NoSQLite

Risk Plan

University of Windsor

Objective

This Risk Plan document lists the possible risks involved in development of project - NoSQLite. It mainly captures the type of risk, it's probability and a contingency plan.

Author – Team-1
Recipient – Dr. Kalyani Selvarajah
Written on – 12-06-2021

Index

Risk Plan

Risk ID	Description/Impact	Probability	Contingency plan or Mitigation technique
User Indulgence	The user might not find the library useful or might even loose interest in case library lacks – - Ease of integration - Performance - Simple interface	High	- While designing the interface a user-friendly design shall be targeted and the final product shall be easy to integrate. - Market Research shall be performed before deciding the open-source libraries to integrate. The decided libraries shall be checked for performance, offered features, scalability, and robustness.
Schedule Risk	- A blocker bug can result in revamp of code and thereby delayed release. - Developers fumbling their way around new approach often introduced unacceptable schedule risk.	High	- Identify the critical risk areas by researching similar non-structural database libraries and create a list of known risks. Allocate buffer resources and fix the bugs as soon as possible - Use proper standardized and generic coding formats. When the code is tightly integrated with a unified programming architecture, it will increase the speed of creating complex code while reducing schedule risk.
Technical Risk	Following scenarios are possible: - The major server upon which the library is built fails. - University Server access is not available. - Database access not available.	Medium	Following steps shall be considered: - Regular commits to GitHub. - A backup server or a local setup. - Setup database at the start of project itself to avoid issues at later stage.
Critical resource turnover	Critical resource leaves the project with critical information	High	Encourage pair programming, knowledge base expansion, and self-balancing teams. Code walkthroughs shall be held every alternate week.
Communication Risk	-Documentation isn't clear enough for a user/ client to understand the process of integrating the library. -The client and teammates are unaware of the process and progress	High	-Ensure proper documentation for library. - Ensure proper communication within the team and client document them if necessary
Poor Productivity	In academic group projects, the sense to abide by the timelines is lost due to multiple coursework.	Medium	Introduce short iterations, plan the sprints, and monitor the progress of the allocated tasks regularly via scrums.

	Productive time lost at the early stages impacts all phases of the project		
Performance Risk	Performance lag can create an unpleasant experience for the users	Medium	Test the major functionalities of the library using a different set of data and varying loads. Analyse the performance and improve the areas where it lags
Operational Risks	Setup issues like invalid configuration and discrepancies of settings for Dev and QA setups can result in both time and effort wastage.	High	- Testing shall be continued on a fresh setup. - Setup done shall be in sync with dev environment.
Design Risks	Design of the library can be a constraint for future updates and features for example – We can develop a library right now and, in the future, we require extending it to multiple programming languages and databases and the library does not support it then, redesigning architecture will cost a lot of team efforts and time consumption.	Medium	The scalable design will help mitigate the risk of re-architecture / re-design in case of the addition of components or features. We can design the library in such a way that new modules and features can be implemented with minimal interaction with previous design or architecture.
Scope Risk	When developing a database library, we have to consider different types of use cases and user requirements. Developing a library that is specific to one only one database is not feasible as the current market will require us to integrate multiple databases.	Low	While designing the NoSQLite architecture, the team needs to consider the current requirement and the future enhancements to widen the scope.
Quality Risk	The developed database library meets all the requirements but introduces various kinds of bugs and malfunctions.	High	While developing we should implement various robust testing and validation processes to make sure that the library does not contain any critical bugs or database connection breaking functionality.