**Hobby Web Application Project Risk Assessment**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **RISK** | **EVALUATION** | **LIKELIHOOD** | **RISK RATING KEY** | **MATRIX VALUE** | **RESPONSIBILITY** | **RESPONSE** | **CONTROL MEASURE** |
| **1** | Project Specification | The project spec is for a Hobby Web Application. The hobby is not defined, allowing for a free approach and method in completing the project. The developer’s idea may stray from the project spec with this freedom. | POSSIBLE | 3 | **HIGH**  **- 8 -** | Project Manager & Software Developer | Clearly define the Acceptance Criteria. Revert back to the Acceptance Criteria before developing new features to not waste time. | Ensure a thorough plan is constructed before the project is initiated. Use sprint reviews to ensure required tasks are completed. |
| **2** | Software Integration | This project utilizes a vast range of programming languages, scripting languages and software. Problems have raised when interacting or extracting data between the software applications in the API. | PROBABLE | 3 | **HIGH**  **- 9 -** | DevOps, Software Developers | Ensure code is clear to narrow the problem down quickly. | Use OOP to ensure code is easily understood. Regularly test new features and ensure the code is working as intended. |
| **3** | Technical Risks – adding new features affecting existing features | The creation and implementation of a new feature can potentially affect previous features in the project, or create new bugs. For example, adding CSS styling to classes, headings or ids have created some unwanted styling or bugs. | POSSIBLE | 4 | **HIGH**  **- 8 -** | Software Testers & Software Developers | Regression Testing – Retest all previous features to ensure they’re working. If some features are not working, compare previous version to current version to narrow down the problem. | Ensure the project has a clear testing procedure which is maintained throughout the scope. This should include Regression Testing, Unit Testing and Integration Testing. |
| **4** | Inaccurate time estimations | Time spent on particular tasks exceeds estimated time. Can lead to delays in project deployment or poor-quality product. Some User Stories defined for this project were under-estimated. | PROBABLE | 2 | **MEDIUM**  **- 5 -** | Software Developers or Software Engineers | Sprint review. Re-allocate the estimated time for future tasks in the project based on previous estimation and completion differences. | Breakdown tasks into sub-tasks and allocate time. Allocate extra time for complex or unfamiliar tasks. Do research prior when allocating time. |
| **5** | Backups & Restores | Not regularly backing-up the code whilst working on new features can cause problems such as: Difficulty reverting changes to previous code given the lack of backed-up versions, poor quality code etc. | IMPROBABLE | 2 | **LOW**  **- 4 -** | Software Developers or Software Engineers | Test the existing product to ensure all features added are working as intended. Implement and encourage more backups with each feature added or task complete. | Ensure after each feature or task is complete, the project is saved and stored with a description of the feature added. This is essential for effective version control. |
| **6** | Testing | New features are added without testing. Errors and bugs may occur as a result. For example, when creating the database and table within the database in MySQL for development, the CRUD functions were no longer functional in the Spring Project. Changes to the Domain and Controller were made to fix this. | POSSIBLE | 2 | **MEDIUM**  **- 5-** | Software Testers & Software Developers | Revert back to previous commits when the features were functional. Compare the altered files and identify potential problems. | Ensure the project has a clear testing procedure which is maintained throughout the scope. Stage changes on GitHub regularly so new problems are identified quickly. |
| **7** | Being too creative or adventurous | Attempting to approach the project using unfamiliar features. For example, when designing the front-end, tables were used to display the created entities. However, being unfamiliar with tables, problems were encountered when attempting to update and delete created entities. | POSSIBLE | 4 | **EXTREME**  **- 11 -** | Software Developers or Software Engineers | Simplify the process by changing the approach to the feature creation. Revert back to previous Git commits. | Ensure in planning that adventurous features are researched fully and allocated enough time for completion. This may not decrease the risk and time to complete the feature, so different approaches should be considered. |
| **8** | Compromising on Designs | Developing the software to meet the acceptance criteria. Several design improvements can be made but limited to time. The main focus in the given timeframe is the Minimum Viable Product (MVP). | POSSIBLE | 1 | **LOW**  **- 2 -** | Software Developers or Software Engineers | Work towards finishing the MVP before considering adding new features or functions to the project. | Use the project Scrum board to add tasks for project completion. Each task should have a priority. High priority tasks should be completed first and extra features should have low priority. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **RISK ASSESSMENT MATRIX** | | |  |  |
| **RISK RATING KEY** | **LOW**  **1**  **Acceptable** | **MEDIUM**  **2**  **Tolerable** | **HIGH**  **3**  **Undesirable** | **EXTREME**  **4**  **Intolerable** |
|  |  |  |  |  |
| **LIKELIHOOD** |  |  |  |  |
| IMPROBABLE | **LOW**  **- 1 -** | **LOW**  **- 4 -** | **MEDIUM**  **- 6 -** | **HIGH**  **- 10 -** |
| POSSIBLE | **LOW**  **- 2 -** | **MEDIUM**  **- 5 -** | **HIGH**  **- 8 -** | **EXTREME**  **- 11 -** |
| PROBABLE | **MEDIUM**  **- 3 -** | **HIGH**  **- 7 -** | **HIGH**  **- 9 -** | **EXTREME**  **- 12 -** |