1. Planning: User Acceptance Tests must encompass the following areas in order to be successful.
   1. Time Management: 1) Clarify basic questions; Collect information on UAT schedule, QA agents and testers. 2) Central repository for information; Draft a concept of what the testing group should look like. Set clear expectations.
   2. Team Requirements: Plan who is involved in UAT and identify tasks and responsibilities. Clarity and constant communication are paramount.
   3. Communication & Issue Strategy: Develop workflow that deals with bugs, issues and other problems.
   4. UAT Checklist: Provide testers with a comprehensive User Acceptance Testing checklist. This will help them stay focused and keep testers on task.
2. Execution: UAT users will test and evaluate our application (SMaRT). Testing will include python code and application flow as outlined in the workflow.
3. Documentation: Execution and Documentation should happen simultaneously.
4. Evaluation: UAT users need to evaluate if the defined criteria have been tested and met. Collect, aggregate and analyze UAT data.
   1. Test scenarios should cover a specific process of the workflow application
   2. Test scenarios should have clear test steps
   3. Focus on quality not quantity.
   4. Test you test assignments yourself.
   5. Start tests early and often.

Then determine:

* 1. Has any test case failed?
  2. What problems did the UAT user encounter?
  3. How could problems be resolved?
  4. Did every process in the workflow get tested?
  5. The goal of UAT testers is not to focus on finding defects but how the process flows and if a defect is found it should be reported in step 5

Quantitative and Qualitative data documented:

* How many testers completed the test cases?
* What was the overall rating of these test cases?
* What was the overall understanding of the process by each tester?
* What problems arose during the test cases?

1. Reporting & Lessons Learned: During this phase you need to gather insights and any lessons learned that will help you improve future test cases and UAT workflows as well as improve the development of the python code and its application (SMaRT). Clarify all findings and provide communication feedback.