Deliverable 4 Risk Report

Daniel Benesh - CMPT 371 - Team 3

1. Test Matrix: Probability: 5; Impact: 5

   In this deliverable we were able to have a peer review on the test matrix in hopes to improve clarity, documentation and readability. We found that when going through the test matrix from previous deliverables that updates were necessary. With greater understanding of the test matrix, it is easier to confirm which tests need to be redone, tests that are no longer valid due to new features being implemented and improves overall grasp on what future tests need to be done. We will be adding links to the pages where the test exists so that it is easier to confirm test properties. We will be archiving/removing tests that are no longer valid. With these measures in place we hope to mitigate any confusion on the test matrix, improving readability and saving time spent going through the test matrix.

2. Communication: Probability:6; Impact 8; Recurring

   Poor communication leads to tasks not being understood and slows down efforts to complete deliverable goals. Team leads have a responsibility to make sure their team members understand their tasks. With daily communication, team meetings and updates from team members, we have lessened the impact of this risk. Although we are making great improvements to our methodologies of communication, there is still room for improvements. Our QA and Dev leads are having meetings to make sure that each side is working together. This will ensure that issues on either side are talked about and hopefully prevent any confusion. With these measures in place we hope to better improve communication between teams.

3. Playwright Issues: Probability: 8, Impact: 7

   When using playwright to automate tests for UI and backend elements, new playwright releases interfered with being able to complete these tests. When the project dependencies expect a certain release of playwright browsers, playwright will not be able to run its tests successfully. It will refuse to run tests until the correct browsers are downloaded. This creates confusion and ends up burning time to complete playwright test tasks. This issue can be resolved, by making sure at the beginning of the deliverable, all branches are using the same version of playwright and branches are configured to that version, and the dependent browsers for that versions.

4. Time estimations: Probability: 5, Impact: 4; Recurring

Improving time estimations will improve the accuracy in what we are able to complete in a deliverable. This will ensure that the team stays on track and completes what we set out to complete in a deliverable. In planning poker, instead of estimating in days (4 hr/day), we will be moving to hours in planning poker. This will improve the accuracy in our estimations. Estimating in days is less accurate than hours. The impact of having a less accurate estimation, is that tasks are not defined appropriately. This leads to over/under-estimating time to complete a task which is against the point of estimating in the first place. The more precise the estimation, the more accurate real vs. actual can be measured. Changing from days to hourly estimations will help mitigate against inaccuracies.

5. Auto Closing CI Pipeline: Probability: 7, Impact: 4

   When making a PR, our CI pipeline is supposed to close the issue, when the branch of the tasks being worked on is merged into development. Auto closing of the issue on github saves time in having someone having to go through the entire PR again to close the issue. At the moment we are having difficulty in having this automatic behavior. Once resolving this issue in the CI pipeline, it will free up more time for group members. Since when making a PR, 2 team members need to go through it for the merge to be complete, it should be unnecessary to have a 3rd extra step to go through the PR.

6. Backend Function Stubs: Probability:6; Impact: 8

   Knowing exactly how a function works is essential for proper implementation of said function. If the developer is unaware of function pre/post conditions, and what the function does, it is more difficult to implement the function. This will improve readability and time use for the developer. We will mitigate this risk to lower its impact by having members go through each function and complete javadoc style stubs so that the developer understands the function better, making it easier to implement in our project.

7. QA Productivity during the start of Deliverable: Probability: 6; Impact: 5; Recurring

   Having idle team members, on either side of dev or QA, will result in lost time. QA members have been assigned many other tasks that do not depend on the development side of the project. These tasks include tutorials, documentation and other tasks that improve understanding of different project features. This helps mitigate the lack of understanding of new technologies and greatly improves understanding of different feature components. In turn, it helps improve productivity of the team.

8. Profiling the backend: Probability 8; Impact: 7

   With the current profiling suite we are using, it may be difficult to fully profile the backend functions. This will affect the time to complete the task of profiling the entire app. We intend to research how the profiler functions to improve understanding of the profiler. This will make it easier to implement. At present, the profiler is fairly straightforward for UI elements. In the future we will need to profile how long uploads/downloads take, while implementing features that let the user know the progress of those features. This will let the user know that the app is working on uploading/downloading and not frozen. To profile these functions, we will need to know how long the calls to the backend elements take. In knowing these times, it will let us know if there are problems in the calls themselves since they should respond in a reasonable timeframe. In writing tutorials or sharing knowledge from research, it will mitigate the time it takes to profile the backend.

9. Incorporation Backend Assertions: 7, Impact:7

   Assertions are an integral part of ensuring the state of the system that the developer expects is actually the case. Without assertions the developer may be unaware of what is causing the bug or defect in the system. Our team is working on writing backend assertions to ensure that everything is functioning as expected, enabling the developer to narrow down any defects they may encounter. This is necessary for our project especially since we did not write the backend. In writing these assertions we hope to improve development times as well as QA concerns when they arise.

10. Documentation: 5, Impact: 8

    .Documentation is an essential part of our project. With so many different tools being used, it is important that each member of the project understands how the tool works and how to install/update it. When differences in tool versions/updates happen, the behavior of these tools will impact progress. We are using tools such as playwright, eslint, prettier, pre-commit hooks and others. Without the understanding of how these tools work, errors that occur can be hard to diagnose. As written above about playwright versions, this impedes the progress made since the bug isn't the code itself but the tool being used. In documenting installing, updating and use of the tools, we hope to mitigate the time it takes to complete tasks.