Risk Management Plan

SEG4105 Group Project - G03

Objective

The primary objective of the Risk Management Plan is to identify, assess, and mitigate potential risks that could impact the completion and implementation of this project. This plan is to keep the project on track by ensuring that any potential risks are managed before they become issues, which then protects the project's goals, timeline, and resources.

Risk Inventory

Risk ID	Risk Name	Impact	Likelihood
R-01	API Downtime	High	Medium
R-02	Exceeding API Rate Limits	Medium	Low
R-03	Integration Issues	High	Medium
R-04	Time Constraints	High	Medium
R-05	Scope Creep	Medium	Low
R-06	Code Merge Conflicts	Medium	Medium
R-07	UI-UX Misalignment	High	Medium

Risk Breakdown

R-01: API Downtime

- **Cause:** The dependency on a third-party API for data retrieval and sort functions.
- **Affected Project Areas:** Search functionality, sort functionality, data display, user experience.
- **Impact Severity:** High
    - Core functionality of the website would no longer be available, thus showing the high severity.

- **Mitigation:** Set up error handling for API requests. If the API call fails, the system then triggers a message informing the user of the issue. Display a “Retry” button that allows the user to attempt the action again after a short delay.

R-02: Exceeding API Rate Limits

- **Cause:** Excessive API calls due to unexpected volume of usage.
- **Affected Project Areas:** User experience, API function access reliability.
- **Impact Severity:** Medium
    - The website will still contain its general functionality, but select features would be unavailable
- **Mitigation:** Monitor API usage, implement rate-limiting logic, or restrict API calls.

R-03: Integration Issues

- **Cause:** Complexity in integrating API data with the frontend UI components.
- **Affected Project Areas:** Functionality, data consistency, UI behavior.
- **Impact Severity:** High
    - API integration issues would render the website non-functional
- **Mitigation:** Allocate additional time for integration testing and debugging. Commit to early integration tests so that the design can be changed accordingly..

R-04: Time Constraints

- **Cause:** Tight sprint schedules, potential blockers, or unforeseen issues.
- **Affected Project Areas:** Task completion, feature delivery, sprint goals, user experience
- **Impact Severity:** High
    - Any time constraints may result in later deliveries of promised features which would affect the overall website
- **Mitigation:** Prioritize the essential tasks, have clear timelines, and track progress weekly using Jira and standup meetings among developers.

R-05: Scope Creep

- **Cause:** Changes or additions to the project scope mid-sprint.
- **Affected Project Areas:** Project timeline, feature completion, team workload.
- **Impact Severity:** Medium
    - This risk will negatively affect the development process of the project, but not the final product itself
- **Mitigation:** Define a clear project scope and avoid adding new features without proper discussion. Assign scrum master to monitor scope adherence.

R-06: Code Merge Conflicts

- **Cause:** Multiple developers working on the same files or features concurrently.
- **Affected Project Areas:** Code integrity, development efficiency, Task completion
- **Impact Severity:** Medium
    - This will affect the delivery of the product, but not the overall quality
- **Mitigation:** Schedule regular code reviews, communicate task ownership clearly, and use proper version control branching strategies through GitHub.

R-07: UI/UX Misalignment

- **Cause:** Differences in UI design expectations between developers and stakeholders.
- **Affected Project Areas:** User experience, development rework, development efficiency
- **Impact Severity:** High
    - UI misalignment would result in unhappy stakeholders that would then not be satisfied with the final product
- **Mitigation:** Hold early design review meetings to clarify expectations and finalize design mockups. Make confirmation of designs with wireframes.

Effort Estimates

Each mitigation task will include estimated effort measured in story points, where 1 story point = 1 day:

- **R-01: API Error Message and "Retry" Button** — 2 story points
- **R-02: Rate-Limiting Checks** — 1 story point
- **R-03: Integration Testing and Debugging** — 2 story points
- **R-04: Progress Tracking and Buffer Time** — 1 day per sprint
- **R-06: Code Reviews** — 1 day weekly
- **R-07: Design Review Meetings** — 1 day early in each sprint

Risk Owner Assignments

- **R-01 (API Downtime):** Assigned to Muiz Zafar (Developer)
- **R-02 (API Rate Limits):** Assigned to Azan Mubashar (Developer)
- **R-03 (Integration Issues):** Joint responsibility between Muiz and Azan (Developers)
- **R-04 (Time Constraints):** Scrum Master (Responsible for tracking progress)
- **R-05 (Scope Creep):** Product Owner (Responsible for scope control)
- **R-06 (Code Merge Conflicts):** All Developers (Scrum Master for oversight)
- **R-07 (UI/UX Misalignment):** Azan Mubashar and Product Owner(Developer)

Risk Timeline

Weekly Review: Risks R-01 to R-05 will be reviewed at the end of each weekly sprint to assess any changes or need for escalations.
Daily Standups: Track risks like R-06 and R-07 during daily standups for early detection and quick implementation of mitigation strategies.
Integration Milestones: Review R-03 (Integration Issues) at each feature completion milestone to catch any problems early.

Dependencies and Triggers

Dependencies:

- **Sorting Functionality Dependencies**
    - **SCRUM-6, SCRUM-7, SCRUM-22** (User stories for sorting by alphabetical order, release date, and rating) depend on:
        - **SCRUM-23** (UI element for sorting selection) to allow users to choose the sorting method and in what order they want their results displayed.
        - **SCRUM-24, SCRUM-26, SCRUM-30** (Implementing sorting functions for rating, alphabetical order, and release date) to provide the necessary backend support.
    - **SCRUM-23** (UI sorting element) should be implemented first to ensure the front end can handle sorting inputs.

- **Details Display Dependencies**
    - **SCRUM-12, SCRUM-14, SCRUM-15, SCRUM-16** (User stories for displaying the director, streaming links, trailers, and images) depend on:
        - **SCRUM-32, SCRUM-39, SCRUM-42, SCRUM-43** for the implementation of display functions and any data extraction from the API for each detail wanted by the user.
    - **SCRUM-25** (Display page for selected movie) serves as the base for displaying any selected movie details and should be completed before adding individual data elements.

- **Streaming and Trailer Information Dependencies**

- **SCRUM-14** (User wants to access streaming links) relies on **SCRUM-32** (Display function for streaming services) and **SCRUM-44** (Streaming service discovery function implementation).
- **SCRUM-15** (User wants to see movie trailers) depends on **SCRUM-42** (Trailer extraction function implementation).

- **Data Extraction Dependencies**
  - **Basic Movie Information**
    - **Dependencies:**
      - **SCRUM-19** (Function for title search) is required to retrieve and display basic movie search results.
      - **SCRUM-21** (Function for genre search) is required to retrieve movies by genre.
      - **SCRUM-28** (Function for rating-based search) is required to filter movies by rating.
    - **Impacted User Stories: SCRUM-2, SCRUM-3, SCRUM-4** (search and display of movies based on title, genre, and rating).
  - **Sorting Functions**
    - **Dependencies:**
      - **SCRUM-24** (Sort by rating function), **SCRUM-26** (Sort by alphabetical order function), and **SCRUM-30** (Sort by release date function) are necessary to provide sorted movie lists.
    - **Impacted User Stories: SCRUM-6, SCRUM-7, SCRUM-22** (sorting of search results).
  - **Movie Details Display Page**
    - **Dependencies:**
      - **SCRUM-36** (Display movie title), **SCRUM-37** (Display movie synopsis), and **SCRUM-38** (Display movie poster) are required for the basic display of movie details on the movie detail page.
    - **Impacted User Stories: SCRUM-8, SCRUM-9, SCRUM-10** (display title, poster, and synopsis for selected movies).

- **Streaming and Trailer Information Display**
    - **Dependencies:**
        - **SCRUM-32** (Display function for streaming services) and **SCRUM-44** (Streaming service discovery function) are required for showing streaming options.
        - **SCRUM-42** (Trailer extraction function) is necessary for displaying official trailers.
    - **Impacted User Stories: SCRUM-14** (access to streaming service links) and **SCRUM-15** (display movie trailers).

- **Additional Movie Attributes**
    - **Dependencies:**
        - **SCRUM-40** (Director name extraction), **SCRUM-41** (Release date extraction), and **SCRUM-43** (Official image extraction) are necessary for providing additional movie details.
    - **Impacted User Stories:**
        - **SCRUM-12** (display director name)
        - **SCRUM-13** (display release date)
        - **SCRUM-16** (display official images)

- **Cast Information Display**
    - **Dependencies:**
        - **SCRUM-39** (Cast extraction function) is required to retrieve and display cast details.
    - **Impacted User Story: SCRUM-11** (display cast information).

Triggers:

1. **Event: Stakeholder Requests Additional Features Mid-Sprint**
    - **Description:** Stakeholders might request new functionalities or changes during a sprint (e.g., adding a new sorting filter or additional information in the movie details page).
    - **Escalated Risk: R-05: Scope Creep**
    - **Contingency Action:** Define a clear scope and require that all new requests go through an approval process managed by the Product Owner. Then, ensure that new requests are only addressed at the start of a sprint.

2. **Event: Delay or Incomplete Data Extraction from API for Details Display (SCRUM-32, SCRUM-39, SCRUM-42, SCRUM-43)**
    - **Description:** If the API gives issues during the data extraction, it would make the integration with the frontend components difficult, which then affects the functionality and consistency of the displayed data.
    - **Escalated Risk: R-03: Integration Issues**
    - **Contingency Action:** Implement error handling for missing data, and use placeholder content until the complete data is available to display. Commit to early integration testing to identify any API-related issues early.

3. **Event: Multiple Developers Working on Interdependent Features Simultaneously**
    - **Description:** Developers may be working on related areas of the codebase (e.g., SCRUM-24 and SCRUM-30 for sorting functions), which can lead to merge conflicts if there is overlap or conflicting code in the files.
    - **Escalated Risk: R-06: Code Merge Conflicts**
    - **Contingency Action:** Use feature branches in Git and have developers frequently pull updates from the main branch. Maintain regular communication about which files each developer is working on.

4. **Event: Delay in Display Page Completion (SCRUM-25)**
    - **Description:** The display page is the base for showing movie details, so any delay here could impact multiple dependent tasks and affect sprint goals.
    - **Escalated Risk: R-04: Time Constraints**
    - **Contingency Action:** Prioritize the display page task early in the sprint. Use a simplified version of the display page as a temporary solution for testing until the final version is ready.

5. **Event: Design Changes Made Late in the Sprint**
    - **Description:** Stakeholders might request changes to the design after development has started, which could impact the current sprint.
    - **Escalated Risk: R-07: UI/UX Misalignment**
    - **Contingency Action:** Set a cutoff point for design changes, and ensure that they are finalized and approved before the sprint begins.

6. **Event: Dependency on Basic Search Functionality Not Being Ready (SCRUM-19, SCRUM-21, SCRUM-28)**
    - **Description:** Issues in retrieving data from the API for the basic search functionality will risk the availability of core website functions, which then directly affects the search, filtering, and display functions..
    - **Escalated Risk: R-01: API Downtime**
    - **Contingency Action:** Use mock data for UI testing if API calls are unavailable. Monitor API connectivity closely and plan for error or loading pages if data retrieval is delayed.

7. **Event: API Limitations or Rate Limit Exceeded**
    - **Description:** Exceeding API rate limits due to high usage would lead to the unavailability of certain features that rely on API calls, which affects the user experience.
    - **Escalated Risk: R-02: Exceeding API Rate Limits**
    - **Contingency Action:** Implement rate limiting in the code to avoid any excessive or unnecessary calls. Monitor usage to adjust the frequency of API calls if we feel that it will reach the limit.

Resolution Criteria

For each risk, a checklist of criteria that must be met for a risk to be considered "resolved" is given below:

- **R-01 (API Downtime)**:
    - Upon API failure, the following is true.
        - The error message is displayed properly.
        - The "Retry" button is displayed properly.
        - When the "Retry" button is clicked, the website retries the API request.
        - The user may continue to navigate the website (i.e., the website does not crash).
- **R-02 (API Rate Limits)**:
    - API usage monitor is implemented.
    - Rate-limiting of API calls is implemented.
- **R-03 (Integration Issues)**:
    - For each integration task,
        - A review of expected story points is conducted at least once.
        - Integration tests that verify a successful integration are developed.

- **R-04 (Time Constraints)**:
    - Weekly sprint reviews are upheld.
    - Weekly sprint retrospectives are upheld.
- **R-05 (Scope Creep)**:
    - Project scope is clearly defined.
    - Regular review of product backlog for out-of-scope items is upheld.
- **R-06 (Code Merge Conflicts)**:
    - Weekly sprint reviews are upheld.
    - Proper and effective use of Git versioning is upheld.
- **R-07 (UI/UX Misalignment)**:
    - For each UI/UX update,
        - A design plan (mockup) is presented during the sprint planning session the update is proposed in.
        - A wireframe is developed within the first day of the sprint.

Review and Approval

This risk management plan will be reviewed by stakeholders on Friday, November 1st. It will be reviewed during the sprint review, during which the plan will be presented to stakeholder Shabnam for review and approval. Should this plan see changes or updates in the future, the updated plan will be presented to stakeholder Shabnam at the next sprint review (next available Friday). If, for whatever reason, stakeholder Shabnam is unavailable to conduct their review of the risk management plan, the responsibility will be fully delegated to our Product Owner.