ENG1 Assessment 1: Risk Assessment

Greenfield Development

Group 5

Members

Mark Faizi

Ben Faulkner

Christian Foister

Lewis Hanson

Ziyad Rashad Hassan

James Haworth

Risk Assessment

Our team adopted a structured risk management process to identify, assess, and mitigate risks throughout the software development lifecycle. Given the small-scale nature of the project, we maintained a straightforward yet effective approach to ensure that all potential risks were adequately addressed.

1. Risk Identification: Risks were initially identified during the planning phase and continually reviewed throughout development. Regular team discussions ensured that any emerging risks were promptly identified.
2. Risk Assessment: Each identified risk was evaluated in terms of its likelihood of occurrence and potential impact on the project. A simple scale of Low (1), Medium (2), and High (3) was used to quantify both likelihood and impact.
3. Risk Mitigation: Mitigation strategies were devised for each risk, focusing on either reducing the likelihood of the risk occurring or lessening its impact on the project. Preventative measures and contingency plans were established where necessary.
4. Risk Ownership: Each risk was assigned to a team member responsible for monitoring and managing the risk. The owner would act if the risk materialised or showed signs of escalation.
5. Risk Monitoring: Risks were reviewed at regular project milestones, ensuring that mitigation plans were up to date, and any new risks were properly managed.

Our risk register is structured with the following columns

---

b) Risk Register

	Description	Likelihood	Impact	Mitigation	Member
R1	Team member unavailable due to illness or personal reasons.	Medium (2)	Medium (2)	Ensure thorough documentation of tasks and knowledge sharing across the team. Assign secondary responsibilities to other team members.	
R2	Misunderstanding of customer requirements leading to incorrect implementation.	Medium (2)	High (3)	Hold regular meetings with the customer for clarification. Ensure all requirements are documented clearly	

				and signed off by the customer.	
R3	Integration problems between different modules developed by team members.	Medium (2)	High (3)	Conduct frequent code reviews and perform early integration testing. Clearly define module interfaces and communication protocols.	
R4	Scope creep resulting from evolving customer demands or feature requests.	Low (1)	High (3)	Define the project scope at the outset and handle changes through a formal change request process. Communicate scope limits to the customer regularly.	
R5	Delays due to reliance on external libraries or tools.	Low (1)	Mediu m (2)	Select well-documented, widely used libraries. Include buffer time in the project schedule to account for potential issues with third-party tools.	
R6	Lack of sufficient testing leading to undetected bugs or usability issues.	Medium (2)	High (3)	Implement test-driven development (TDD) and conduct both unit and integration testing throughout development.	

R7	Insufficient time for bug fixing and polishing before the deadline.	Medium (2)	High (3)	Set internal deadlines for feature completion to allow time for testing and refinement. Prioritise key features and critical bugs.	

Rating System

- Likelihood:
    - Low (1) – Unlikely to happen
    - Medium (2) – May occur
    - High (3) – Likely to happen
- Impact:
    - Low (1) – Minor effect on the project
    - Medium (2) – Moderate effect on the timeline or quality
    - High (3) – Significant disruption to the project