**Functionality** – The system must be able to choose the offer that will incur the least expenses. The system must also accept excel files as input. Finally, the system must export the winning offer(s) as a pdf file.

Security features? Data anonymization?

**Usability** – The user interface will not be a priority of the system, so it may be a CLI. Therefore, it may be difficult to navigate and use. If we have time to implement a GUI, then it should be simple and easy to navigate without many bright colors or unnecessary buttons.

**Reliability** – The calculations for finding the best offer should be correct 100% of the time. If there is some failure in the system, it will need to be restarted.

**Performance** – The program should be very fast in every aspect, except for the user interface which may take some time to navigate if it is only a CLI, but if it is a GUI it should fast and easy to use.

**Supportability** – The system should be created using TDD. The system will need to be updated if any criteria for choosing the best offer change or if it needs some extra functionality beyond what is currently required.

**Constraints** – It must accept excel files as input and output pdf files.

“The + reminds us of a few additional needs that a customer could have:”

 **Design constraints** - Do things like I/O devices or DBMS constrain how the software must be built?

 **Implementation requirements** - Do the programmers need to adhere to standards? Is the use of TDD required? Is statistically sound testing in the context of Cleanroom required?

 **Interface requirements** - What downstream feeds must be created? What other systems must this one interface with? How frequent are feeds produced?

 **Physical requirements** - What hardware must the system be deployable on? Must we be able to deploy to a machine no larger than 12" square, to be stationed in the Iraqi desert?