Help Queue Risk Assessment

Background

The Help Queue project may encounter some risks, especially as it is over a number of weeks and during work time. This document aims to identify those risks in the Risk Matrix and steps taken to Mitigate/Transfer/Accept those during the course of the project

Risk rating

The below highlights the risk levels with likelihood and severity of those risks if they were to occur. A detailed look is in the Risk Matrix table.

LOW	MEDIUM	HIGH	EXTREME
- Acceptable - Ok to proceed	- As low as reasonably practicable - Take mitigation efforts	- Generally unacceptable - Seek support	- Intolerable - Place event on hold

LIKELIHOOD	SEVERITY			
	ACCEPTABLE *Little to no effect on event*	**TOLERABLE** *Effects are felt, but not critical to outcome*	**UNDESIRABLE** *Serious impact to the course of action and outcome*	**INTOLERABLE** *Could result in disaster*
IMPROBABLE *Risk is unlikely to occur*		**Project Spec Creep**	**Code Loss Risk**	**Data Risk**
POSSIBLE *Risk will likely occur*			**Technical Risk**	**Time Risk**
PROBABLE *Risk will occur*				

Risk Matrix

Risk Name	Risk Description	Evaluation	Likelihood	Impact	Responsibility	Response	Control Measure
Data Risk	Data for the database could be lost/removed, such as deletion of SQL files.	Database would have no data which would mean the application cannot manage tickets, nor could any development work happen.	IMPROBABLE	HIGH - (INTOLERABLE)	@ parvir chomber	- Create SQL scripts again from existing ER Diagram - Try to look in Local History of IDE to recover scri	- Mitigate - Store SQL scripts in Github - Keep initial scripts short
Time Risk	The project is to take place over a month during work hours. Time to allocate to the project may be difficult depending on work schedule	The application would not have all the required functionality along with three features, and a part of it could be unfinished.	POSSIBLE	VERY HIGH - (INTOLERABLE)	@ parvir chomber	- Inform QA of the situation - Try to complete as much as possible	- Mitigate - Block out time in calendar to dedicate time to work on the project - Evaluate time at the end of every sprint and adjust time spent.

Technical Risk	The project requires some understanding of initially 3 parts, the UI, API and Database. Later knowledge of deployment on AWS /Kubernetes/Docker will be required, which may be lacking.	Application development may stall and story point targets may not be hit when arriving at the unfamiliar parts of the project. This might result in the application being undeployable or a mistake happen during deployment.	POSSIBLE	HIGH - (UNDESIREABLE)	@ parvir chomber	- Inform QA trainers to seek support - Inform Barclays mentor for software related issues - Take time on Pluralsight /other learning resources - Seek peer help with those familiar with the technologies.	- Mitigate - Read ahead /brush up on technologies known in the project - Try to setup my own pipeline using free resources - Accept - Factor this time in during story point estimation
Project Spec Creep	Hours maybe spent on trying to beautify the UI when its not required, or add features to the API which functionally do not add value to the final project.	This might result spending lots of time on one feature to make it look good when time could have been spent on other functionality.	IMPROBABLE	MEDIUM- (TOLERABLE)	@ parvir chomber	- Refer back to Jira and the product backlog and what constitutes as done. - Re-read project spec - Move on to another required de	- Mitigate - Consistently refer back to the ticket being complete. - Review after every sprint with a retrospective to see if time was better spent on another feature.
Code Loss Risk	The UI/API code bases may be lost due to forgetting to commit often (and not saving locally) OR power loss of main computer. This has occurred recently.	In the event of a power loss then any un-committed changes would be lost, which could mean loss of work /time waste.	IMPROBABLE	HIGH - (UNDESIREABLE)	@ parvir chomber	- Inform Barclays mentor - Pull latest versions from Github - Migrate work from personal computer to a laptop	- Mitigate - Commit/Push to github regularly through each ticket - Configure laptop to have the same software as main computer

✅ Action items

☐