Software Project Risk Identification & Mitigation

Prepared by:

SOOBRAYEN Keshia 2412920

BEEDASSY Nirvana Luxmi 2413850

RAGHOONUNDUN Rishab 2412024

AUCKEL Ayush Manav 2412549

THUMMANAH Kentish 2411831

Course : BSc (Hons) Computer Science

Module : ICT 1208Y Software Engineering Principles

Lecturer : Dr. Mungur Avinash Utam

Date : 5th of April 2025

Table of Contents

1.0 Software Project Management Plan Overview

1.1 Project Scope

The Food Delivery Management System (FDMS) represents a solution to streamline multiple operations of food ordering with automated delivery logistics and restaurant coordination services. Customers use this system to select food items from menus while also placing their orders and paying for them and following their delivery status. Subsystems include:

- **Customer Ordering Subsystem:** Allows users to browse, select, and order food from a variety of restaurants.
- **Restaurant Management Subsystem:** Assists restaurants with keeping the orders, their menus, and overall management via the platform.
- **Delivery Management Subsystem:** Manages the deliveries, optimize route building, driver assignment, and real-time updates.
- **Payment Gateway Subsystem:** Handles payments safely.
- **Admin Dashboard Subsystem:** Administrators have control over the whole process of the system.

1.2 Project Timeline

Phase	Duration	Tasks
Requirements Gathering	1 week	Define features, gather user stories
Design	1 week	UI/UX mockups, architecture design
Development	3 weeks	Coding each subsystem in phases
Integration	1 week	Combine all subsystems and test APIs
Testing & QA	2 weeks	Unit testing, system testing, bug fixing
Deployment	1 week	Deploy on cloud, setup database
Maintenance	Ongoing	Bug fixes, feedback implementation

1.3 Key Stakeholders

- **Project Manager:** Leads the entire project process to achieve all defined targets within set timeframes and financial limits and coordination demands.
- **Developers:** Design the system's functionalities and code implementation while resolving technical faults during the process.
- **QA Engineers:** Test the system by validating performance limitations while ensuring complete quality standards compliance for the final product.
- **Restaurant Partners:** Manages menu distribution and executes order delivery alongside delivery staff cooperation.
- **Delivery Staff:** Act as the order picker between restaurants and customers to handle quick delivery services.
- **End Users (Customers):** The system's main users who perform browsing, ordering, payment processing and provide feedback to the system.

1.4 Project Objectives & Expected Outcomes

- **Build a scalable and secure food delivery application:** Design a platform which will scale according to business needs but also implements security standards in the infrastructure.

- **Enhance customer experience and reduce delivery time:** The application requires better interface design with efficient delivery algorithms for improved user satisfaction and fast deliveries.

- **Enable real-time order and delivery tracking:** The application should have real-time tracking functions through GPS and tracking Application Programming Interfaces (API) to show delivery status updates.

- **Support seamless interaction between restaurants and delivery staff:** The system should enable effortless communication among restaurants and their delivery staff through implemented features for notifications and coordinated task management.

2.0 Risk Identification

2.1 Potential Risks

Risk Number	Category	Risk Description	Likelihood	Impact
1	Technical	Inaccurate requirement gathering	Medium	High
2	Technical	Poor performance under high load	High	Medium
3	Technical	Server downtime	Medium	High
4	Technical	Size underestimation for system development	Medium	High
5	Operational	Delay in development timeline	High	High
6	Operational	Payment gateway failure	Medium	High
7	Operational	Poor delivery schedule	Medium	medium
8	Operational	Unreliability of project team	Medium	High
9	Security	System security breach	Medium	High
10	Security	Customer data loss or corruption	Low	High
11	Financial	Underestimated budget	Medium	Medium
12	Legal	Legal non-compliance with data regulations	Low	High

2.2 Risks Description

1. **Inaccurate requirement gathering**

   A system developed outside the scope of business needs emerges when stakeholders' requirements are poorly identified during the development process.

2. **Poor performance under high load**

   The system lacks ability to manage a high volume of simultaneous user transactions which leads to program delays until it reaches its timeout limit or results in complete system breakdown.

3. **Server downtime**

   Server outages and maintenance actions that happen suddenly lead to service disruptions which negatively impact user experience and operational processes.

4. **Size underestimation for system development**

   When project teams fail to accurately assess system requirements their tasks become longer and more expensive causing resources to exceed initial estimates.

5. **Delay in development timeline**

   The failure to achieve project milestones makes delivery schedules unavailable due to technical, resource or planning issues.

6. **Payment gateway failure**

   Payment processing system breakdowns or technological errors result in customer transaction failures which cause trust issues with the service providers.

7. **Poor delivery schedule**

   Unrealistic delivery deadlines create delays along with coordination problems which result in product quality reduction and dissatisfied customers.

8. **Unreliability of project team**

   A team's (developers, QA testers, and project managers) poor performance and communication issues lead to delayed tasks together with reduced quality standards as well as project success deterioration.

9. **System security breach**

   The unauthorized entry of the system leads to compromised sensitive data and reduced functionality which might result in both legal ramifications and negative reputational impacts.

10. **Customer data loss or corruption**

    An incident that results in customer data loss or damage because of technical problems will trigger both compliance problems and unhappy customers.

11. **Underestimated budget**

    Tangible outlays might exceed forecasted amounts because of budgeting unpredictability as well as resource allocation requirements and project expansion needs.

12. **Legal non-compliance with data regulations**

    Organizations that violate legislation such as GDPR or HIPAA face penalties and legal consequences alongside damaging their professional image.

3.0 Risk Analysis & Prioritization

3.1 Top 5 most critical Risks and Justification

1. **Delay in development timeline (Operational)**

Such risk stands as an urgent matter because its combination of high probability and substantial impact ranks it among the highest priority factors. Project delays will cause budget overruns because the prolonged duration raises both labor and operational expenses. Project failure to meet deadlines drives the loss of market opportunities when competitors succeed in releasing similar software before the project team does thus devaluing the project along with its market position. Suppliers face unpredictable delays which cause organizations to miss stakeholder expectations thus damaging their reputation while lowering trust between clients and investors. The process of development extending beyond normal timelines creates negative effects on employee morale together with team operational performance. The speed of markets today demands quick delivery so delays in schedule will result in major financial losses and strategic disadvantages (Schwalbe, 2020). Proper planning with backup strategies stands as a vital requirement for achieving project success due to this risk.

2. System security breach (Security)

The probability of system security breach stands at medium while its impact is high so projects with digital transactions and online platforms need to prioritize this risk carefully. Modern organizations require cybersecurity to be their fundamental business priority because of the digital world we live in today. The exposure of sensitive customer information due to system breaches creates legal liabilities under GDPR and may produce extensive consequences that include heavy fines and legal suits as well as damage to company reputation (Pfleeger & Pfleeger, 2015). Any such security breach damages customer faith in the platform which results in reduced platform usage and business losses alongside lasting harm to the brand reputation. Single breaches of service-based platforms erase all the years of platform development since trust forms the core of such platforms. The required protection needs to consist of encryption practices and regular audits as well as secure system designs to safely defend against this risk.

3. Unreliability of the project team (Operational)

Logistics-based services which include online food delivery particularly experience high impact from this risk that demonstrates medium probability. An operative risk stems from unreliable work performance of project team members since team consistency is crucial for project success. Workflow interruptions triggered by employee illness or team member departures or scheduled absences achieve multiple adverse effects which result in delayed essential responsibilities and knowledge depletion. The online delivery platform experiences service quality degradation and revenue loss through customer disappointment due to brief disruptions in servicebased operations. The imperative nature of effective human resource management combined with contingency planning makes sure this risk stays minimized (Heldman, 2018).

4. Inaccurate requirement gathering (Technical)

Software and system development environments are particularly at risk because this issue has a medium likelihood to occur along with high potential impact. A system built from inaccurate or incomplete requirements demonstrates user need failure which requires additional work and causes scope extension and higher expenses and delays in project completion. The early development of mistakes creates prolonged challenges which increase both the difficulty and expenses needed to fix problems later on. When essential user requirements fail to be identified teams might create an incorrect system design that leads users to both dislike it and reject its use. Frequent changes in the development process create setbacks and resource consumption while reducing employee enthusiasm. Defects in requirements at this stage will negatively impact the entire project process because requirements establish the project's structural foundation. (Pressman & Maxim, 2014).

5. Server downtime (Technical)

The compromise of server availability represents a significant threat to online businesses due to its medium appearance rate but consequential effects on platform operation. Server unavailability creates multiple problems that disrupt services and trigger transaction errors and diminish user experience standards. Brief periods of system downtime at high usage times result in both financial losses and diminished customer faith. Continuous system downtime leads both to customer retention problems and damaged brand image because users cannot access the system around the clock. The interruption causes setbacks in both internal business operations as well as service delivery delays. Infrastructure problems related to poor scalability or inadequate monitoring in the system could be signalling point. (Sommerville, 2016).

4.0 Risk Mitigation Strategies

4.1 Mitigation Plan

Risk Number	Risk Description	Mitigation technique
1	Inaccurate requirement gathering	➢ Conduct user story mapping sessions with stakeholders. ➢ Use prototyping to validate requirements early. ➢ Implement sign-offs at each requirement phase.
2	Poor Performance Under High Load	➢ Perform load testing during development. ➢ Optimise database queries and caching ➢ (e.g Redis).

3	Server Downtime	➢ Use redundant servers (failover systems). ➢ Monitor uptime with tools with Nagios/Prometheus.
4	Size Underestimation for System Development	➢ Break project into smaller sprints for better estimation. ➢ Use historical data from past projects.
5	Delay in Development Timeline	➢ Use Gantt charts and Critical Path Method (CPM). ➢ Hold daily stand-ups to track blockers
6	Payment Gateway Failure	➢ Integrate multiple payment providers (e.g., Stripe + PayPal). ➢ Test gateway APIs in sandbox environments.
7	Poor delivery schedule	➢ Real time tracking of delivery drivers ➢ Use a backup system made up of freelance drivers or part-time drivers.
8	Unreliability of project team	➢ Pair programming for knowledge transfer. ➢ Document processes in Confluence.
9	System Security Breach	➢ Conduct regular penetration testings and code audits. ➢ Implement multi-factor authentication(MFA).
10	Customer data loss or corruption	➢ Perform regular backups or automated backups. ➢ Database should make use of ACID properties.
11	Underestimated budget	➢ Include a 10 – 20 % contingency buffer ➢ Track expenses using tools like QuickBooks

12	Legal non-compliance with data regulations	➢ Consult legal experts for GDPR compliance. ➢ Audit data handling processes quarterly.

4.2 Mitigation Strategies

1. Inaccurate requirement gathering

Projects get failed most of the times due to inaccurate or incomplete requirement gathering. To reduce this risk, agile development methods that support continuous involving the stakeholders during the life cycle of the project are a must. Changing requirements and uncertainties can be captured early through regular meetings and requirement reviews and prototypes sessions (Moe et al., 2017). Furthermore, through techniques such as user stories and acceptance criteria requirements are clarified and implemented accurately (Nerur & Balijepally, 2015).

2. Poor performance under high load

The system needs to be capable of dealing with very high transaction volumes, especially during peak hours. To catch bottlenecks in the architecture at different stages of development, it is advised to perform performance testing using load simulators like Apache JMeter. In addition, system performance is also affected by architectural optimization such as use of the caching mechanism, database indexing and scalable cloud services (Pressman and Maxim, 2020).

3. Server downtime

The entire food delivery service can be interrupted by server unavailability. Various redundancy and failover mechanisms such as deploying multi-zone on cloud platforms (AWS, Azure) should be used for the purposes of mitigating this. Routine preventive maintenance and the use of monitoring tools such as Nagios or New Relic can prevent failure or downtime due to issues (Bass et al., 2022; Kim & Kim, 2021).

4. Size underestimation for system development

Underestimating the size and complexity of the system can result in schedule and cost overruns. The team will use Function Point Analysis (FPA) and judgement to estimate efforts and duration.

The project will be divided into separate modules that each have specific time allocation. A 15–20% buffer section exists in the scheduling plan to handle unpredicted complexities. The Agile model enables us to deliver parts of the software in smaller increments which minimizes the risks associated with extensive feature groups (Schwaber & Sutherland, 2021).

5. Delay in Development Timeline

The adoption of Agile methodology through Scrum sprints ensures timed delivery of products supported by immediate feedback processes. The project uses Jira to track task management although its progress. Through daily stand-ups and sprint retrospectives with velocity tracking the system will detect early delays that allow resource redistribution when necessary (Schwaber & Sutherland, 2021; Kerzner, 2022).

6. Payment Gateway Failure

To manage payment failures, multiple payement gateways like Stripe, PayPal will be integrated to implement failover mechanisms. The platform will include retry logic combined with payment attempt logging and transaction recovery procedures. Payments that fail will be recorded by both automated and manual systems until an administrator reviews them (Pfleeger & Atlee, 2017; Nair & Shah, 2019).

7. Poor delivery schedule

The delivery module enables tracking of drivers in real-time through its component. The company will sustain a backup system made up of freelance or part-time drivers. The system will distribute tasks preferentially to drivers who are currently available or located near their delivery routes to minimize delays. Staffing forecasts dependent on both historical patterns and seasonal patterns will be generated by the HR system (Kerzner, 2022; Liu & Guo, 2021).

8. Unreliability of project team

Our technical team will establish clear documentation rules alongside methods for knowledge sharing that include mentoring sessions and code evaluation and pair programming techniques. The project relies on Confluence together with Git as its main documentation management platforms. A transfer procedure must be followed by each main project team member before their departure from the project (PMI, 2017)

9. System security breach

The implementation of security features will abide by OWASP standards through processes such as input validation together with secure authentication functions and session management protocols. Compatibility standards and encryption rules (SSL/TLS) along with access control policies will be implemented. Testing of penetration security and code analysis will be conducted on a regular basis. The organization will develop its Security Incident Response Plan (SIRP). (Stallings, 2018; Sarker et al., 2021).

10. Customer data loss or corruption

The database will support ACID properties (Atomicity, Consistency, Isolation, and Durability) while its daily backups go to a protected cloud storage. The system enables quick recovery by maintaining database replication features together with logging procedures. Data validation implemented across client side and server will secure information from corruption. Regular backup restoration drills are planned to take place (Peltier, 2019)

11. Underestimated budget

A detailed budget needs to be developed with contingency reserves amounting to 15–20% of the total budget. The project team will utilize MS Project or Trello with budgeting plugins as cost tracking tools to monitor actual versus estimated costs data. Budget reviews that happen frequently with scenario-based forecasting methods will maintain budget health across the entire project period (Project Management Institute, 2017).

12. Legal non-compliance with data regulations

Data protection laws have severe legal penalties for non compliance. Therefore, in the early stages of system design, legal advisors in data protection fields should be consulted, to prevent this. In addition, the system has to be regularly audited for GDPR (General Data Protection Regulation) and local data privacy compliance as well as privacy impact assessments (PIAs) should be done, if required (Voigt & von dem Bussche, 2017).

5.0 Risk Monitoring & Review Plan

5.1 Risk Monitoring Strategy Across the Project Lifecycle

Throughout the project life cycle, regular updates should be sent by the team and project manager so that all project team and stakeholders are on the same page (Asana 2025). As a risk register is created that records all details of potential risks and helps to monitor and control risks, any change occurring must be monitored and adjusted depending on the priority level (Asana 2024).

Lifecycle	Phase	How to Track Risks
1	Requirements Gathering	Identify unclear requirements and scope with stakeholders or clients by conducting interviews and workshops (PMI 2017).
2	Design	Review system architecture for workability and constraints by including security and scalability risks in the design (Schwalbe K, 2015).
3	Development	Track progress using tools like Jira and making use of automated testing thus reducing human error. Training and support can be given to identified persons that have skill gaps (Heldman, K., 2018).
4	Integration	Focus should be made for system incompatibility and dependency issues.
5	Testing & QA	Perform risk-based testing to prioritize high-impact areas and ensure analyzing defect trends to point out any systematic risks (Lewis, J., 2008).
6	Deployment	A deployment checklist is created with rollback procedures to communicate clearly with stakeholders (Kerzner, H., 2017).
7	Maintenance	Monitor system logs and incidents to flag new risks and conduct regular reviews.

Risk monitoring frequency: Bi-weekly

Risk monitoring frequency refers to how often the risk assessor will check on and analyse the risks to ensure they remain manageable.

Responsible Risk Assessor: Project Manager

As a project manager is the bridge between the development team and the stakeholders of the project, this makes him the best suitor for responsibility of risk assessment.

5.2 Report and Management of Risks

The project manager will maintain a bi-weekly report of the risk assessment by using a risk register and using project management tools such as Jira and Trello. The project manager will regularly assess each of the identified risks to decide whether that risk is increasing in terms of probability or not.

5.3 Contingency Plan

A contingency plan is a backup plan designed to assess unexpected events that could impact the project's timeline, budget or quality (Wrike. (n.d.)).

Steps to take when risks become critical:

1. **Trigger Conditions:**
   Each high-priority risk will have clearly defined trigger conditions such as early warning signs or thresholds that indicate when the contingency plan should be activated (IEEE, 1998).
2. **Predefined Response Protocols:**
   Each risk in the risk register should be linked to a documented contingency response. These should be developed during the planning phase and refined during risk reviews (PMI, 2017).
3. **Resource Allocation:**
   A portion of the project's budget and resources should be set aside specifically for risk responses and set as an emergency fund.
4. **Documentation and Communication:**
   When a contingency plan is triggered, the event is logged in the Risk Register, stakeholders are informed, and project documentation is updated to reflect changes (PMI, 2017).

5. **Post-Event Review:**

   After executing a contingency plan, a brief post-mortem assessment of the effectiveness of the response is conducted and the remaining project risks are reassessed for any knock-on effects (PMI, 2017).

6.0 Additional Techniques and Tools for Effective Risk Management

Although it is the case that conventional risk management methods, including tools like risk registers and qualitative and quantitative analysis, are fundamental building blocks for effective risk management, there are several more advanced tools and techniques that can greatly enhance the accuracy and responsiveness of risk management initiatives. Not only do these more advanced methods offer better visibility into the risks that may impact a project, but they also facilitate the development of proactive measures to mitigate them. Moreover, they foster better coordination among all stakeholders of a project, thus enhancing a more concerted approach to risk management.

6.1 Risk Burndown Charts

Risk Burndown Charts are truly powerful visualization tools, giving project teams a highly effective, clear, and simple way of monitoring and carefully assessing the up-to-the-minute status of numerous risks across the entire project lifespan. These descriptive charts visually depict how the total risk exposure—which includes all uncertainties and potential hazards that the project can encounter—is predicted to decrease in size as varied mitigation tactics and strategy are implemented thoughtfully and in a systematic manner. Through careful comparison and review of planned results vs. actual results in terms of closing risks, project teams can strictly assess and identify whether their risk management plans are, in fact, becoming truly efficient and effective. This tool is even more important in agile environments, where it is key for project teams to continually perform iterative reviews and assessments of risk advances from time to time, thus ensuring they are highly attentive to any transformations or progress made. (Sommerville I, 2016).

6.2 Failure Mode and Effects Analysis (FMEA)

FMEA, or Failure Mode and Effects Analysis, is a structured, step-by-step process used to identify possible modes of failure within a system. The technique also entails analyzing the possible effect of these failures and estimating the risks involved in advance, ranked according to three primary factors: severeness, likelihood, and detectability. By performing FMEA early in the project cycle, teams can address important points of potential failures proactively before these can impact project outputs negatively. FMEA is highly useful in the engineering and manufacturing sectors, but it's also being used in software development, as well as in services projects, as a way of effective risk avoidance. (Software Engineering 10th ed.)

6.3 Simulation Tools

Sophisticated software tools, ranging from highly specialized applications such as Risky Project and Microsoft Project, which have powerful risk analysis capabilities, to others, allow professionals to carry out simulation-based risk analysis very effectively. These newer tools have the ability to model the effect of different risk scenarios that may emerge during a project, using advanced methods such as Monte Carlo simulation to accomplish this. By simulating various probable outcomes based on the recognized risks, project managers can gain an understanding of the extent of probable impacts that may take place. This thorough knowledge enables them to make knowledgeable decisions regarding key factors such as resource allocation, project timing, and proper contingency planning, and ultimately, overall project success. (Pearson Education; Project Management Institute, 2017)

6.4 Benefits of Using Risk Management Tools

The use of modern tools in the practice of risk management has become a critical necessity in the context of the modern project management environment. These advanced and complex tools not only help a great deal in identifying and analyzing possible threats or dangers, but they also include complete systems that are meant to take counteractive steps to respond to such threats. The use of these modern tools in the different working processes related to project management brings in a whole range of advantages that play a direct role in increasing the chances of success for the project, improving the overall efficiency, and creating higher levels of confidence and assurance for the stakeholders involved in the project.

1. Improved and More Efficient Decision-Making Processes

One of the most strikingly valuable benefits resulting from the efficient use of risk management tools is the excellent boost in decision-making processes, one that can be incredibly useful in a wide variety of disparate contexts and environments. The newest and most modern tools were thoroughly designed and developed specifically to compile, thoroughly examine, and excellently present risk data in a structured, organized way, one not just aesthetically pleasing but also highly intuitive in use. In addition, the wide variety of functions delivered by these new tools, such as risk heat maps, probability-impact matrices, and extensive trend analysis, brings essentially invaluable insights into the nature, likelihood, and possible impact of a diverse range of disparate risks, which are capable of influencing organizations in many, diverse ways.

As a consequence of the adoption of sophisticated data analytics and decision-making tools, project managers and stakeholders are much better positioned to make strategic and informed decisions that influence the success of their projects. Instead of basing decisions on gut instinct or what may be incomplete and unreliable data, their decisions are now strongly underpinned by exhaustive real-time information and rigorously analyzed historical trends that offer great insights. This forward-thinking methodology results in more precise predictions of project outcomes, better prioritization of tasks according to key drivers, and careful contingency planning that considers multiple scenarios—ultimately bringing about a significant decrease in the risk of project failure, thereby improving overall project success and effectiveness.

2. Ongoing and Instantaneous Observation with Notifications

Current risk management software comes with highly sophisticated real-time monitoring features, which allow project groups to keep a persistent and careful watch in real-time on the up-to-the-minute status of already identified risks. Using interactive and real-time dashboards, giving real-time information, automated periodic alarms, dispatched from time to time, along with frequent reports, these groups are able to keep themselves adequately informed regarding any upcoming dangers or substantial risks in the change in the magnitude of the already known risks, capable of blocking their path.

The capability of acting in real time is crucially important and essentially critical in project arenas typified by relentless change and continually shifting conditions, in which a number of factors can change rapidly, in many cases unpredictably. By giving project managers real-time notification of developing or rising dangers, they are in a position to take immediate corrective action without hesitation or uncertainty—this may include changing project timelines, in effect redirecting resources in response to new requirements, or adopting effective counter-measures, as required. Such responsiveness through decision-making reduces enormously the total response time, and is key in maintaining consistently project momentum during the project's entire lifecycle.

3. Improved Communication and Comprehensive Documentation

Successful and effective risk management requires the active involvement and coordination of a wide range of stakeholders who can come from different departments of the same organization, or maybe from different organizations altogether. The tools used in risk management often have integrated collaborative platforms that are specially designed to promote teamwork and improve communication among all the stakeholders of the process. Team members can access, read, update, and comment on the risk registers in real time, while also being able to delegate certain tasks to individual team members and effectively track the progress of mitigation activities.

These tools play a fundamental role as efficient centralized repositories that contain all the documentation that pertains to risk management and its various related processes. By integrating features like version control, detailed audit trails, and carefully designed templates, it ensures that every activity performed within the system is completely traceable and has strict compliance with the defined governance standards. This all-inclusive and comprehensive methodology greatly promotes a culture that is marked by accountability and transparency. In this setting, every person involved has a clear picture of his or her individual roles and responsibilities when it comes to managing risk. Further, they have ready access to the most updated and pertinent information available to them.(Sommerville, 2016; PMBOK® Guide)

7.0 References

1. Pressman, R.S. and Maxim, B.R. (2020). *Software engineering : a practitioners approach*. New York, Ny: Mcgraw-Hill Education.

2. Bass, L., Clements, P. and Kazman, R. (2012). *Software Architecture in Practice*. Addison-Wesley.

3. ISO/IEC 27001:2013 Information Technology. (2013).

4. Kerzner, H. (2022). *Project management: a Systems Approach to planning, scheduling, and Controlling*. 13th ed. John Wiley & Sons, Inc.

5. Moe, N. B., et al. (2017). The impact of agile practices on the software development process. *Journal of Software: Evolution and Process*, 29(8), e1936.

6. Project Management Institute (2017). *A guide to the project management body of knowledge*. 6th ed. Newtown Square, Pennsylvania, USA: Project Management Institute.

7. Nair, S., & Shah, S. (2019). Resilience of payment systems: A framework for financial operations. *Information Systems Frontiers*, 21(1), 141-157.

8. Nerur, S., & Balijepally, V. (2015). The role of requirements engineering in agile software development. *International Journal of Software Engineering and Knowledge Engineering*, 25(3), 443-460.

9. Peltier, T. R. (2019). *Information Security Policies, Procedures, and Standards: A Practitioner's Reference* (3rd ed.). Auerbach Publications.

10. Schwaber, K., & Sutherland, J. (2021). *The Scrum Guide: The Definitive Guide to Scrum: The Rules of the Game.* Scrum.org.

11. Asana. (2025) '6 Steps to Requirements Gathering for Project Success'. Available at: https://asana.com/resources/requirements-gathering (Accessed: 7 April 2025)

12. Asana. (2024) 'Step-by-step guide to the risk management process'. Available at: https://asana.com/resources/project-risk-management-process (Accessed: 7 April 2025).

13. Schwalbe, K., 2015. *Information Technology Project Management*. 8th ed. Boston, MA: Cengage Learning.

14. Heldman, K., 2018. *Project Management JumpStart*. 4th ed. Hoboken, NJ: Wiley.

15. Lewis, J., 2008. *Software Testing and Continuous Quality Improvement*. 3rd ed. Boca Raton, FL: Auerbach Publications.

16. Wrike. (n.d.). 'What is a Contingency Plan in Project Management?'. Available at: https://www.wrike.com/project-management-guide/faq/what-is-contingency-plan-in-projectmanagement/ (Accessed: 7 April 2025).

17. IEEE, 1998. *IEEE Std 1058-1998 – IEEE Standard for Software Project Management Plans*.

18. New York: Institute of Electrical and Electronics Engineers.

19. Pfleeger, Charles P, et al. *Security in Computing*. India, Pearson, 2009. . [Accessed 5 April 2025]

20. Sommerville, Ian. *Software Engineering.* S.L., Pearson Education India, 2016.