**Ambiguous Requirements:**

Many software requirements suffer from ambiguity. Ambiguity means that a single reader can interpret the requirement in more than one way or that multiple readers come to different interpretations. In either case, [ambiguous requirements lead to confusion](http://searchsoftwarequality.techtarget.com/feature/Writing-requirements-Common-sense-measures-for-success), wasted effort and rework.

Following are the examples of Ambiguous Requirements :

## **I) Unclear Context**

Don’t depend on external context. The people reading the requirements may not have been in the same meeting that you were and the developers don’t generally have the kinds of access to the clients that the writers of requirements do. A missing context can create a vital flaw in the requirements.

Is that large screen specification for a region in China that the majority of screens are still only up to 720px? That would make all the difference for this requirement.

#### **ii ) Negative Requirements**

Negative requirements say what the system should not do, rather than saying what it should do. For example, if the requirement stated “The system should not allow external users to access…” does that mean that consultants working on-site get access or not? The requirement is better phrased as “The system will only allow employees to access…”

Note that : The word **only**is what creates the limitation. Avoid double negatives, which are even more ambiguous.

1. **PRONOUNS**  
    Pronouns offer another opportunity for confusion if the antecedent is for each pronoun is not absolutely clear. If you say "this" or "that," there should be no confusion in the reader's mind as to what you are referring to. Pronouns can be ambiguous when their antecedents are unclear.

Here's an **example** of a requirement with a pronoun whose antecedent is unclear:

“When in a date field, the user should be able to open a pop-up calendar. This allows the user to…” What is the this that allows the user to do something? It could be the date field or the calendar. Unclear antecedents introduce ambiguities.

That requirement could be **rewritten** like this: “When in a date field, the user should be able to open a pop-up calendar which allows the user to …” I have found that tools like [grammarly](https://www.grammarly.com/" \t "_blank) can be very good at catching unclear antecedent errors.

When writing requirements, write “the user” rather than “you” because there is less opportunity for confusion. For example, “The user should be able to…” rather than “You should be able to…”.