Coding Guidelines for XARM Dataset Annotation

This document provides the coding guidelines for annotating end-user feedback into four distinct categories: **Feature, Issue, User Experience, and Other Information**. Each category is explained with definitions and examples taken from the dataset to ensure consistent and accurate annotation.

1. Feature

Definition:
The "Feature" category encompasses user feedback that specifically requests the addition of new functionalities or the improvement of existing features within the software. These reviews often point out capabilities that the user believes the software should have but currently lacks. The feedback may include suggestions for entirely new features, modifications to enhance current functionality, or ideas for improving usability based on the user's specific needs or expectations. Users expressing feature-related feedback tend to focus on gaps between what they need the app to do and what it presently offers, often describing how certain functions or integrations could improve their overall experience. Reviews in this category might also highlight frustrations with missing features or compare the software to other apps that offer the desired functionality. These requests are important for developers to understand, as they provide insights into user priorities and the areas where the software could evolve to better meet the expectations and demands of its user base. By addressing these feature requests, developers can improve the application's utility, expand its appeal, and increase user satisfaction.

Examples from the dataset:

1. "I like it, but it doesn't do what I need. I need better integration with my other devices."
2. "I wanted some straight-up recipes. This app is more focused on fancy features that I don’t need."
3. "Deleted the app in less than 5 minutes. It lacks the basic functions I was looking for."
4. "Not what I wanted, so I deleted it from my Kindle. It should include a simpler recipe list."
5. "I was hoping for a feature to export my shopping list to other apps, but that option is missing."

2. Issue

Definition:
The "Issue" category includes feedback where users describe problems, malfunctions, or errors they encounter while using the software. These reviews typically highlight things that are broken, dysfunctional, or not performing as expected. Common issues include crashes, bugs, poor performance, glitches, or unexpected behavior that hinders the app’s normal functioning. Users providing feedback in this category often share their frustration, as the problems they face prevent them from using the app effectively. This type of feedback is vital for developers, as it directly points to areas of the software that need attention to improve reliability, stability, and user experience. Issues may vary in severity, from minor inconveniences to critical failures, such as

crashes or loss of data. These reviews provide important insights into the app's stability across different devices, operating systems, or updates. Addressing these issues helps ensure that the application performs as users expect, which in turn helps improve the software's ratings and retain a satisfied user base. Fixing reported problems also reflects the developer's responsiveness and commitment to quality, which can enhance user trust and long-term engagement with the app.

Examples from the dataset:

1. "Initially connecting an iPhone using the ScreenCast app showed strange error messages."
2. "It starts you sort of at the middle.. I learn better from the beginning."
3. "Really easy to set up however I could only get it to work once."
4. "Once I updated it will no longer let me video chat."
5. "Terrible app full of ads."

3. User Experience

Definition:
The "User Experience" category refers to feedback that focuses on how users interact with the software, covering aspects such as usability, design, navigation, and overall satisfaction with the application. This type of feedback typically includes comments on the app's ease of use, responsiveness, layout, and how intuitively the user can navigate through its features. Users often express their satisfaction or frustration with specific design choices, such as how the interface is organized, whether actions are easy to perform, or if the app's performance (e.g., speed, loading times) meets expectations. Positive user experience feedback highlights a seamless, enjoyable, and efficient interaction with the app, while negative feedback may point to confusing design elements, sluggish performance, or challenges in completing tasks. This feedback helps developers understand how their app is perceived in terms of usability and accessibility, providing actionable insights into what makes the app enjoyable to use or where improvements are necessary to reduce friction in the user journey. Enhancing user experience is crucial for retaining users and ensuring they continue to engage with the application in the long term.

Examples from the dataset:

1. "While it's easy to use and user-friendly, it takes too long to load the main menu."
2. "The app's layout is confusing, and it took me a while to find basic features."
3. "I love the idea, but the app feels sluggish and unresponsive sometimes."
4. "Navigating between sections is tricky; the buttons are not where I expect them to be."
5. "The design is great, but it crashes often when switching screens, which makes it frustrating to use."

4. Other Information

Definition:
The "Other Information" category covers user feedback that does not fit into specific functional categories like features, issues, or user experience. This type of feedback often pertains to topics outside the app's direct operation or performance. Reviews categorized as "Other Information" may include comments about legal matters, such as privacy policies or terms of service, pricing and subscription models, customer service experiences, or general app-related suggestions that do not touch on the app's functionality or user interface. Additionally, users may compare the app to competitors or share their thoughts on the app's monetization strategies, such as the presence of advertisements or in-app purchases. These reviews might also reflect on aspects like the company's communication, updates, or even ethical concerns. Although these reviews do not directly address the functionality or technical issues of the app, they provide valuable insights into users' overall impressions and expectations. Developers and stakeholders can use this feedback to better understand how external factors, beyond just the app's features, affect user satisfaction and loyalty.

Examples from the dataset:

1. "I noticed that the app has a privacy policy update, but it's hard to find the details."
2. "I don't mind the ads, but there should be an option to remove them with a one-time purchase."
3. "The customer support was slow to respond, and they didn't resolve my issue."
4. "I think the app's subscription model is too expensive compared to similar services."
5. "The latest update notes were vague and didn't explain what was actually changed."

Here's a set of clear suggestions for coders on how to annotate comments in the dataset, ensuring consistency and accuracy during the annotation process:

Suggestions for Coders

1. **Understand the Categories Clearly**: Before beginning the annotation, familiarize yourself with the definitions of the four categories: Feature, Issue, User Experience, and Other Information. Ensure you understand the nuances of each category to classify the feedback correctly. Refer to the definitions and examples provided for guidance.
2. **Focus on the Main Theme**: Read each review carefully and try to identify the dominant theme of the feedback. If the review mentions multiple issues or comments, prioritize the most prominent or repeated point. For example, if a user mentions both a missing feature and poor usability, choose the category that appears to carry more weight in their complaint.
3. **Use Provided Examples as Reference**: For each category, review the examples provided and compare them to the review you're annotating. If the comment closely resembles one of the examples, annotate it accordingly. For example, if a user mention adding a new feature, annotate it under "Feature." If they discuss customer service or pricing, annotate it under "Other Information."
4. **Dealing with Overlap**: Some reviews may span multiple categories (e.g., discussing both a feature request and a usability issue). In these cases, use your judgment to determine

which category is more emphasized by the user. When in doubt, annotate based on the most actionable or prominent aspect of the feedback.

5. **Be Consistent**: Once you've decided on a category, maintain consistency throughout the annotation process. If a similar type of review appears later, annotate it in the same way. Consistency is key to maintaining the dataset's integrity.
6. **Review Annotations Periodically**: After annotating a batch of comments, review your previous annotations to ensure they align with your current understanding of the categories. Adjust if necessary to maintain uniformity in categorization.