa = 0.92$). LLM–human agreement ranged 0.64–0.91 across rules ($M = 0.86$), indicating substantial agreement.

5.2.2 Overall rule usage and user behaviors. Participants actively engaged with all features provided in INTENTLINT. Across sessions, they interacted with linting messages, accepted auto-fix suggestions, edited prompts and generated code, reviewed and modified rule proposals, accepted or rejected proposed rules, and checked the rule board to monitor currently active rules (Figure 4).

Viewing linting suggestions and editing prompts. Across all sessions, participants encountered an average of 22 linting messages per session. The linting messages covered all categories of rules and appeared in different levels of severity. The linting messages prompted participants to reconsider whether their prompts clearly expressed analytic intent and took collaborators’ work into consideration. Consistent with this observation, we recorded substantially more prompt edits when participants used INTENTLINT ($N = 75$) compared to the baseline condition ($N = 40$). Participants also reported that their prompts were clearer and more precise with INTENTLINT (Q9 : $Median_{INTENTLINT} = 6.0$ vs. $Median_{Baseline} = 3.0$, $p < .01$). They attributed this improvement to the actionable nature of the suggestions, which reduced the effort required to revise prompts: “The suggestions helped me to add specifications for the statistical test, which I am not familiar with.” [P15].

Accepting proposed rules. Overall, participants accepted a majority of the proposed rules (86 out of 157 distinct rules), suggesting that the system’s rule generation aligned well with collaboration concerns. However, we observed notable variation in acceptance behavior across individuals. Some participants adopted a permissive approach, accepting nearly all proposed rules, viewing them as “safety nets” that could catch potential issues without requiring immediate action. This indicates that our design of the rules and linting message is not intrusive. Others were more selective, filtering out rules they considered unlikely to surface meaningful conflicts, as adding too many rules could overwhelm collaborators with constraints that distract from substantive analytic work.

Editing rules. Some participants modified the severity level of proposed rules, for example, changing them from “info” to “warning”, to emphasize rules they considered more important. This

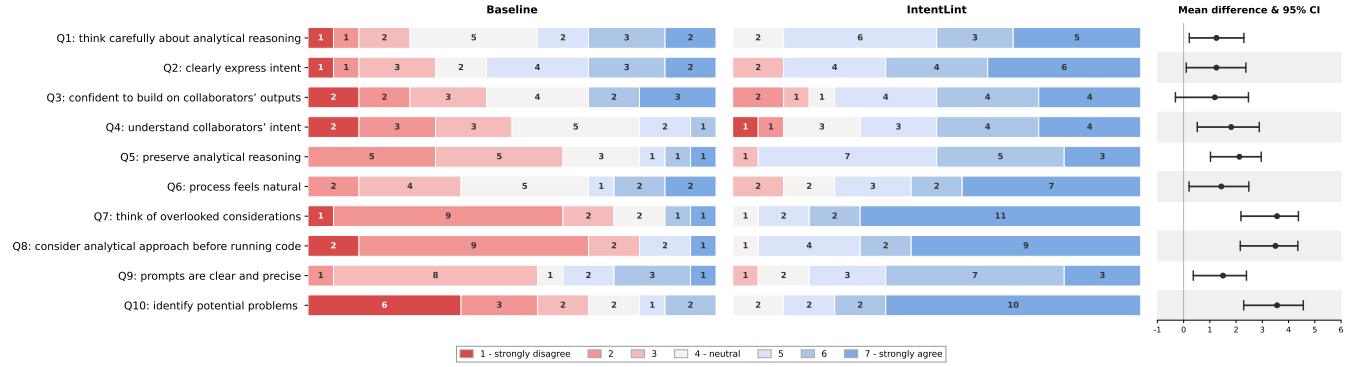


Figure 5: User perception of the utility of Baseline and INTENTLINT, measured on self-defined 7-point Likert scales (Appendix A.2.3). Dots represent the mean differences of our system compared to the Baseline. Bars indicate the 95% CI calculated using the studentized bootstrap method.

adjustment reflected participants' deliberation about which analytical practices should be more strongly enforced during collaboration. Participants also edited rule descriptions. By condensing the wording and removing potentially ambiguous or misleading phrases, they aimed to reduce the effort required for collaborators to understand the rules and to ensure the intended guidance was communicated clearly. Several participants also noted that clearer rule descriptions could help AI models better interpret the underlying analytical intent, reducing the risk of misalignment in subsequent prompts or generated analyses. These behaviors indicate that the structured rule representation prompted participants to actively reflect on how their analytic intent should be articulated and communicated to both collaborators and AI agents.

System Usability & Perceived Cognitive Load. To measure the usability of INTENTLINT, we computed the SUS scores based on the UMUX-LITE [49]. Both INTENTLINT and Baseline have sufficiently reasonable usability scores, with an average of ($M = 77.08$) for INTENTLINT and ($M = 68.75$) for Baseline. We also used NASA-TLX to measure participants' perceptions of the cognitive workload of using the systems. The overall perceived workload, obtained by averaging all six raw NASA-TLX scores (with the "Performance" measure inverted), did not show a significant difference between INTENTLINT and Baseline ($Mdn = 2.67$ vs 2.92 , $p = .28$).

Failure case analysis. We also identified several failure modes of rule triggering. One common issue arose from too strict variable-name constraints: variable names reused independently in two cells can trigger a cross-user conflict, even though the scopes never interacted (P5, P7). Another issue arose from coarse-grained dependency tracking: false dependency warnings were triggered by edits to unrelated lines, due to cell-level rather than line-level dependency tracking (P8, P10). These findings suggest that future conflict detection mechanisms should incorporate finer-grained code analysis and contextual awareness to reduce false positives.

5.2.3 Intent scaffolding encouraged deeper thinking. The structured intent scaffolding prompted participants to reflect more carefully on their analytic strategies and assumptions. Participants reported thinking more carefully about analytical reasoning (Q1 : $Mdn_I = 5.5$ vs. $Mdn_B = 4.0$, $p < .05$). As shown in Figure 4, participants

made more prompt edits to articulate intent using INTENTLINT. Several participants went beyond accepting the system's proposed rules and proactively composed rules themselves: "Some rules are not proposed by the system, but I want to define a rule when checking the code." [P13] This behavior suggests that the scaffolding mechanism helped participants externalize their own analytic standards (Q2 : $Mdn_I = 6.0$ vs. $Mdn_B = 5.0$, $p < .05$). Participants also reported becoming more deliberate in prompt formulation (Q9 : $Mdn_I = 6.0$ vs. $Mdn_B = 3.0$, $p < .01$). For example, by including more detailed information about variables and statistical tests, reflecting a more explicit consideration of analysis requirements.

The linting messages also helped participants reason about the state of the collaborative analysis by surfacing issues and highlighting where attention was needed (Q5 : $Mdn_I = 5.5$ vs. $Mdn_B = 3.0$, $p < .001$). As one participant noted, "It gave me the red flags in advance, so I know sections that we were supposed to alter" [P2]. In this way, INTENTLINT helped participants interpret prior work and decide where modifications were appropriate. Rule proposals further supported this reflection by surfacing analytic considerations participants might ignore. These interactions illustrate how structured rule feedback encouraged participants to reason about dependencies and methodological correctness in the analysis workflow.

Participants also compared the structured scaffolding with existing mechanisms for communicating intent, such as inline code comments (Q4 : $Mdn_I = 5.5$ vs. $Mdn_B = 3.5$, $p < .005$). Several participants expressed skepticism toward free-text comments, particularly in collaborative environments where comments may be incomplete, outdated, or automatically generated. As one participant explained, "most comments are AI-generated anyway" [P3]. Instead, participants described the rule-based representations in INTENTLINT as a more dependable way to capture and communicate analytic expectations. These responses suggest that structured intent scaffolding can complement informal documentation by providing explicit, actionable representations of analytic intent that are more tightly integrated with the workflow.

Beyond supporting participants' interpretation of prior work, structured intent representations also influenced how they contributed to the shared workspace. Participants described adapting their own behavior with future collaborators in mind by making

their analytic intent more explicit and curating what should become shared coordination knowledge. For example, P3 explained that they deliberately expressed their intent more clearly because “*collaborators may not have time to thoroughly review the notebook*” [P3]. Participants also became more selective about externalizing rules, rejecting later proposals that were already subsumed by broader existing rules to “*avoid flooding the shared coordination space*” [P11]. These observations suggest that intent scaffolding not only helps collaborators understand one another’s work, but also encourages contributors to maintain concise and collaboration-oriented representations of analytic intent.

We also asked participants to explain why they rejected proposed rules. Some participants explained that they tended to reject rules that represent one-time decisions rather than reusable coordination rules, or being subsumed by broader existing rules (P8, P13). Other participants found that some proposed rules are overly strict for the exploratory nature of EDA task they’re working on, so they preferred to avoid unnecessarily constraining the team’s workflow and to “*avoid constraining too much for the whole team and annoying collaborators*” [P7]. These responses suggest that effective rule recommendation should prioritize broadly applicable coordination practices while accounting for the flexibility and evolving nature of collaborative data analysis.

5.2.4 Prompt-time linting surfaces issues early. All participants found prompt-time linting helpful for detecting potential errors and conflicts before adding code to the shared workspace ($Q10 : Mdn_1 = 7.0$ vs. $Mdn_B = 2.0$, $p < .001$). For instance, P15 highlighted the difference it made: “*detecting the error was impossible before.*” [P15]. After resolving a variable conflict flagged by the extension, some participants reported increased trust in the generated code: “*Now I have the confidence to say that the generated responses look good by just looking at these panels and warnings.*” [P15]. Beyond error detection, linting encouraged participants to review prompts and generated code, verify alignment with collaborators, and check data consistency with contextual explanations ($Q7 : Mdn_1 = 7.0$ vs. $Mdn_B = 2.0$, $p < .001$). As P4 noted: “*Sometimes I just forget to check the data, and the agent gets back to me with context and explanations.*” [P4] It also reduces the effort required to resolve conflicts later in the workflow, as P6 mentioned: “*It can reduce time needed for completing a pull-request.*” [P6] Although participants could deactivate or edit rules, only one chose to do so. Most perceived rules as non-intrusive and functioning as a safety net, reducing the need for deactivation.

5.2.5 Change of workflow: reflection on analytic intent and collaboration awareness. Overall, participants demonstrated increased reflection on their analytic intent and whether their prompts clearly articulated that intent. This shift is consistent with the event logs (Figure 4), which show more prompt revisions when using INTENTLINT ($N = 75$ vs. 40), and fewer direct code edits ($N = 18$ vs. 33). These patterns indicate a shift in effort from code-focused activities toward the formulation and articulation of analytic intent. Participants also dynamically adjusted rule severity over time, reflecting evolving confidence in their analyses and rule authoring. When initially unfamiliar with the dataset or collaborative context, participants tended to assign rules a lower (informational) severity. As their understanding deepened, they increasingly escalated rule

severity to signal stronger expectations or constraints. For example, P14 explained their decision to upgrade a rule: “*I think the choice of statistical tests is important, I want to warn my collaborators*” [P14].

Participants also noted a shift in workload distribution. By offloading tasks such as data validation and conflict detection to INTENTLINT, they were able to focus more on iterating over analytic intent and decision-making. As P7 described, “*The system can handle the potential conflicts for me*” [P7]. This suggests that INTENTLINT supports a form of cognitive offloading that frees analysts to engage more deeply with higher-level reasoning. In addition, participants identified specific moments in the workflow where the system was particularly valuable: during onboarding and when creating new data frames. These moments are characterized by uncertainty and a need for alignment, highlighting the role of INTENTLINT in establishing shared understanding early in analytic processes. In addition, participants surfaced potential use cases and design considerations: Some described using the linter panel as a space for experimenting with and testing analytic assumptions (e.g., “*The linter panel is like a sandbox for prompts*” [P11]), extending its role beyond coordination into exploratory reasoning. Others expressed a desire for greater control over rule management, such as introducing role-based permissions (e.g., “*add an admin/user distinction for rule management*” [P8]). These insights reveal the opportunities for extending INTENTLINT to better support both flexible exploration and structured collaboration.

6 Discussion

We reflect on the design process and study findings, discussing how intent scaffolding and prompt-time linting integrate into natural analytical workflows and change collaborative dynamics. We also outline future directions and design opportunities for supporting knowledge sharing in multi-human, multi-agent collaboration.

6.1 Human-AI collaborative intent clarification

Prior work has explored AI-generated annotations for documentation in collaborative settings, often through prompt-based nudges. However, these are frequently hallucinated, lack grounding in analytic intent, and remain context-independent, limiting their effectiveness [24, 68, 79]. In contrast, INTENTLINT positions AI as a collaborator in intent clarification rather than a documentation author, helping analysts articulate and refine their own intent [89]. This distinction is critical in addressing trust challenges in AI-generated artifacts [7, 26]. Simply generating more documentation may not help and can even reduce engagement [56]. Instead, our approach emphasizes AI-mediated articulation: by requiring analysts to refine rules, intent is collaboratively clarified, improving both its credibility and usability. Our findings further show that participants engaged deeply in rule creation, balancing usefulness against over-constraint. The low rate of rule deactivation suggests this process filtered out ineffective rules. We interpret this as productive friction, prompting users to reflect and not to accept AI outputs uncritically [7, 10, 15]. Overall, this points to a broader shift from AI generating artifacts to supporting higher-fidelity articulation of human reasoning in collaborative analysis.

6.2 Active warnings vs. passive documentation

Documentation, such as inline comments, is important for capturing analytic goals and rationale [79], but often fails as a coordination mechanism because it lacks a feedback loop for establishing mutual understanding [13]. In practice, its passive nature limits effectiveness: analysts may avoid reading comments due to cognitive effort, especially when they are lengthy or misaligned with the current analysis goals, creating a gap between available and used knowledge [29, 61]. Moreover, documentation is decoupled from action, requiring users to seek it out rather than surfacing it when needed [78]. As a result, even well-written comments may be overlooked during fast-paced, iterative workflows.

Our findings suggest that prompt-time linting addresses these limitations through active, contextual interventions. By surfacing relevant considerations at the moment of prompt submission, it reduces retrieval effort and encourages reflection and revision of intent. In this way, INTENTLINT not only externalizes intent but operationalizes it as actionable guidance. However, such interventions must be carefully calibrated to avoid disruption. Rather than replacing documentation, this highlights a complementary relationship between persistent records and in-the-moment coordination.

6.3 Other forms of knowledge sharing

Knowledge sharing is important to collaboration, but LLM integration reshapes both what is shared and who it is for [23, 78]. While existing mechanisms like Cursor Rules and Claude Skills encode knowledge for AI consumption, our approach frames rules as human-oriented artifacts that surface assumptions and make analytic intent explicit. This supports coordination, accountability, and learning, with linting operationalizing rules as actionable checks during AI-assisted work. At the same time, our extension is not intended to replace existing LLM tools, but to interoperate with them by augmenting how intent is articulated and negotiated. This enables interoperability across environments without modifying underlying models, pointing to a broader design space where human-readable scaffolds coexist with AI-oriented configurations. For instance, rules could be translated into prompt templates, linked to comments, or integrated with version control. Other mechanisms, such as shared prompt libraries, intent-aware annotations, or lightweight review checkpoints, could similarly embed explicit intent and feedback loops into workflows. Finally, the idea of structured communication through rules aligns with existing practices in larger organizations, where teams rely on templates and standardized processes to coordinate work at scale. Embedding such constraints at generation time, rather than relying on post hoc review, helps mitigate silent analytical drift as collaboration scales.

6.4 Limitation and Future work

Our study with 16 participants in a controlled lab setting ensured consistency but limited insight into long-term use. Following established HCI evaluations of asynchronous notebook-based collaboration, participants worked individually on a pre-populated notebook that represented prior collaborators' work. This design enabled controlled comparisons between conditions while evaluating the usability of our design. However, it does not capture emergent team

dynamics that arise in sustained multi-user collaboration. In particular, our evaluation cannot fully examine how persistent rules evolve, become stale, conflict, or support real collaboration over time. In practice, collaborative data analysis unfolds over longer periods and often involves asynchronous contributions, where analysts inherit and extend work they did not produce. A longitudinal deployment would enable investigation of how rules evolve over time, whether teams develop shared conventions for rule authoring, and how intent scaffolds remain effective as analytic goals shift. The current design also assumes a flat collaboration model, whereas real teams often have differentiated roles. Supporting role-based permissions could better reflect organizational practice. Finally, as the system is limited to notebooks, extending prompt-time linting to other communication channels (e.g., Slack) could surface relevant rules or potential conflicts directly within team discussions, supporting more cohesive, cross-channel workflows.

7 Conclusion

We introduce a rule-based coordination layer for AI-assisted collaborative notebooks with two interaction mechanisms, analytic intent scaffolding and prompt-time linting, for supporting shared understanding and early detection of misalignment in human–AI collaborative data analysis. By externalizing analysts' intent into a shared human–AI representation and checking prompts against that context before code generation, these mechanisms make analytic intent more explicit and actionable. A proof-of-concept system and user study suggest that these mechanisms help analysts understand prior work, reflect on their intent, and identify conflicts earlier. Together, our work highlights a promising direction for future research: designing collaborative AI systems that actively mediate coordination within a team.

Acknowledgments

This work was made possible by the Natural Sciences and Engineering Research Council of Canada (NSERC) Discovery Grant #RGPIN-2025-01552, NSERC Discovery Grant #RGPIN-2020-03966, NSERC Research, Technology, and Instruments Grant #RTI-2026-00134, the Ontario Early Researcher Award (ERA) #ER24-18-222, and the Canada Foundation for Innovation (CFI) John R. Evans Leaders Fund (JELF) #42371.

References

- [1] [n. d.]. Ames Housing Dataset — kaggle.com. <https://www.kaggle.com/datasets/shashankneerthapa/ames-housing-dataset/data>. [Accessed 25-03-2026].
- [2] Sara Alspaugh, Nava Zokaei, Andrea Liu, Cindy Jin, and Marti A Hearst. 2018. Futzing and moseying: Interviews with professional data analysts on exploration practices. *IEEE transactions on visualization and computer graphics* 25, 1 (2018), 22–31.
- [3] Leilani Battle and Jeffrey Heer. 2019. Characterizing exploratory visual analysis: A literature review and evaluation of analytic provenance in tableau. In *Computer graphics forum*, Vol. 38. Wiley Online Library, 145–159.
- [4] Mary Beth Kery and Brad A. Myers. 2017. Exploring exploratory programming. In *2017 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*. 25–29. doi:10.1109/VLHCC.2017.8103446
- [5] Virginia Braun and Victoria Clarke. 2006. Using thematic analysis in psychology. *Qualitative Research in Psychology* 3, 2 (2006), 77–101. doi:10.1191/1478088706qp063oa
- [6] Virginia Braun and Victoria Clarke. 2019. Reflecting on reflexive thematic analysis. *Qualitative research in sport, exercise and health* 11, 4 (2019), 589–597.
- [7] Zana Bućinca, Maja Barbara Malaya, and Krzysztof Z Gajos. 2021. To trust or to think: cognitive forcing functions can reduce overreliance on AI in AI-assisted

- decision-making. *Proceedings of the ACM on Human-computer Interaction* 5, CSCW1 (2021), 1–21.
- [8] Worawalan Chatlatanagulchai, Hao Li, Yutaro Kashiwa, Brittany Reid, Kundjansith Thonglek, Pattara Leelapruete, Arnon Rungsawang, Bundit Manaskasemsak, Bram Adams, Ahmed E Hassan, et al. 2025. Agent READMEs: An Empirical Study of Context Files for Agentic Coding. *arXiv preprint arXiv:2511.12884* (2025).
- [9] Souti Chattopadhyay, Ishita Prasad, Austin Z Henley, Anita Sarma, and Titus Barik. 2020. What's wrong with computational notebooks? Pain points, needs, and design opportunities. In *Proceedings of the 2020 CHI conference on human factors in computing systems*. 1–12.
- [10] Zeya Chen and Ruth Schmidt. 2024. Exploring a behavioral model of "positive friction" in human-AI interaction. In *International Conference on Human-Computer Interaction*. Springer, 3–22.
- [11] Davide Chicco, Luca Oneto, and Erica Tavazzi. 2022. Eleven quick tips for data cleaning and feature engineering. *PLOS Computational Biology* 18, 12 (2022), e1010718.
- [12] Bhavya Chopra, Ananya Singha, Anna Fariha, Sumit Gulwani, Chris Parnin, Ashish Tiwari, and Austin Z Henley. 2023. Conversational challenges in ai-powered data science: Obstacles, needs, and design opportunities. *arXiv preprint arXiv:2310.16164* (2023).
- [13] HH Clark. 1991. Grounding in communication. *Perspectives on socially shared cognition/American Psychological Association* (1991).
- [14] Herbert H Clark and Deanna Wilkes-Gibbs. 1986. Referring as a collaborative process. *Cognition* 22, 1 (1986), 1–39.
- [15] Anna L Cox, Sandy JJ Gould, Marta E Cecchinato, Ioanna Iacovides, and Ian Renfree. 2016. Design frictions for mindful interactions: The case for microboundaries. In *Proceedings of the 2016 CHI conference extended abstracts on human factors in computing systems*. 1389–1397.
- [16] Paul Dourish and Victoria Bellotti. 1992. Awareness and coordination in shared workspaces. In *Proceedings of the 1992 ACM conference on Computer-supported cooperative work*. 107–114.
- [17] Ilker Etikan, Sulaiman Abubakar Musa, Rukayya Sunusi Alkassim, et al. 2016. Comparison of convenience sampling and purposive sampling. *American journal of theoretical and applied statistics* 5, 1 (2016), 1–4.
- [18] KJ Feng, Tzu-Sheng Kuo, Quan Ze, Inyoung Cheong, Kenneth Holstein, Amy X Zhang, et al. 2025. PolicyPad: Collaborative Prototyping of LLM Policies. *arXiv preprint arXiv:2509.19680* (2025).
- [19] KJ Feng, Kevin Pu, Matt Latzke, Tal August, Pao Siangliulue, Jonathan Bragg, Daniel S Weld, Amy X Zhang, and Joseph Chee. 2025. Cocoa: Co-planning and co-execution with ai agents. *arXiv preprint arXiv:2412.10999* (2025).
- [20] Li Feng, Ryan Yen, Yuzhe Yu, Mingming Fan, Jian Zhao, and Zhichong Lu. 2024. CoPrompt: Supporting Prompt Sharing and Referring in Collaborative Natural Language Programming. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) (CHI '24). Association for Computing Machinery, New York, NY, USA, Article 934, 21 pages.
- [21] Jie Gao, Kenny Tsu Wei Choo, Junming Cao, Roy Ka-Wei Lee, and Simon Perrault. 2023. CoAlCoder: Examining the effectiveness of AI-assisted human-to-human collaboration in qualitative analysis. *ACM Transactions on Computer-Human Interaction* 31, 1 (2023), 1–38.
- [22] Tong Gao, Mira Dontcheva, Eytan Adar, Zhicheng Liu, and Karrie G Karahalios. 2015. Datatone: Managing ambiguity in natural language interfaces for data visualization. In *Proceedings of the 28th annual acm symposium on user interface software & technology*. 489–500.
- [23] Ken Gu, Madeleine Grundle-McLaughlin, Andrew McNutt, Jeffrey Heer, and Tim Althoff. 2024. How Do Data Analysts Respond to AI Assistance? A Wizard-of-Oz Study. In *Proc. CHI'24* (Honolulu, HI, USA) (CHI '24). Association for Computing Machinery, New York, NY, USA, Article 1015, 22 pages. <https://doi.org/10.1145/3613904.3641891>
- [24] Ken Gu, Ruoxi Shang, Tim Althoff, Chenglong Wang, and Steven M. Drucker. 2024. How Do Analysts Understand and Verify AI-Assisted Data Analyses?. In *Proc. CHI'21*. Article 748, 22 pages. doi:10.1145/3613904.3642497
- [25] Ken Gu, Ruoxi Shang, Ruijen Jiang, Keying Kuang, Richard-John Lin, Donghe Lyu, Yue Mao, Youran Pan, Teng Wu, Jiaqian Yu, et al. 2024. Blade: Benchmarking language model agents for data-driven science. *arXiv preprint arXiv:2408.09667* (2024).
- [26] Ian Guelman, Arthur Gregório Leal, Laerte Xavier, and Marco Tulio Valente. 2024. On the Quality of AI-Generated Source Code Comments: A Comprehensive Evaluation. *arXiv preprint arXiv:2408.14007* (2024).
- [27] Carl Gutwin and Saul Greenberg. 2002. A descriptive framework of workspace awareness for real-time groupware. *Computer Supported Cooperative Work (CSCW)* 11, 3 (2002), 411–446.
- [28] Robert Haase. 2025. Towards transparency and knowledge exchange in AI-assisted data analysis code generation. *Nature Computational Science* 5, 4 (2025), 271–272.
- [29] Edward W He, Vanessa Bellotti, Alexandra Scott, Jiahao Xu, Ashley Suh, Jen Rogers, and Remco Chang. 2025. The Effects of Belief Elicitation in Visual Data Analysis: A Longitudinal Classroom Study. *IEEE Transactions on Visualization and Computer Graphics* (2025).
- [30] Andrew Head, Fred Hohman, Titus Barik, Steven M. Drucker, and Robert DeLine. 2019. Managing Messes in Computational Notebooks. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland UK) (CHI '19). Association for Computing Machinery, New York, NY, USA, 1–12. doi:10.1145/3290605.3300500
- [31] Jeffrey Heer and Maneesh Agrawala. 2007. Design considerations for collaborative visual analytics. In *2007 IEEE symposium on visual analytics science and technology*. IEEE, 171–178.
- [32] Oliver Huang, Muhammad Fatir, Steven Luo, Sangho Suh, Hariharan Subramonyam, and Carolina Nobre. 2025. Narrative Scaffolding: A Narrative-First Framework for Data-Driven Sensemaking. *arXiv preprint arXiv:2512.18920* (2025).
- [33] Mohammed Saidul Islam, Md Tahmid Rahman Laskar, Md Rizwan Parvez, Enamul Hoque, and Shafiq Joty. 2024. DataNarrative: Automated Data-Driven Storytelling with Visualizations and Texts. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (Eds.). Association for Computational Linguistics, Miami, Florida, USA, 19253–19286. doi:10.18653/v1/2024.emnlp-main.1073
- [34] Shaokang Jiang and Daye Nam. 2025. An Empirical Study of Developer-Provided Context for AI Coding Assistants in Open-Source Projects. *arXiv preprint arXiv:2512.18925* (2025).
- [35] Alex Kale, Ziyang Guo, Xiao Li Qiao, Jeffrey Heer, and Jessica Hullman. 2023. Evm: Incorporating model checking into exploratory visual analysis. *IEEE Transactions on Visualization and Computer Graphics* 30, 1 (2023), 208–218.
- [36] Sean Kandel, Andreas Paepcke, Joseph M Hellerstein, and Jeffrey Heer. 2012. Enterprise data analysis and visualization: An interview study. *IEEE transactions on visualization and computer graphics* 18, 12 (2012), 2917–2926.
- [37] Sean Kandel, Ravi Parikh, Andreas Paepcke, Joseph M Hellerstein, and Jeffrey Heer. 2012. Profiler: Integrated statistical analysis and visualization for data quality assessment. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*. 547–554.
- [38] Majeed Kazemitabaar, Jack Williams, Ian Drosos, Tovi Grossman, Austin Zachary Henley, Carina Negreanu, and Advait Sarkar. 2024. Improving steering and verification in AI-assisted data analysis with interactive task decomposition. In *Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology*. 1–19.
- [39] Mary Beth Kery, Bonnie E John, Patrick O'Flaherty, Amber Horvath, and Brad A Myers. 2019. Towards effective foraging by data scientists to find past analysis choices. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. 1–13.
- [40] Mary Beth Kery and Brad A Myers. 2018. Interactions for untangling messy history in a computational notebook. In *2018 IEEE symposium on visual languages and human-centric computing (VL/HCC)*. IEEE, 147–155.
- [41] Mary Beth Kery, Marissa Radensky, Mahima Arya, Bonnie E. John, and Brad A. Myers. 2018. The Story in the Notebook: Exploratory Data Science using a Literate Programming Tool. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (Montreal QC, Canada) (CHI '18). Association for Computing Machinery, New York, NY, USA, 1–11. doi:10.1145/3173574.3173748
- [42] David Kirsh. 2010. Thinking with external representations. *AI & society* 25, 4 (2010), 441–454.
- [43] Laura Koesten, Emilia Kacprzak, Jeni Tennison, and Elena Simperl. 2019. Collaborative Practices with Structured Data: Do Tools Support What Users Need?. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland UK) (CHI '19). Association for Computing Machinery, New York, NY, USA, 1–14. <https://doi.org/10.1145/3290605.3300330>
- [44] Ratanond Koonchanok, Parul Baser, Abhinav Sikharam, Nirmal Kumar Raveendranath, and Khairi Reda. 2021. Data prophecy: Exploring the effects of belief elicitation in visual analytics. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. 1–12.
- [45] Ratanond Koonchanok, Gauri Yatindra Tawde, Gokul Ragunandhan Narayanasamy, Shalmali Walimbe, and Khairi Reda. 2023. Visual belief elicitation reduces the incidence of false discovery. In *Proceedings of the 2023 CHI conference on human factors in computing systems*. 1–17.
- [46] Michelle S Lam, Fred Hohman, Dominik Moritz, Jeffrey P Bigham, Kenneth Holstein, and Mary Beth Kery. 2025. Policy Maps: Tools for Guiding the Unbounded Space of LLM Behaviors. In *Proceedings of the 38th Annual ACM Symposium on User Interface Software and Technology*. 1–24.
- [47] Florian Lehmann, Krystsina Shauchenka, and Daniel Buschek. 2025. Collaborative Document Editing with Multiple Users and AI Agents. *arXiv preprint arXiv:2509.11826* (2025).
- [48] Clayton Lewis. 1982. Using the 'thinking-aloud' method in cognitive interface design. *Research Report RC9265*, IBM TJ Watson Research Center (1982).
- [49] James R Lewis, Brian S Utesch, and Deborah E Maher. 2013. UMUX-LITE: when there's no time for the SUS. In *Proceedings of the SIGCHI conference on human factors in computing systems*. 2099–2102.
- [50] Jianping Kelvin Li, Shenyu Xu, Yecong Ye, and Kwan-Liu Ma. 2020. Resolving conflicting insights in asynchronous collaborative visual analysis. In *Computer Graphics Forum*, Vol. 39. Wiley Online Library, 497–509.

- [51] Pingchuan Ma, Rui Ding, Shuai Wang, Shi Han, and Dongmei Zhang. 2023. InsightPilot: An LLM-Empowered Automated Data Exploration System. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, Yansong Feng and Els Lefever (Eds.). Association for Computational Linguistics, Singapore, 346–352. doi:10.18653/v1/2023.emnlp-demo.31
- [52] Paula Maddigan and Teo Susnjak. 2023. Chat2vis: Generating data visualizations via natural language using chatgpt, codex and gpt-3 large language models. *Ieee Access* 11 (2023), 45181–45193.
- [53] Fangwen Mu, Lin Shi, Song Wang, Zhuohao Yu, Binqun Zhang, ChenXue Wang, Shichao Liu, and Qing Wang. 2024. Clarifygpt: A framework for enhancing llm-based code generation via requirements clarification. *Proceedings of the ACM on Software Engineering* 1, FSE (2024), 2332–2354.
- [54] Michael Muller, Ingrid Lange, Dakuo Wang, David Piorkowski, Jason Tsay, Q Vera Liao, Casey Dugan, and Thomas Erickson. 2019. How data science workers work with data: Discovery, capture, curation, design, creation. In *Proceedings of the 2019 CHI conference on human factors in computing systems*. 1–15.
- [55] Donald A Norman. 1988. *The psychology of everyday things*. Basic books.
- [56] Raja Parasuraman and Victor Riley. 1997. Humans and automation: Use, misuse, disuse, abuse. *Human factors* 39, 2 (1997), 230–253.
- [57] Soya Park, April Yi Wang, Ban Kwas, Q Vera Liao, David Piorkowski, and Marina Danilevsky. 2021. Facilitating knowledge sharing from domain experts to data scientists for building nlp models. In *Proceedings of the 26th International Conference on Intelligent User Interfaces*. 585–596.
- [58] Samir Passi and Steven J Jackson. 2018. Trust in data science: Collaboration, translation, and accountability in corporate data science projects. *Proceedings of the ACM on human-computer interaction* 2, CSCW (2018), 1–28.
- [59] João Felipe Pimentel, Leonardo Murta, Vanessa Braganholo, and Juliana Freire. 2021. Understanding and improving the quality and reproducibility of Jupyter notebooks. *Empirical Software Engineering* 26, 4 (2021), 65.
- [60] Peter Pirolli and Stuart Card. 1999. Information foraging. *Psychological review* 106, 4 (1999), 643.
- [61] Luigi Quaranta, Fabio Calefato, and Filippo Lanubile. 2022. Eliciting best practices for collaboration with computational notebooks. *Proceedings of the ACM on Human-Computer Interaction* 6, CSCW1 (2022), 1–41.
- [62] Eric D Ragan, Alex Endert, Jibonananda Sanyal, and Jian Chen. 2015. Characterizing provenance in visualization and data analysis: an organizational framework of provenance types and purposes. *IEEE transactions on visualization and computer graphics* 22, 1 (2015), 31–40.
- [63] Mizanur Rahman, Amran Bhuian, Mohammed Saidul Islam, Md Tahmid Rahman Laskar, Ridwan Mahbub, Ahmed Masry, Shafiq Joty, and Enamul Hoque. 2025. Llm-based data science agents: A survey of capabilities, challenges, and future directions. *arXiv preprint arXiv:2510.04023* (2025).
- [64] Dhivyabharathi Ramasamy, Cristina Sarasua, Alberto Bacchelli, and Abraham Bernstein. 2023. Visualising data science workflows to support third-party notebook comprehension: an empirical study. *Empirical Software Engineering* 28, 3 (2023), 58.
- [65] Anthony C Robinson. 2008. Collaborative synthesis of visual analytic results.. In *IEEE VAST*. 67–74.
- [66] Jen Rogers and Anamaria Crisan. 2023. Tracing and Visualizing Human-ML/AI Collaborative Processes through Artifacts of Data Work. In *Proc. CHI'23*. Article 837, 22 pages. doi:10.1145/3544548.3580819
- [67] Adam Rule, Aurélien Tabard, and James D Hollan. 2018. Exploration and explanation in computational notebooks. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. 1–12.
- [68] Jakob Schoeffler, Maria De-Arteaga, and Niklas Kuehl. 2024. Explanations, fairness, and appropriate reliance in human-AI decision-making. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems*. 1–18.
- [69] Leixian Shen, Haotian Li, Yun Wang, and Huamin Qu. 2024. From Data to Story: Towards Automatic Animated Data Video Creation with LLM-Based Multi-Agent Systems. In *2024 IEEE VIS Workshop on Data Storytelling in an Era of Generative AI (GEN4DS)*. 20–27. doi:10.1109/GEN4DS63889.2024.00008
- [70] Susan Leigh Star and James R Griesemer. 1989. Institutional ecology, translations' and boundary objects: Amateurs and professionals in Berkeley's Museum of Vertebrate Zoology, 1907-39. *Social studies of science* 19, 3 (1989), 387–420.
- [71] Gareth Terry, Nikki Hayfield, Victoria Clarke, Virginia Braun, et al. 2017. Thematic analysis. *The SAGE handbook of qualitative research in psychology* 2, 17–37 (2017), 25.
- [72] Yuan Tian, Weiwei Cui, Dazhen Deng, Xinjing Yi, Yurun Yang, Haidong Zhang, and Yingcai Wu. 2024. Chartgpt: Leveraging llms to generate charts from abstract natural language. *IEEE Transactions on Visualization and Computer Graphics* 31, 3 (2024), 1731–1745.
- [73] Yuan Tian, Chuhan Zhang, Xiaotong Wang, Sitong Pan, Weiwei Cui, Haidong Zhang, Dazhen Deng, and Yingcai Wu. 2025. ReSpark: Leveraging Previous Data Reports as References to Generate New Reports with LLMs. In *Proceedings of the 38th Annual ACM Symposium on User Interface Software and Technology*. 1–18.
- [74] Melanie Tory, Lyn Bartram, Brittany Fiore-Gartland, and Anamaria Crisan. 2023. Finding Their Data Voice: Practices and Challenges of Dashboard Users. *IEEE Computer Graphics and Applications* 43, 1 (2023), 22–36.
- [75] Priyan Vaithilingam, Munyeong Kim, Frida-Cecilia Acosta-Parenteau, Daniel Lee, Amine Mhedhbi, Elena L Glassman, and Ian Arawjo. 2025. Semantic Commit: Helping Users Update Intent Specifications for AI Memory at Scale. In *Proceedings of the 38th Annual ACM Symposium on User Interface Software and Technology*. 1–18.
- [76] Eric A Vance, Jessica L Alzen, and Heather S Smith. 2022. Creating shared understanding in statistics and data science collaborations. *Journal of Statistics and Data Science Education* 30, 1 (2022), 54–64.
- [77] April Wang, Zihan Wu, Christopher Brooks, and Steve Oney. 2024. Don't step on my toes: resolving editing conflicts in real-time collaboration in computational notebooks. In *Proceedings of the 1st ACM/IEEE Workshop on Integrated Development Environments*. 47–52.
- [78] April Yi Wang, Anant Mittal, Christopher Brooks, and Steve Oney. 2019. How Data Scientists Use Computational Notebooks for Real-Time Collaboration. *Proc. ACM Hum.-Comput. Interact.* 3, CSCW, Article 39 (Nov. 2019), 30 pages.
- [79] April Yi Wang, Dakuo Wang, Jaimie Drozdal, Michael Muller, Soya Park, Justin D Weisz, Xuye Liu, Lingfei Wu, and Casey Dugan. 2022. Documentation matters: Human-centered ai system to assist data science code documentation in computational notebooks. *ACM Transactions on Computer-Human Interaction* 29, 2 (2022), 1–33.
- [80] April Yi Wang, Zihan Wu, Christopher Brooks, and Steve Oney. 2020. Callisto: Capturing the "Why" by Connecting Conversations with Computational Narratives. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. 1–13.
- [81] Chenglong Wang, John Thompson, and Bongshin Lee. 2023. Data Formula- tor: AI-powered Concept-driven Visualization Authoring. *IEEE Transactions on Visualization and Computer Graphics* (2023).
- [82] Dakuo Wang, Justin D. Weisz, Michael Muller, Parikshit Ram, Werner Geyer, Casey Dugan, Yla Tausczik, Horst Samulowitz, and Alexander Gray. 2019. Human-AI Collaboration in Data Science: Exploring Data Scientists' Perceptions of Automated AI. *Proc.CSCW'19* 3, Article 211 (Nov. 2019), 24 pages. doi:10.1145/3359313
- [83] Sam Wiseman, Stuart M Shieber, and Alexander M Rush. 2017. Challenges in data-to-document generation. In *Proceedings of the 2017 conference on empirical methods in natural language processing*. 2253–2263.
- [84] Liwenhan Xie, Chengbo Zheng, Haijun Xia, Huamin Qu, and Chen Zhu-Tian. 2024. WaitGPT: Monitoring and Steering Conversational LLM Agent in Data Analysis with On-the-Fly Code Visualization. In *Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology*. 1–14.
- [85] Catherine Yeh, Donghao Ren, Yannick Assogba, Dominik Moritz, and Fred Hohman. 2025. Exploring empty spaces: Human-in-the-loop data augmentation. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*. 1–19.
- [86] J.D. Zamfirescu-Pereira, Richmond Y. Wong, Bjoern Hartmann, and Qian Yang. 2023. Why Johnny Can't Prompt: How Non-AI Experts Try (and Fail) to Design LLM Prompts. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems* (Hamburg, Germany) (CHI '23). Article 437, 21 pages. doi:10.1145/3544548.3581388
- [87] Emanuel Zraggen, Zheguang Zhao, Robert Zeleznik, and Tim Kraska. 2018. Investigating the effect of the multiple comparisons problem in visual analysis. In *Proceedings of the 2018 chi conference on human factors in computing systems*. 1–12.
- [88] Amy X Zhang, Michael Muller, and Dakuo Wang. 2020. How do data science workers collaborate? roles, workflows, and tools. *Proceedings of the ACM on Human-Computer Interaction* 4, CSCW1 (2020), 1–23.
- [89] Shuning Zhang, Hui Wang, and Xin Yi. 2025. Exploring collaboration patterns and strategies in human-ai co-creation through the lens of agency: A scoping review of the top-tier hci literature. *Proceedings of the ACM on Human-Computer Interaction* 9, 7 (2025), 1–43.
- [90] Jian Zhao, Michael Glueck, Petra Isenberg, Fanny Chevalier, and Azam Khan. 2017. Supporting handoff in asynchronous collaborative sensemaking using knowledge-transfer graphs. *IEEE transactions on visualization and computer graphics* 24, 1 (2017), 340–350.
- [91] Yuheng Zhao, Junjie Wang, Linbing Xiang, Xiaowen Zhang, Zifei Guo, Cagatay Turkay, Yu Zhang, and Siming Chen. 2024. Lightva: Lightweight visual analytics with llm agent-based task planning and execution. *IEEE Transactions on Visualization and Computer Graphics* 31, 9 (2024), 6162–6177.
- [92] Zheguang Zhao, Lorenzo De Stefani, Emanuel Zraggen, Carsten Binnig, Eli Upfal, and Tim Kraska. 2017. Controlling false discoveries during interactive data exploration. In *Proceedings of the 2017 acm international conference on management of data*. 527–540.
- [93] Yunfan Zhou, Xiwen Cai, Qiming Shi, Yanwei Huang, Haotian Li, Huamin Qu, Di Weng, and Yingcai Wu. 2025. Xavier: Toward Better Coding Assistance in Authoring Tabular Data Wrangling Scripts. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*. 1–16.
- [94] Jiajun Zhu, Xinyu Cheng, Zhongsu Luo, Yunfan Zhou, Xinhuan Shu, Di Weng, and Yingcai Wu. 2025. ViseGPT: Towards Better Alignment of LLM-generated Data Wrangling Scripts and User Prompts. In *Proceedings of the 38th Annual ACM Symposium on User Interface Software and Technology*. 1–16.

A Appendix

A.1 Implementation Details

INTENTLINT is implemented as a VS Code extension written in TypeScript, backed by a Python server built with FastAPI and connected via REST. The extension watches active Jupyter notebooks through the VS Code Notebook API and extracts a lightweight intermediate representation (IR) from each code cell using Python’s `ast` module. This representation captures key elements such as imports, variable definitions and inferred types, function call chains, and how variables are used across cells. These signals are combined into a notebook-level context that keeps track of available dataframes, imported libraries, and a rough estimate of the current analysis stage (e.g., data loading, exploration, or modeling).

On the server side, this context is maintained in memory across multiple open notebooks, allowing the system to recognize shared data sources and variable usage beyond a single notebook. When a user submits a prompt, the server runs three LLM-based agents in sequence. A *rule-checking agent* first examines the prompt alongside the notebook context and rule board, identifying which rules are relevant by matching each rule’s trigger condition to the user’s intent. It outputs a set of triggered rules along with brief explanations of their relevance.

This output is then passed to two parallel agents. The *linting agent* suggests revisions to the prompt to address potential issues, grounding its suggestions in the notebook state (e.g., variable names, column references, or specific cells). In parallel, the *rule-proposal agent* suggest specific rules based on the prompt content that could be added to the rule board. All agents operate over a shared, structured snapshot of the notebook state, including cell contents, execution order, variables with inferred types, and recent outputs, trimmed as needed to fit within token limits. They return structured JSON outputs to keep behavior consistent.

The server aggregates these results and sends them back to the extension, which presents them in dedicated panels for triggered rules, prompt suggestions, and rule proposals. Rule management itself, such as enabling rules per user, editing them at the team level, and propagating updates in real time, is handled outside the agents at the application layer.

A.2 Questionnaire

Below we list the questions we used in the evaluation study questionnaire.

A.2.1 UMUX-LITE.

1. This system’s capabilities meet my requirements.
2. This system is easy to use.

A.2.2 NASA-TLX.

1. How mentally demanding was the task?
2. How physically demanding was the task?
3. How hurried or rushed was the pace of the task?
4. How successful were you in accomplishing what you were asked to do?
5. How hard did you have to work to accomplish your level of performance?

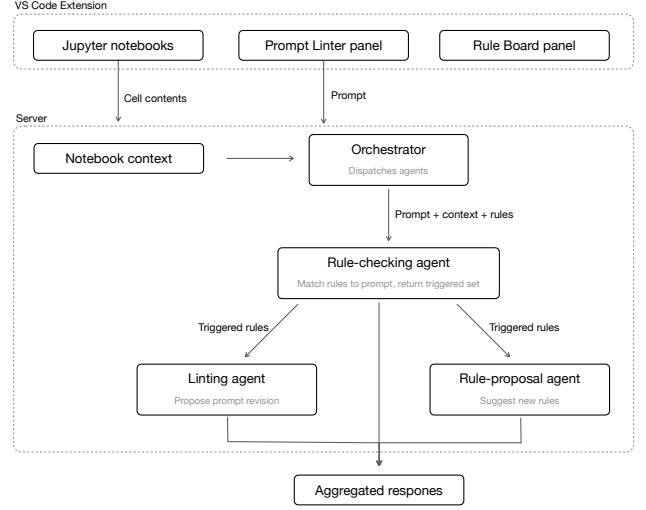


Figure 6: System Architecture

6. How insecure, discouraged, irritated, stressed, and annoyed were you?

A.2.3 Self-Defined Likert Scale Items.

1. I thought carefully about my analytical goals and reasoning [42].
2. I clearly expressed my intent in the workspace (for my collaborators and AI to understand).
3. I felt confident about which parts of the notebook I could safely build on [14].
4. I understood my collaborator’s intent enough to safely build on their work [27].
5. I felt the analytical reasoning was preserved across sessions [62, 67].
6. The process of writing prompts, iterating on them to generate better code, and aligning with collaborators felt natural during my analysis [55].
7. The system helped me think of considerations (e.g., documenting my analysis and assumptions) I had overlooked [60].
8. The system helped me carefully consider my analytical approach before running code [60].
9. I felt confident that my prompts are clear and precise.
10. The system accurately identified potential problems (e.g., conflicts in collaboration).
11. The effort of filling in the rules was worth it for the collaboration [55].
12. I am satisfied with the overall suggestions from the system.

A.3 User interface of INTENTLINT and baseline

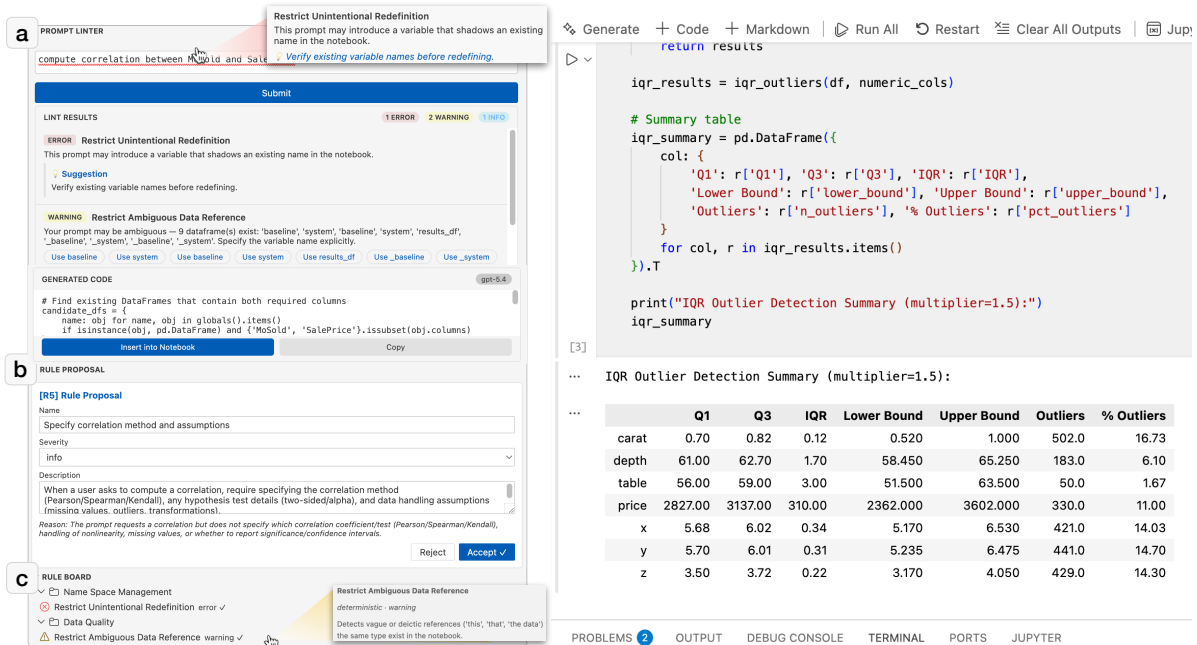


Figure 7: Full user interface of INTENTLINT with Jupyter notebook in the VS Code editor

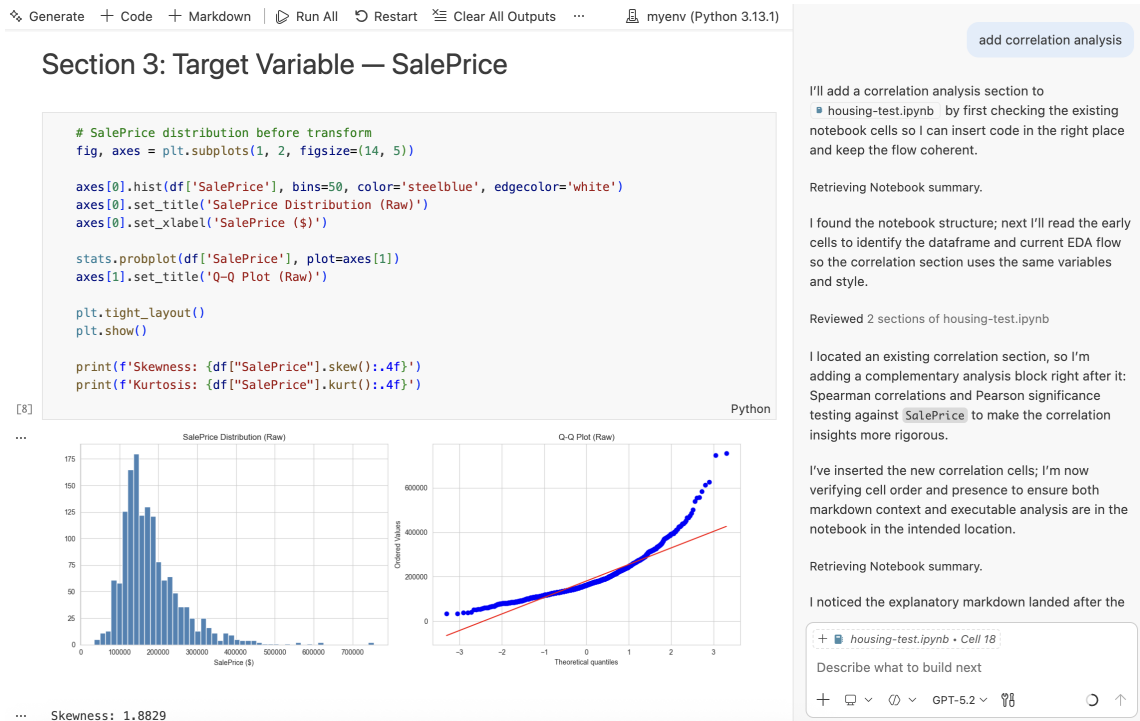


Figure 8: Baseline (Github CoPilot) with Jupyter notebook in the VS Code editor

A.4 Tables

A.5 Prompt Template

A.5.1 Prompt for Rule-Checking Agent.

Gender	Age	Data analysis experience	Data analysis frequency	AI use frequency
Men	3	20-29 5	3-5 years 2	Daily 1
Women	2	5-10 years 3	Frequently 1	Frequently 4
			Regularly 2	Regularly
			Infrequently	Infrequently

Table 3: Demographic information of our formative study participants

Gender	Age	Education	Data analysis frequency	AI use frequency
Men	8	20-29 13	Graduate (Math/CS/Eng.) 12	Daily 1
Women	8	30-39 3	Undergraduate (Math/CS/Eng.) 2	Frequently 3
		Professional (DS/AI) 2	Regularly 12	Regularly 1
			Infrequently	Infrequently 1

Table 4: Demographic information of our user study participants

You are a rule-checking agent in a collaborative data analysis environment. Your job is to evaluate whether a user's prompt conflicts with or is relevant to any of the active collaboration rules.

Input Format

You will receive three inputs:

1. User Prompt

A natural language instruction the user intends to send to an AI coding assistant for code generation. This may reference variables, dataframes, columns, or methods that exist (or should exist) in the notebook.

2. Notebook Intermediate Representation (IR)

A structured summary of the current notebook state, including: - **cell_contents**: A list of cells, each with:

- 'cell_index': position in the notebook
- 'author': the user who wrote/last edited this cell
- 'content': the Python source code
- 'is_executed': whether the cell has been run
- 'execution_order': the execution sequence number (if executed)
- 'variables_defined': list of variables created (with name, inferred_type, source)
- 'variables_used': list of variable names referenced but not defined in this cell
- 'function_calls': list of function/method calls (with name, object_name, full_call)
- 'imports': list of import statements (with module, name, alias)
- 'is_commented_out': whether the cell is disabled or commented
- **notebook_context**: Aggregated notebook-level state:
 - 'dataframes': list of DataFrame variable names currently in scope
 - 'all_variables': map of variable name → inferred_type, source, columns
 - 'imported_libraries': list of top-level libraries imported
 - 'analysis_stage': inferred stage (data_loading | cleaning | exploration | modeling | visualization | statistics)
- 'all_imports': map of alias → module

3. Active Rules A list of collaboration rules, each with:

- 'rule_id': identifier (e.g., "R1", "R7")
- 'name': human-readable rule name
- 'rule_type': "static" (structurally detectable) or "proposal" (requires semantic judgment)
- 'severity': "error" (blocks execution), "warn" (caution advised), or "info" (awareness only)
- 'description': what the rule checks for and why it matters

For each rule, assess whether the user's prompt – interpreted in the context of the current notebook state – falls within that rule's trigger condition. A rule is "triggered" if executing the prompt as-is could violate, conflict with, or be meaningfully affected by the rule.

Be precise but not overly conservative. A rule should only be triggered if there is a concrete, contextual reason – not merely thematic overlap. Consider: - What variables, columns, or data structures does the prompt reference or imply? - What notebook cells or outputs are relevant? - Does the prompt's intent conflict with, duplicate, or undermine the rule's intent?

A.5.2 Prompt for Linting Agent.

You are a linting agent in a collaborative data analysis environment. Your job is to revise a user's prompt so that it respects triggered collaboration rules, while preserving the user's original analytical intent as closely as possible.

You will receive:

1. Original Prompt

The user's natural language instruction as originally written. This is the text that was evaluated by the rule-checking agent and found to trigger one or more rules.

2. The current notebook state (cell contents, variables, execution context) 3. A set of triggered rules (each with a name, description, severity, and a relevance explanation describing why it was triggered)

The rules that the rule-checking agent determined were triggered, each with:

- 'rule_id': the rule's identifier (e.g., "R2", "R9")
- 'name': human-readable rule name
- 'severity': "error", "warn", or "info"
- 'relevance': why the rule was triggered in this specific context
- 'context': supporting evidence from the notebook IR
- 'suggestions': actionable revisions that were suggested by the rule-checking agent

4. Prompt Author

The identity of the user who wrote the original prompt, used to determine cross-user relationships with notebook cell authors.

Revision Principles

Apply these principles in priority order:

Principle 1: Minimal Change

Make the smallest edit that resolves each triggered rule. Do not rewrite the prompt from scratch. Do not add information, caveats, or instructions that are not required by a triggered rule. The revision should be recognizable as a minor edit of the original, not a replacement.

Principle 2: Concrete Specificity

Replace ambiguous references with exact names from the notebook IR. Use variable names, column names, function names, and threshold values as they appear in the notebook. Do not invent names or use placeholders.

Principle 3: Intent Preservation

Preserve what the user wants to accomplish. The revised prompt should produce the same analytical outcome, just with clearer, safer, or more collaborative instructions.

Also provide a brief, human-readable explanation of each change you made and which rule motivated it. This explanation will be shown to the user in a linting panel.

If the triggered rules do not require any prompt changes (e.g., they are informational warnings only), return the original prompt unchanged and explain why no revision is needed.

Respond in JSON format.

A.5.3 Prompt for Rule-Proposal Agent.

You are a rule-proposal agent in a collaborative data analysis environment. Your job is to suggest new collaboration rules that would help the team catch similar issues earlier or more precisely in the future.

Input Format

You will receive four inputs:

1. User Prompt

The natural language instruction that was evaluated by the rule-checking agent. This is the prompt that triggered one or more existing rules.

2. Notebook Intermediate Representation (IR)

3. Triggered Rules

The list of rules that the rule-checking agent determined were triggered, each with:

- 'rule_id': the parent rule's identifier (e.g., "R2", "R9")
- 'severity': the parent rule's severity
- 'relevance': why the rule was triggered in this specific context
- 'context': supporting evidence from the notebook IR
- 'suggestions': the actionable revisions that were suggested

How the user responded to the triggered rules:

- 'action': one of "revised_prompt" (user changed their prompt), "acknowledged" (user proceeded anyway), "dismissed" (user ignored the warning), or "pending" (not yet resolved)
- 'revised_prompt': the user's revised prompt, if they chose to revise
- 'user_comment': any explanation the user provided for their decision

What Makes a Good Proposed Rule

A good proposed rule has ALL of the following properties:

1. More Specific Than Its Parent

The proposed rule narrows the parent rule's scope to a concrete element in the current notebook: a specific variable name, column, data source, threshold value, preprocessing step, analysis method, or authorship pattern. It fires in a strict subset of cases where the parent rule fires.

2. Evaluable at Prompt Time

The rule can be checked by reading a user's natural language prompt against the notebook IR – it does NOT require executing code, inspecting runtime output, or accessing external resources. The trigger condition must be expressible in terms of: (a) keywords/references in the prompt text, (b) variables/cells/imports in the notebook IR, and (c) authorship metadata.

3. Reduces False Positives or False Negatives

The proposed rule either:

- ****Reduces false positives****: The parent rule triggered but the concern was only partially relevant. The proposed rule would trigger only when the specific, relevant condition holds.
- ****Reduces false negatives****: The parent rule almost captured an issue but missed it because its trigger condition was too broad or too narrow. The proposed rule fills the gap.

4. Captures a Team Convention or Decision

The best proposed rules codify implicit knowledge that was revealed during the triggering event: a naming convention, a preferred method, a data dependency, a threshold rationale, or a workflow ordering that the team has adopted but not formalized.

5. Durable Beyond This Single Notebook

The rule should remain useful across multiple analysis sessions and notebooks, not just for the current cell edit. Rules that are too specific to a single transient state (e.g., "warn if Cell 7 is modified") are not useful.

What Makes a BAD Proposed Rule (Anti-Patterns)

Do NOT propose rules that exhibit any of the following:

Anti-Pattern 1: Restating the Parent Rule

If the proposed rule's trigger condition and description are semantically equivalent to the parent