Risk Assessment Report

Project: *Companion Animal Surveillance Data Collection App*

Document Control

- **Document Version**: 1.1
- **Last Updated**: Jan 28, 2024

1. Introduction

This Risk Assessment Report outlines the approach and processes for identifying, assessing, and mitigating risks associated with developing and implementing the Companion Animal Surveillance Data Collection App for the CASI project.

2. Risk Identification

1. *Technology Incompatibility Risk* **(Id: R1) → Updated in version 1.1**
    - **Risk**: Typescript Express proves to be harder to work with, and poses a challenge to the development and integration process of the application, potentially leading to delays in implementation and increased complexity in troubleshooting and debugging tasks.
    - **Handling**:
        - Mitigation Plan: Ensure preparedness by having a backup plan in place. Maintain a Java Spring backend as an alternative solution with dependency injection, testing, and logging configurations. This approach enables developers to seamlessly fall back to Java Spring in the event of complications with Typescript Express, minimizing downtime and facilitating a smooth integration process.
        - Prototyping:
            - Have two backend setups ready - one with Typescript Express, and the other with Java Spring, both having dependency injection, testing and logging configurations in place.
        - Contingency Plan:
            - If Typescript/Express proves to be too complicated to work with and takes longer for things to set up, then the Developer will prepare to switch to Java Spring. Since Java Spring will

already be set up with dependency injection and logging and testing configuration, we should be able to switch with no downtime.
- In that case, Developers will have to continue on the **java-backup** branch rather than the backend-setup branch (which is configured for Typescript/Express)

- Analyzing Contingency Plan:
  - Java Spring has many easy out-of-the-box ready-to-use implementations of concepts such as Controllers, Dependency Injections, and DB Driver integration.
  - Fast set-up and easy configurations compared to express
  - Strongly typed, and object-oriented which is what everyone is used to

2. *Team Communication Risks* **(Id: R2) → From ID0**
   - **Risk**: Poor communication among team members, leading to confusion and inefficiency, results in missed deadlines, duplicated efforts, and a lack of alignment towards the common goal.
   - **Handling**:
     - Mitigation Plan:
       - Have clear communication channels and have regular team meetings and stand-ups to discuss progress, address challenges, and realign priorities as needed.
       - Use GitHub issues for tracking tasks, assigning responsibilities, and documenting discussions transparently.
     - Contingency Plan:
       - Have a meeting with everyone to openly discuss and clear the confusion/challenges

3. *Dependency on a Single Individual Risk* **(Id: R3) → From ID0**
   - **Risk**: The project relies heavily on one individual, making it vulnerable to disruptions if that person is unavailable (due to midterms, finals or any other reason) or if he drops the class (lower bus number). This could result in missed deadlines and unexpected pressure on the rest of the team.
   - **Handling**:
     - Mitigation Plan
       - Knowledge sharing (sharing expertise with other team members)

            - This could be achieved by pair programming/peer reviews
        - Implement thorough documentation throughout the software development lifecycle.
        - Conduct periodic reviews of the GitHub commits to help identify and address dependencies on specific individuals before they become critical issues.
    - <u>Contingency Plan</u>:
        - Identify potential successors who can assume critical roles in the absence of an individual (if the Developer is unavailable/drops, have a Tester take his role and vice-versa)
        - If both the developer and tester are unavailable, the Dev Lead takes the task if it pertains to the development process.
        - If both the developer and tester are unavailable, the Test Lead takes the task if it pertains to the testing process.
        - If the Risk Officer is unavailable/drops out, the responsibility to scan for risks and mitigate them falls on one of the developers and/or testers.

4. *<u>Requirements Not Fully Understood By Everyone in the Team</u>* **(Id: R4) → From ID0**
    - **Risk**: Incomplete understanding of requirements can lead to misinterpretations, implementation errors, wrong implementations, and which all can lead to a delay in project delivery and deliverable goals.
    - **Handling**:
        - <u>Mitigation Plan</u>:
            - Clear documentation of requirements shared amongst every team member
            - Regular stand-up meetings to address any conflicting understandings
            - Have the stakeholder review the requirements document to ensure alignment with expectations, and agreement on project scope and deliverables.
            - Continuous Feedback Loop: team members regularly provide feedback on requirements understanding

        - Prototyping: make use of a prototype/mock design to create a visual representation of requirements for a better and correct understanding amongst everyone.
    - <u>Contingency Plan</u>:
        - Have everyone review the requirements document
        - Have an immediate team meeting clarifying requirements to everyone

5. *<u>Team Member Availability Risk</u>* **(Id: R5) → From ID0**
    - **Risk**: Team members, such as developers, testers, or other essential contributors, may become unavailable for various reasons, leading to delays or interruptions in project progress and delivery.
    - **Handling**:
        - <u>Mitigation Plan</u>:
            - Use Doodle to plan and manage the team's schedule and track everyone's availability
            - Establish clear communication channels for team members to update their availability.
            - Notify the project manager or risk officer about schedule conflicts or unavailability beforehand.
        - <u>Contingency Plan</u>:
            - Establish a rapid response plan for redistributing tasks in case of sudden unavailability
                - Substitute team members based on the criticality of the tasks they are working on (refer to R3 for role substitution guidelines when someone is unavailable).
6. *<u>Team Member Skill Gap Risk</u>* **(Id: R6) → Updated in version 1.1**
    - **Risk**: there is a disparity between the skills needed to fulfill project tasks and the skills available within the team (Most of the team is unfamiliar with Typescript and React-Native)
    - **Handling**:
        - <u>Mitigation Plan</u>:
            - Share TS/React-Native tutorial videos through Discord
            - Identify team members with familiarity in specific areas (some people are more familiar with the backend and some

            with the frontend) and assign tasks based on familiarity and expertise.
            - Explicitly tell everyone within the group to become familiar with TS and React-Native on their own time.
            - Have a flexible project timeline that accommodates potential delays due to skill development (at least have 1-2 days of buffer period)
        - <u>Contingency Plan</u>:
            - In case of a significant skill gap, be prepared to reallocate tasks among team members who have become familiar with the technology.

7. *<u>Technology Lacks Resources and Documentation</u>* **(Id: R7) → Resolved**
    - **Risk**: The technology decided by the team lacks resources and documentation
    - **Handling**:
        - <u>Mitigation Plan</u>:
            - Before finalizing the technology, conduct thorough research to ensure it has adequate resources and documentation available
        - <u>Contingency Plan</u>:
            - During critical resource shortages for a particular technology, have an alternative technology to fall onto
            - Take help from someone who is experienced with the technology

3. Updates Since ID0

<u>Risks that were mitigated but nonetheless carried forward from the previous deliverable:</u>

- R1: *<u>Technology Incompatibility Risk</u>*
  - Actions we took:
    - Followed the mitigation plan stated in Risk Identification for R1
    - Created prototypes to test Ts/Express and Java Spring backend
    - Have a Java Spring backend all set up if we need to fall back onto that
  - Reason for carrying it forward:
    - We're not familiar with the compatibility issues with TS/Express and other technologies yet. Should be able to resolve this risk by ID2.
- R2: *<u>Team Communication Risks</u>*
  - Actions we took:
    - Followed the mitigation plan stated in Risk Identification for R2
    - Risk is almost negligible now that we have a good communication protocol set up but since it still has a possibility to manifest, it will be carried forward to ID1, and we plan to mitigate that completely by ID2.
- R3: *<u>Dependency on a Single Individual Risk</u>*
  - Actions we took:
    - Knowledge sharing
    - Documented everything we have been working
  - Reason for carrying it forward:
    - This is still a very big risk since certain people in the team have more knowledge and expertise in certain technologies than others, so the development process can become dependent on them.
- R4: *<u>Requirements Not Fully Understood By Everyone in the Team</u>*
  - Actions we took:
    - Followed the mitigation plan stated in Risk Identification for R4 (and also documented meeting with the stakeholder for better understanding of the requirements)
  - Reason for carrying it forward:
    - As we transition into the implementation phase of the project, it's probable that there will be changes in requirements or shifts in our

understanding of them (so there'll be this risk associated going forward)

- R5: *Team Member Availability Risk*
    - Actions we took:
        - Set up a doodle according to the mitigation plan stated in Risk Identification for R5
    - Reason for carrying the risk forward:
        - There remains a risk of team members being unavailable due to ongoing CSPIP interviews and other commitments (e.g., midterms)
- R6: *Team Member Skill Gap Risk*
    - Actions we took:
        - Followed the mitigation plan stated in Risk Identification for R6 as is.
    - Reason for carrying the risk forward:
        - Many members of the team are unfamiliar with TS, React-Native and AWS Cognito which we plan to use
        - The team is still in the learning process so this risk will certainly persist

Risks that were mitigated and resolved completely (not carried forward):

- R7: *Technology Lacks Resources and Documentation*
    - Actions we took:
        - Followed the mitigation plan stated in Risk Identification for R7 before finalizing the technology.
    - Reason for eliminating the risk:
        - We've selected TS and/or Java Spring (as a backup) for the backend tech, React Native for the front end, and MySql for the database. These technologies offer extensive documentation and learning resources readily available on platforms like Google, YouTube, and GitHub. As a result, we can confidently mitigate this risk.

<u>Additional Risks Identified for the Next Sprint:</u>

1. *<u>Risk Arising from Role Switching Among Team Members:</u>* **(Id: R8)**
    - **Risk:** The team's plan to swap testers and developers poses a risk to workflow efficiency and task completion. With testers assuming development roles and developers taking on testing responsibilities, there's a potential for decreased productivity and delays in deliverables due to unfamiliarity with new roles.
    - **Handling:**
        - <u>Mitigation Plan</u>:
            - Before role switching, each team member should document their completed tasks and responsibilities.
            - Developers are required to compile documentation detailing the tools utilized, codebase structure, setup instructions, and upcoming project objectives.
            - Testers should document their methodologies, testing tools employed, and guidelines for working with those tools as most developers will be unfamiliar with testing.
            - Both teams must maintain summaries of completed, ongoing, and pending tasks to ensure clarity and alignment within the project.
        - <u>Contingency Plan</u>:
            - If team members encounter difficulties adapting to their new roles, assign individuals from the original team to provide guidance and assistance to those transitioning into new roles.
            - If role switching proves to be overly disruptive or ineffective, then redistribute tasks among new team members based on their skills and expertise.
            - At worst, consider temporarily reverting to original role assignments if the switch results in **significant** project delays or quality issues.
2. *<u>Scope Creep Risk</u>*: **(Id: R9)**
    - **Risk:** Scope creep poses the risk of expanding project and/or deliverable boundaries beyond initial specifications, leading to delays, and potential deviations from project/deliverable objectives.
    - **Handling:**
        - <u>Mitigation Plan</u>:

            - Communication between developers and dev lead on features in progress
            - Each issue on GitHub should have clearly defined tasks of what to do.
            - The objectives of each deliverable should be communicated with the rest of the team in a meeting
        - <u>Contingency Plan</u>:
            - Prioritize project objectives and deliverables to focus efforts on essential tasks
            - The leads should communicate with everyone their specific tasks

3. *<u>Gold Plating</u>*: **(Id: R10)**
    - **Risk:** Risk of putting in more hours than necessary to increase software quality/appearance at the expense of time and cost.
    - **Handling:**
        - <u>Mitigation Plan:</u>
            - Prioritization: each feature should be broken down into “nice to have” and “must-have”
            - Effective communication between developers and dev leads to the progress of their work
            - Conducting a review of the time logs to ensure the absence of gold plating
        - <u>Contingency Plan</u>:
            - Prioritize project objectives and deliverables to focus efforts on essential tasks
4. *<u>Unequal team contributions</u>*: **(Id: R11)**
    - **Risk:** It's expected that people in different roles will have varying levels of responsibility, leading to different time commitments. However, if some individuals consistently spend significantly more time than others, especially those in similar roles, it could create negativity within the group and impact the project's success.
    - **Handling:**
        - <u>Mitigation Plan:</u>
            - Tasks are to be divided almost equally among the team members
            - If a team member feels unsure of what they should be doing, they are to check the tasks on Git issues to work on

- If still uncertain, a team member is to contact their team lead for further assistance
- If logs indicate that an individual is consistently logging significantly fewer hours of work compared to the average, then a review or discussion may be necessary to understand the cause and address the issue effectively.

3. Exposure Analysis

Definitions:

Likelihood	Interpretation
1	Rare
2	Unlikely
3	Possible
4	Likely
5	Almost Certain

Consequence	Interpretation
1	Negligible
2	Minor
3	Moderate
4	Major
5	Catastrophic

Risk **(likelihood x consequence)**	**Risk Level**	**Condition**
1 to 6	Low Risk	***Acceptable, but review to see if it can be further reduced***
8 to 12	Medium Risk	***Appropriate management required***
15 to 25	High Risk	***Must not proceed, until adequate actions are taken to minimize the risk***

Risk Matrix

ID	Risk Description	Likelihood	Consequence	Risk Exposure
R1	Technology Incompatibility Risk	3	4	12 Medium Risk
R2	Team Communication Risks	2	3	6 Low Risk
R3	Dependency on a Single Individual Risk	4	4	16 High Risk
R4	Requirements Not Fully Understood By Everyone in the Team	3	4	12 Medium Risk
R5	Team Member Availability Risk	3	2	6 Low Risk
R6	Team Member Skill Gap	3	3	9 Medium Risk
R7	~~Technology Lacks Resources and Documentation~~	~~2~~	~~3~~	~~6~~ Resolved
R8	Role Switching Among Team Members	4	4	16 High Risk
R9	Scope Creep	2	3	6 Low Risk
R10	Gold Plating	2	3	6 Low Risk
R11	Unequal team contributions	4	3	12 Medium Risk