Agreement-Based Confidence for LLM-Based Functional Classification of User Stories

Carlos E. M. de Souza
Federal University of Campina Grande
Campina Grande, Paraíba, Brazil
carlos.souza@virtus.ufcg.edu.br

Emanuel Dantas
Federal Institute of Pernambuco
Jaboatão dos Guararapes
Pernambuco, Brazil
emanuel.dantas@virtus.ufcg.edu.br

Ademar França de Sousa Neto
Federal Rural University of the Semi-Arid Region
Mossoró, Rio Grande do Norte, Brazil
ademar.sousa@virtus.ufcg.edu.br

Mirko Perkusich
Federal University of Campina Grande
Campina Grande, Paraíba, Brazil
mirko@virtus.ufcg.edu.br

Danyllo W. Albuquerque
Federal University of Campina Grande
Campina Grande, Paraíba, Brazil
danyllo.albuquerque@virtus.ufcg.edu.br

Kyller Costa Gorgônio
Federal University of Campina Grande
Campina Grande, Paraíba, Brazil
kyller@virtus.ufcg.edu.br

Angelo Perkusich
Federal University of Campina Grande
Campina Grande, Paraíba, Brazil
perkusich@virtus.ufcg.edu.br

Abstract

Classifying user stories into functional categories can support backlog organization, reuse, and requirements analysis, but manual classification is costly and difficult to scale. Large Language Models (LLMs) can automate part of this task, although accuracy alone is insufficient for practical adoption: analysts also need to know when a classification should be accepted, reviewed, or corrected. This paper presents a controlled empirical study of confidence-aware user-story classification with GPT-5.2, DeepSeek, and Llama. The models classified 191 user stories from 10 projects using the same prompt, taxonomy, output schema, and execution constraints, with three repeated runs per model. The study evaluates exact classification performance, repeated-run stability, and inter-model agreement as a practical confidence signal. Results show that model accuracy and stability do not necessarily coincide. GPT-5.2 achieved the highest exact accuracy, while DeepSeek was the most stable across runs. Agreement among models provided useful evidence for selective automation: when all three models agreed, exact accuracy reached 84.4%; when only two agreed, it dropped to 47.8%; and when all models disagreed, it dropped to 22.2%. The results also show that disagreement is associated with underspecified stories, multi-intent stories, out-of-coverage cases, and problematic taxonomy labels. These findings suggest that multiple LLMs can be used as complementary decision-support agents rather than as isolated classifiers.

Keywords

Requirements Engineering, User Stories, Large Language Models, Functional Classification, Inter-Model Agreement, Intelligent Software Engineering

1 Introduction

User stories are widely used in agile software development to express requirements in a concise and stakeholder-oriented format [10, 13]. As projects grow, however, backlogs often become large, heterogeneous, and difficult to organize. This creates challenges for requirements analysis, reuse, traceability, and planning [3, 4]. One way to address this problem is to classify user stories according to their functional characteristics. Functional labels can help teams identify recurring patterns, compare similar requirements, and support reuse across projects [6].

Manual classification is nevertheless effort-intensive and hard to maintain in dynamic backlogs. LLMs offer a promising alternative because user stories are expressed in natural language and often depend on semantic cues that are difficult to capture with rules alone [9, 15]. Recent research has explored LLMs and related language models for requirements analysis and classification tasks [5, 11]. Yet practical adoption requires more than aggregate accuracy. In real requirements workflows, analysts need to decide whether a specific automated classification can be accepted, whether it should be reviewed, or whether the story or taxonomy requires correction.

This concern is aligned with broader work on human-AI decision support, which shows that global performance indicators can hide instance-level uncertainty and encourage overreliance on automated systems [1, 12]. In classification settings, confidence should therefore be understood not only as a model's reported score, but also as evidence that helps decide when automation is safe enough and when human review is necessary [7, 8].

This paper investigates confidence-aware functional classification of user stories using three LLMs: GPT-5.2, DeepSeek, and Llama. The models classify the same set of 191 user stories under the same prompt, WIS taxonomy, output schema, and execution constraints, with three repeated runs per model. The study addresses three

Table 1: WIS taxonomy used in the study.

Module	Operations
Registration	Insert data; Retrieve data; Update data; Remove data; Modify input behavior
Authentication	Login with username and password; Login with OAuth; Password recovery; First login; Validate user permissions; Update profile; Create account; Remove account
Management	View dashboard; Export report to PDF; Export report to XLS; Notify via app; Notify by email

research questions: (RQ1) how the evaluated LLMs differ in performance and stability; (RQ2) whether inter-model agreement can support acceptance or review decisions; and (RQ3) how story characteristics and taxonomy limitations contribute to disagreement and error. These questions follow a practical workflow. First, the individual behavior of each model must be understood. Second, the agreement among models must be evaluated as evidence for triage. Third, difficult cases must be inspected to understand whether the problem comes from the model, from the story text, or from the taxonomy itself.

The contribution is a compact empirical evaluation showing that confidence in LLM-based user-story classification should be treated as a process-level property. Instead of relying on a single prediction or on self-reported confidence, the proposed analysis combines accuracy, stability, and agreement to support selective automation. This perspective is particularly relevant for Intelligent Software Engineering because it treats LLMs as components of a decision-support process, not as autonomous replacements for requirements analysts.

2 Background and Related Work

This section summarizes the concepts needed to position the study. It first describes the classification task and the WIS taxonomy, and then relates this task to prior work on user-story quality, requirements classification, and trustworthy use of LLMs in Software Engineering.

User stories are lightweight requirements artifacts commonly written from the perspective of a user or stakeholder. Their flexibility supports communication, but it also creates quality problems such as ambiguity, incompleteness, and inconsistent granularity [2, 10]. Such problems can reduce the usefulness of user stories for systematic analysis and automation.

The classification task in this study is based on the WIS taxonomy proposed by Dilorenzo et al. [6]. The taxonomy represents functionality through *module–operation* pairs. A module denotes a broader functional area, while an operation specifies the action expressed by the story. In the configuration adopted here, the label space includes three modules: Registration, Authentication, and Management. Table 1 summarizes the operations used in the experiment.

The task is multi-label: a story may express more than one functional concern and may therefore receive more than one module–operation pair. Conversely, some functional stories may not be adequately represented by the current taxonomy. For this reason, the protocol includes an explicit out-of-coverage decision, represented as `n/a`. This prevents the model from forcing an unsupported taxonomy label when the story falls outside the available label space.

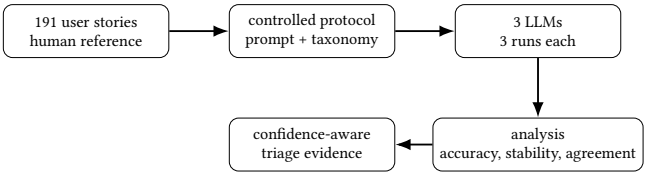

Figure 1: Overview of the study design.

Prior research on user-story quality has emphasized that structural clarity affects downstream analysis [2, 10]. Work on NLP and LLMs for Software Engineering shows that language models can support several requirements-related tasks, including classification, quality assessment, and requirements analysis [9, 11, 15]. However, most evaluations still emphasize overall performance. Less attention has been paid to operational confidence: how to identify classifications that are reliable enough to accept and cases that require human inspection. This paper addresses that gap through repeated executions and inter-model agreement.

The WIS setting is also stricter than many binary or coarse-grained requirements-classification tasks. A prediction is only considered exactly correct when the model selects the same normalized set of module–operation pairs as the human reference. A prediction with the correct module but the wrong operation is therefore treated as incorrect, because the operation is the reusable functional unit expected by the taxonomy. This conservative evaluation is appropriate for decision support: if an automated label is used to organize a backlog or retrieve reusable assets, a near miss may still lead the analyst to the wrong group of stories.

3 Method

The study was designed to evaluate LLM-based user-story classification as a controlled process rather than as a single model output. Figure 1 summarizes the procedure. The same dataset, prompt, taxonomy, and output schema were used for all models. Each model was executed three times, and the resulting outputs were analyzed from three complementary perspectives: model behavior and stability, agreement-based confidence, and sources of disagreement.

3.1 Dataset and Reference Classification

The dataset contains 191 user stories from 10 projects. The stories were selected from real product backlog items and screened to retain functional stories suitable for taxonomy-guided classification. Items focused mainly on constraints, quality attributes, or non-functional concerns were excluded. Each retained story includes its text, a consolidated human reference classification, and the outputs produced by each model in each execution.

The reference classification was produced by three specialists with experience in agile software development and requirements-related activities. The specialists analyzed coverage, module, and operation labels. Disagreements were reviewed and consolidated into a single ground truth. Before consolidation, agreement was high across the main labeling dimensions, with Cohen's Kappa values of 1.000 for coverage, 0.985 for module, and 0.946 for operation.

Table 2: Summary of the structured prompt constraints.

Constraint	Purpose
Closed taxonomy	Prevents unsupported modules and operations from being invented.
Multi-label output	Allows a story to express more than one functional concern.
Explicit n/a	Supports abstention when the story is outside taxonomy coverage.
Raw JSON schema	Makes outputs comparable and easier to normalize.
Review fields	Records reasons and possible issues for later human inspection.

Table 3: Metrics used in the analysis.

Metric	Definition
Exact accuracy	The predicted normalized set exactly matches the human reference.
Covered accuracy	Exact accuracy computed only over stories whose reference is not n/a.
Coverage accuracy	The model correctly decides whether the story is covered by the taxonomy.
Exact agreement	Two runs of the same model produce the same normalized output.
All-3 identical	The three runs of the same model produce identical normalized outputs.
Agreement level	The number of models that converge on the same representative prediction.

This supports the use of the consolidated labels as a reliable basis for model evaluation.

3.2 Controlled Classification Protocol

All models received the same structured prompt. The prompt defined the task as WIS-based functional classification, provided the taxonomy as a closed label space, allowed multiple module–operation pairs, and required a JSON output. When a story did not fit the taxonomy, the model was instructed to return a single n/a line, mark the case as requiring review, and explain the reason in an auxiliary field.

Although the output schema included fields such as self-reported confidence, rationale, evidence, and review indication, the primary analysis used only the normalized classification output. Each prediction was converted into an order-invariant set of module–operation pairs. This normalization allowed consistent comparison across models, runs, and the human reference. The decision not to rely on self-reported confidence is consistent with evidence that confidence scores produced by models are often poorly calibrated [8, 16].

The protocol also constrained the models in three ways. First, the label space was closed: models could only use the modules and operations defined in the taxonomy. Second, the task was explicitly multi-label, avoiding the loss of information when a story expressed more than one functional concern. Third, the n/a option acted as a controlled abstention mechanism. These constraints made the outputs easier to compare and reduced the risk of invented labels.

The evaluated models were GPT-5.2, DeepSeek, and Llama. All executions used the same prompt wording, taxonomy, JSON schema, and output requirements. The decoding temperature was fixed at 0.0 for all models. Each model classified all 191 stories three times, allowing the analysis to distinguish correctness from repeated-run stability.

Table 2 summarizes the main prompt constraints. These constraints are important because the study does not evaluate free-form generation. Instead, it evaluates whether LLMs can operate under an explicit requirements-engineering protocol, where the label space, abstention behavior, and output structure are defined in advance. This makes the setting closer to a tool-supported workflow than to an open-ended question-answering task.

3.3 Analysis Procedure

For RQ1, each model was compared with the consolidated human reference using exact label accuracy, exact label accuracy over covered stories only, coverage accuracy, and n/a prediction rate. Exact accuracy required the full normalized prediction set to match the reference. Coverage accuracy measured whether the model correctly distinguished covered from out-of-coverage stories. Pairwise differences in exact correctness were tested using McNemar's test.

Repeated-run stability was analyzed using exact run-to-run agreement, mean Jaccard similarity between normalized label sets, mean Cohen's Kappa, and the proportion of stories for which all three runs produced identical outputs. For agreement-based analysis, a representative prediction was selected for each model and story. When at least two runs produced the same normalized output, that majority output was used. When all three runs differed, the model-story pair was flagged as unstable.

For RQ2, stories were grouped by agreement among the representative predictions: full agreement, partial agreement, total disagreement, or presence of at least one unstable model. For each group, exact and coverage accuracy were computed. For RQ3, the analysis examined story-level factors, such as absence of an explicit *so that* clause and multi-intent stories, and taxonomy-level factors, such as out-of-coverage cases and labels with recurrent disagreement.

4 Results and Discussion

This section reports the empirical results according to the three research questions. The analyses use the 191-story dataset and the three repeated runs of each model. Unless otherwise stated, model-level comparisons are based on the representative prediction selected from repeated executions.

4.1 RQ1: Model Behavior and Stability

Table 4 summarizes model performance and stability. GPT-5.2 achieved the highest exact accuracy on the full dataset (67.5%), the highest exact accuracy on covered stories (69.9%), and the highest coverage accuracy (86.9%). DeepSeek ranked second in exact accuracy (57.6%) and showed the strongest tendency to abstain, predicting n/a for 35.1% of the stories. Llama showed the weakest exact accuracy (29.3%) and did not predict n/a in the representative run.

The pairwise differences in exact correctness were statistically significant. GPT-5.2 outperformed DeepSeek ($\chi^2 = 8.308$, $p = 0.00395$) and Llama ($\chi^2 = 64.000$, $p < 0.000001$). DeepSeek also outperformed Llama ($\chi^2 = 31.211$, $p = 0.000000023$). These results indicate that the observed differences are not only descriptive.

Stability produced a different picture. DeepSeek was the most stable model, with 97.9% mean exact run-to-run agreement and

Table 4: Performance and repeated-run stability of the evaluated models.

Model	Exact acc.	Covered acc.	Cov. acc.	n/a rate	Exact agr.	All-3 ident.
GPT-5.2	67.5	69.9	86.9	20.9	83.8	78.0
DeepSeek	57.6	51.5	78.0	35.1	97.9	96.9
Llama	29.3	41.2	71.2	0.0	95.5	93.2

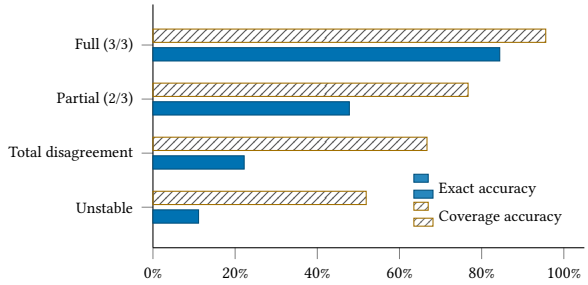

Figure 2: Accuracy by inter-model agreement level.

96.9% of stories receiving identical outputs in all three runs. Llama was also highly stable, with 95.5% mean exact agreement and 93.2% all-three identical outputs. GPT-5.2 was less stable, with 83.8% mean exact agreement and 78.0% all-three identical outputs. Thus, the most accurate model was not the most stable, and the most stable models were not necessarily the most correct. This distinction is important because practical confidence depends on both correctness and reproducibility.

The answer to RQ1 is therefore twofold. The models differ significantly in exact classification performance, but their reliability profiles are not captured by accuracy alone. GPT-5.2 is the strongest classifier in this setting, while DeepSeek provides more reproducible behavior. Llama's high stability, combined with low exact accuracy, shows why stability cannot be used as a substitute for correctness.

4.2 RQ2: Agreement as a Confidence Signal

Figure 2 shows a clear relationship between inter-model agreement and correctness. Full agreement among the three models produced the strongest results, with 84.4% exact accuracy and 95.6% coverage accuracy. When only two models agreed, exact accuracy dropped to 47.8%. When all models disagreed, exact accuracy dropped further to 22.2%. Cases with at least one unstable model showed the weakest behavior, with 11.1% exact accuracy.

These results support the use of agreement as a practical confidence signal. Full agreement identifies classifications that are more suitable for streamlined acceptance, while partial agreement, total disagreement, and instability indicate cases that should receive human attention. However, agreement should not be interpreted as proof of correctness. Among the 45 full-agreement stories, seven were still misclassified. Two of these false-consensus cases involved stories outside the taxonomy's effective coverage, suggesting that all models may converge on the same incorrect label when the available taxonomy is inadequate.

Table 5: Operational interpretation of agreement levels.

Agreement	Observed pattern	Use in workflow
Full agreement	Highest exact and coverage accuracy	Candidate for low-priority review
Partial agreement	Intermediate accuracy; often one correct model	Human arbitration with model alternatives
Total disagreement	Low exact accuracy	Careful review and possible clarification
Unstable output	Weakest overall behavior	Escalate before using the label

Table 6: Story-level factors associated with error.

Group	N	Exact acc.	Disagr./instab.
No *so that* clause	75	47.1%	17.3%
With *so that* clause	116	54.3%	17.2%
Multi-intent	17	43.1%	17.6%
Atomic	174	52.3%	17.2%

Disagreement should also not be treated simply as failure. In partial-agreement cases, at least one model was correct in 76.1% of the stories. In total-disagreement cases, at least one model was correct in 66.7% of the stories. This means disagreement can still be useful for decision support: it exposes alternative classifications that analysts can inspect, while warning against automatic acceptance. Table 5 summarizes this interpretation.

4.3 RQ3: Story and Taxonomy Factors

RQ3 examined whether errors and disagreement were associated with properties of the user stories and with limitations of the taxonomy. Stories without an explicit *so that* clause had lower exact accuracy than stories containing that clause (47.1% versus 54.3%), although their disagreement or instability rate was almost the same (17.3% versus 17.2%). This suggests that underspecification may reduce correctness even when models do not visibly disagree.

Multi-intent stories showed a similar pattern. Their exact accuracy was lower than atomic stories (43.1% versus 52.3%), and their full-agreement rate was lower (11.8% versus 24.7%). These results indicate that story formulation affects classification quality, but not always in a way that can be detected only through disagreement. Table 6 summarizes the story-level factors.

Taxonomy limitations produced stronger effects. The human reference contained 55 stories labeled `n/a`, indicating that they were outside the effective coverage of the WIS taxonomy. These cases had lower exact accuracy (44.9%) and a very low full-agreement rate (3.6%). The results suggest that part of the uncertainty comes from label-space limitations rather than model behavior alone.

Figure 3 shows the most problematic WIS labels. *Registration/Modify input behavior* was the clearest pressure point, with 19 instances, only 3.5% exact accuracy, and 47.4% disagreement or instability. Other difficult labels included *Authentication/Update profile*, *Management/Notify via app*, *Authentication/Password recovery*, and *Registration/Insert data*. These labels suggest where the taxonomy may need clearer boundaries, examples, or refinement.

Overall, RQ3 shows that classification uncertainty has multiple sources. Some errors are linked to story formulation, especially

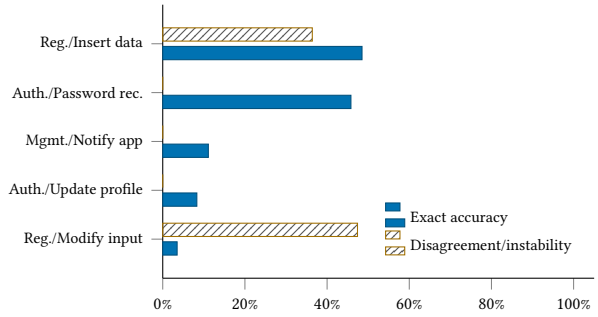

Figure 3: Problematic labels ordered by exact accuracy.

underspecification and multiple intentions. Others are linked to taxonomy coverage and label granularity. Therefore, disagreement should be interpreted not only as a model-level problem, but also as a diagnostic signal about the quality of the stories and the adequacy of the label space.

This finding is important for practice because it changes how disagreement should be handled. If disagreement is treated only as model failure, the natural response is to replace the model or tune the prompt. The results suggest a broader response: analysts should also inspect whether the story should be rewritten, whether the story bundles multiple requirements, and whether the taxonomy needs a new or clearer operation. In this sense, confidence-aware classification can support both automation and requirements improvement.

A simple use case illustrates the point. In a large backlog, full-agreement predictions could be used to pre-fill functional labels, allowing analysts to confirm them quickly. Partial-agreement predictions could be displayed with the competing labels proposed by the models, helping the analyst choose among alternatives. Total-disagreement and unstable cases could be sent to a review queue, because they are more likely to involve ambiguous stories, insufficient taxonomy coverage, or inconsistent model behavior. This workflow does not require blind trust in LLMs. It uses their convergence and divergence as signals for allocating human attention.

5 Implications and Threats to Validity

The results have methodological and practical implications. Methodologically, they show that evaluating LLM-based classifiers only through aggregate accuracy is insufficient. Exact accuracy, repeated-run stability, and inter-model agreement describe different aspects of model behavior. A model can be stable and still wrong, or more accurate but less reproducible. Confidence is therefore better understood as a property of the classification process rather than as a single score.

For practitioners, the results support selective automation. Full-agreement cases can be treated as stronger candidates for acceptance with lower review priority, while disagreement and instability should trigger human review. This does not remove the analyst from the process. Instead, it helps prioritize manual effort by identifying which classifications are more likely to be reliable and which require inspection. A possible workflow is to accept high-agreement predictions after lightweight inspection, send partial-agreement cases to analyst arbitration, and reserve detailed discussion for total-disagreement or unstable cases. In this sense, the committee of models functions as an attention-allocation mechanism.

For taxonomy maintainers, recurrent disagreement can reveal where the label space needs revision. The low accuracy and high disagreement around *Registration/Modify input behavior*, for example, suggest that this operation may be too broad or semantically unclear. Out-of-coverage cases also indicate that the taxonomy may not represent all functional patterns found in real backlogs. Agreement-based analysis can therefore support not only classification, but also taxonomy maintenance.

Threats to validity should be considered when interpreting the findings [14]. Conclusion validity may be affected by the size of some subgroups, especially in the story-factor and problematic-label analyses. Internal validity is mitigated by the use of the same prompt, taxonomy, output schema, and execution settings for all models, but model-specific API behavior may still introduce variation. Construct validity is affected by the use of exact-match accuracy, which is conservative and may underestimate partially correct predictions. External validity is limited to the 191 user stories, the WIS taxonomy, the selected models, and the adopted prompt protocol. Replications with other datasets, taxonomies, and model committees are needed.

6 Conclusion

This paper presented a controlled study of confidence-aware LLM-based functional classification of user stories. The study evaluated GPT-5.2, DeepSeek, and Llama over 191 user stories, using the same WIS taxonomy, prompt, output schema, and execution constraints, with three repeated runs per model.

The results show that model performance and stability do not necessarily coincide. GPT-5.2 achieved the strongest exact classification performance, while DeepSeek was the most stable model across repeated runs. Inter-model agreement provided a useful confidence signal: full agreement was associated with much higher exact accuracy than partial agreement, total disagreement, or unstable outputs. At the same time, false consensus showed that agreement is not a guarantee of correctness.

The study also found that uncertainty is not only a model-level issue. Underspecified stories, multi-intent stories, out-of-coverage cases, and problematic taxonomy labels all contributed to classification errors. These findings support a shift from one-shot model evaluation to confidence-aware workflows in which multiple LLMs act as complementary decision-support agents. In such workflows, agreement helps identify classifications that are more suitable for acceptance, while disagreement helps prioritize human review and taxonomy refinement.

Future work should evaluate other model committees, including models with different architectures and deployment settings. It should also test additional requirements artifacts, alternative taxonomies, and industrial datasets with different levels of story quality. A more detailed qualitative analysis of false consensus cases is also necessary. These cases are especially important because they reveal situations in which agreement may create an illusion of reliability even though all models converge on the wrong label.

Artifact Availability

All artifacts related to this study—including prompts, scripts, detailed methodology, results, and evaluation resources—are available to facilitate transparency, replication, and further research [1].

Generative AI Use

The authors used ChatGPT to support language editing, clarity, readability, and minor textual refinement during manuscript preparation. No AI tool was used to define the study design, construct the dataset, produce the reference classifications, execute the experiment, analyze the data, interpret the results, or draw the conclusions. All AI-assisted text was reviewed, revised, and validated by the authors, who take full responsibility for the content of this manuscript.

Acknowledgments

This work has been partially funded by the project "ISOP Engineering: Métodos e Ferramentas de Engenharia Inteligente para Sensoriamento Inteligente" supported by CENTRO DE COMPETÊNCIA EMBRAPII VIRTUS EM HARDWARE INTELIGENTE PARA INDÚSTRIA - VIRTUS-CC, with financial resources from the PPI HardwareBR of the MCTI grant number 055/2023, signed with EMBRAPII.

This study was financed in part by the Paraíba State Research Foundation (FAPESQ), grant number 997/2025.

References

[1] Saleema Amershi, Dan Weld, Mihaela Vorvoreanu, Adam Fourney, Besmira Nushi, Penny Collisson, Jina Suh, Shamsi Iqbal, Paul N. Bennett, Kori Inkpen, Jaime Teevan, Ruth Kikin-Gil, and Eric Horvitz. 2019. Guidelines for Human-AI Interaction. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–13. doi:10.1145/3290605.3300233

[2] Anis R. Amna and Geert Poels. 2022. Ambiguity in User Stories: A Systematic Literature Review. *Information and Software Technology* 145 (2022), 106824. doi:10.1016/j.infsof.2022.106824

[3] Anis Rahmawati Amna and Geert Poels. 2022. Systematic Literature Mapping of User Story Research. *IEEE Access* 10 (2022), 51723–51746. doi:10.1109/ACCESS.2022.3173745

[4] Edna Canedo, Angelica Calazans, Geovana Silva, Eloisa Masson, and Isabel Brito. 2024. On the Challenges to Documenting Requirements in Agile Software Development: A Practitioners' Perspective. In *Anais do XXVII Congresso Ibero-Americano em Engenharia de Software*. SBC, 286–300. doi:10.5753/cibse.2024.28454

[5] Porchourng Chuor, Ashwin Ittoo, and Samedi Heng. 2024. User Story Classification with Machine Learning and LLMs. In *Knowledge Science, Engineering and Management: 17th International Conference, KSEM 2024, Birmingham, UK, August 16–18, 2024, Proceedings, Part I* (Birmingham, United Kingdom). Springer-Verlag, 161–175. doi:10.1007/978-981-97-5492-2_13

[6] Ednaldo Dilorenzo, Emanuel Dantas, Mirko Perkusich, Felipe Ramos, Alexandre Costa, Danyllo Albuquerque, Hyggo Almeida, and Angelo Perkusich. 2020. Enabling the Reuse of Software Development Assets Through a Taxonomy for User Stories. *IEEE Access* 8 (2020), 107285–107300. doi:10.1109/ACCESS.2020.2996951

[7] Yonatan Geifman and Ran El-Yaniv. 2017. Selective classification for deep neural networks. In *Proceedings of the 31st International Conference on Neural Information Processing Systems* (Long Beach, California, USA) *(NIPS'17)*. Curran Associates Inc., Red Hook, NY, USA, 4885–4894.

[8] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. 2017. On Calibration of Modern Neural Networks. In *Proceedings of the 34th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 70)*. 1321–1330.

[9] Xinyi Hou, Yanjie Zhao, Yue Liu, Zhou Yang, Kailong Wang, Li Li, Xiapu Luo, David Lo, John Grundy, and Haoyu Wang. 2024. Large Language Models for Software Engineering: A Systematic Literature Review. arXiv:2308.10620 [cs.SE] https://arxiv.org/abs/2308.10620

[10] Garm Lucassen, Fabiano Dalpiaz, Jan Martijn E. M. van der Werf, and Sjaak Brinkkemper. 2016. Improving Agile Requirements: The Quality User Story Framework and Tool. *Requirements Engineering* 21, 3 (2016), 383–403. doi:10.1007/s00766-016-0250-x

[11] Johannes J. Norheim, Eric Rebentisch, Dekai Xiao, Lorenz Draeger, Alain Kerbrat, and Olivier L. de Weck. 2024. Challenges in applying large language models to requirements engineering tasks. *Design Science* 10 (2024), e16. doi:10.1017/dsj.2024.8

[12] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16)*. Association for Computing Machinery, New York, NY, USA, 1135–1144. doi:10.1145/2939672.2939778

[13] Tor Sporsem, Torgeir Dingsøyr, and Klaas-Jan Stol. 2026. User stories as boundary objects in agile requirements engineering: A theoretical literature review. *Journal of Systems and Software* 233 (2026), 112693. doi:10.1016/j.jss.2025.112693

[14] Claes Wohlin, Per Runeson, Martin Höst, Magnus C. Ohlsson, Björn Regnell, and Anders Wesslén. 2012. *Experimentation in Software Engineering*. Springer. doi:10.1007/978-3-642-29044-2

[15] Quanjun Zhang, Chunrong Fang, Yang Xie, Yaxin Zhang, Shengcheng Yu, Weisong Sun, Yun Yang, and Zhenyu Chen. 2026. A Survey on Large Language Models for Software Engineering. *Science China Information Sciences* 69, 4 (March 2026), 141102. doi:10.1007/s11432-025-4670-0

[16] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena. In *Advances in Neural Information Processing Systems*, Vol. 36. 46595–46623.

[1] Available at: https://doi.org/10.6084/m9.figshare.32113492