**Possible Testing Approaches**

The various approaches that could be taken for testing would be:

1. The most obvious testing strategy that first comes to mind would be something akin to the agile process to test the requirements as stated in the Software requirements document. To accomplish this, we would have to incorporate testing as we move through the development stage. Such testing is also referred to as "proactive." This strategy allows for the creation of "black box" tests, which are created solely based on the requirements. Another variation of this approach could include a “grey box” test where limited implementation information is available.
2. Another approach that could be used is to conduct testing analysis after the system has been fully implemented. Such an approach is monikered “reactive” testing since we start testing at the end stages of the development. Since we are fully aware of all the implementation details while the tests are being developed, the approach uses a technique such as “white box” tests.
3. Combining the two methods mentioned above is another testing strategy that might be used. In essence, this method has the advantage of allowing time for test case design or refinement based on implementation details at the conclusion of the testing process. In this strategy, we primarily rely on "grey box" testing, where only partial details of the implementation are known. With this approach, testers are able to use both white-box testing techniques to ensure proper functionality early in the development process as well as black-box techniques to fine tune test cases closer to the release date. This combination of techniques gives the testers an advantage, as it allows them to start testing early while also giving them time to create and refine test cases closer to the release date. By using the combination of white-box and black-box testing techniques in this approach, testers can ensure that all facets of the system are properly tested before it is released.
4. Another testing strategy would be that of heuristic testing; Goldilocks testing is a straightforward, heuristic-based testing approach that encourages testers to find the "just right" value for whatever element they are testing. It uses well-known heuristic testing techniques used by previous software testers.
5. Iterative testing is another testing strategy that may be implemented. Testing would be carried out at the climax of each "iteration" if the development process were divided into smaller "iterations." This would seamlessly combine testing and development, making it possible to find errors before moving on to the next "iteration," and it would also rely on grey-box testing, where only a limited amount of detail about the implementation specifics is available. By combining white-box and black-box testing, we can ensure that the software developed meets the requirements in an efficient manner. This iterative testing approach would provide great benefits, such as a structured approach, better communication between the development and testing teams, and a faster response to changing requirements.
6. The last testing strategy that could be applied is the DevOps continuous testing strategy. In this method, all tests are run simultaneously using an automation tool and pipeline, and if any bugs or errors are found, immediate feedback is generated. This is the least practical option for the chosen software since the project doesn't scale up enough, nevertheless it is a strategy worth mentioning because it essentially has very few drawbacks.

**Chosen Testing approach:**

We must be mindful of aspects like risks, resource costs, and schedule-timeline conflicts when selecting a testing approach. The third testing strategy was chosen because of the following factors: This testing strategy is appropriate for the project because it is easy to set up, enables the quick and effective execution of a variety of tests, and provides insightful feedback that can be used to improve the product. This method also has the advantage of allowing time for test case design or refinement based on implementation details at the conclusion of the testing process. In this strategy, we primarily rely on "grey box" testing, where only partial details of the implementation are known. With this approach, testers can use both white-box testing techniques to ensure proper functionality early in the development process as well as black-box techniques to fine tune test cases closer to the release date. This combination of techniques gives the testers an advantage, as it allows them to start testing early while also giving them time to create and refine test cases closer to the release date. By using a combination of white-box and black-box (as well as grey-box) testing techniques in this approach, testers can ensure that all facets of the system are properly tested before it is released. This strategy of combining white-box, black-box, and grey-box testing techniques is effective in ensuring that the system is tested thoroughly before its release. With this approach, testers can use a variety of techniques, from white-box testing to ensure functionality early in the development process, black-box techniques for fine tuning closer to the release date, and grey-box testing for more in depth inspection. By combining these three techniques, testers can ensure that the system is fully functional before its release and increase their confidence in the final product.