| Deliverable 2 - Dev | | | | | | | Deliverable 2 - Test | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Risk** | **Score** | **Severity (1/5)** | **Likelihood (1/5)** | **Mitigate** | **Contingency** | | **Risk** | **Score** | **Severity (1/5)** | **Likelihood (1/5)** | **Mitigate** | **Contingency** |
| 1. Dropping class | 5 | 5 | 1 | - be respectful to group members<br>- encourage group members | - have someone shadow high priority roles | | 1. Dropping class | 5 | 5 | 1 | - be respectful to group members<br>- encourage group members | - have someone shadow high priority roles |
| 2. Changing requirements | 4 | 2 | 2 | - meet with stakeholder frequently | - chosen hosting platform: Firebase allows easy database integrations if requirements change | | 2. Code changes close to deadline | 12 | 4 | 3 | - put in strict deadlines | - shadow dev team to prepare for what tests may be needed |
| 3. Lag with datasets | 12 | 4 | 3 | - use small datasets<br>- test early with dataset to observe need of optimizations | - create a server using node. js or other server language to offload | | 3. Unoptimized code | 6 | 3 | 2 | - testing to find the unoptimized code | - prioritize refactoring code |
| 4. Inadequate documentation | 4 | 2 | 2 | - review code (methods and classes) before pushing to codebase<br>- follow strict documentation rules | - refactor for self explanatory code | | 4. Inadequate documentation | 4 | 2 | 2 | - review code (methods and classes) before pushing to codebase<br>- follow strict documentation rules | - refactor for self explanatory code |
| 5. Poor communication | 3 | 3 | 1 | - discord always available<br>- discussions frequently | - use discord for emergency meeting | | 5. Poor communication | 3 | 3 | 1 | - discord always available<br>- discussions frequently | - use discord for emergency meeting |
| 6. Learning new tech stack (gherkin, webxr, etc.) takes up time, higher chance of mistakes | 12 | 4 | 3 | - open communication between dev and test team to share what they know and help each other out<br>- share where to learn this new tech stack | - prepared to help each other<br>- walk group member through their problem<br>- take on another task | | 6. Learning new tech stack (gherkin, webxr, etc.) takes up time, higher chance of mistakes | 12 | 4 | 3 | - open communication between dev and test team to share what they know and help each other out<br>- share where to learn this new tech stack | - prepared to help each other<br>- walk group member through their problem<br>- take on another task |
| 7. Scope creep | 4 | 4 | 1 | - become familiar with requirements<br>- regular meetings with stakeholder | - communicate if you notice some scope creep and plan for redirection | | 7. Manual integration test plan | 12 | 4 | 3 | - research automatic testing<br>- test different automatic testing tools | - well document the manual testing<br>- clean test code<br>- create proper test cases with manual testing |
| **ID2 - NEW RISKS** | | | | | | | | | | | | |
| 8. Midterm exams | 2 | 2 | 1 | - know everyones schedule<br>- plan around midterms | - have someone ready to take over or help complete task | | 8. Midterm exams | 2 | 2 | 1 | - know everyones schedule<br>- plan around midterms | - have someone ready to take over or help complete task |
| 9. CSPIP interviews (time conflicts amongst team) | 20 | 4 | 5 | - know everyones schedule<br>- plan around midterms | - have someone ready to take over or help complete task | | 9. CSPIP interviews (time conflicts amongst team) | 20 | 4 | 5 | - know everyones schedule<br>- plan around midterms | - have someone ready to take over or help complete task |
| 10. Possible incompatability with modules/libraries | 12 | 3 | 4 | - research how to set up and use libraries/modules<br>- test that libraries/modules work together | - have someone who set up libraries/modules available to help or fix problems | | 10. Jest not performing as expected for unit tests | 9 | 3 | 3 | - research how to set up and use Jest | - have someone who set up Jest available to help or fix problems |
| 11. indexedDB may overflow browser memory | 12 | 4 | 3 | - research and test indexedDB if it can be used | - pivot to a server<br>- drop columns from data point instantiation before quering indexedDB | | 11. indexedDB may overflow browser memory | 12 | 4 | 3 | - research and test indexedDB if it can be used | - pivot to a server<br>- drop columns from data point instantiation before quering indexedDB |
| 12. Managing security/permissions for pipelines to avoid downtime | 10 | 5 | 2 | - staggered pipeline deployment<br>- regularly update and maintain pipeline<br>- multiple reviews before Git commit to branch | - update security/permissions as needed during downtime | | 12. Managing security/permissions for pipelines to avoid downtime | 10 | 5 | 2 | - staggered pipeline deployment<br>- regularly update and maintain pipeline<br>- multiple reviews before Git commit to branch | - update security/permissions as needed during downtime |
| 13. Limited early testing/debugging capabilities | 8 | 2 | 4 | - test as best as you can with unit tests<br>- robust logging to find errors<br>- use assertions | - prioritize debugging<br>- allow for roll backs | | 13. Limited early testing/debugging capabilities | 8 | 2 | 4 | - test as best as you can with unit tests<br>- robust logging to find errors<br>- use assertions | - prioritize debugging<br>- allow for roll backs |