

Comparison of Risk Management between Waterfall, Prototype and Incremental Development

Fidya Farasalsabila^{a,1}, Supardi^{b,2} Aulia Agung Dermawan^{c,3}, Alvendo Wahyu Aranski^{d,4}, Is Fadhilah^{e,5}, Machsun^{f,6}, Ahmad Rizky Saputra^{g,7}

^{a,c,d,g}Institut Teknologi Batam, Jl. Gajah Mada, Tiban, Kota Batam, 29425, Indonesia

^bPoliteknik Pariwisata Batam, Jl. Gajah Mada, Tiban, Kota Batam, 29425, Indonesia

^{e,f}IBMT, Jl. Raya kupang-Surabaya- Jatim, 60189, Indonesia

fidya@iteba.ac.id¹, supardimm14@gmail.com², agung@iteba.ac.id³, alvendo@iteba.ac.id⁴,
isfadhilah@gmail.com⁵, machsun.bb@gmail.com⁶, 2421057@student.iteba.ac.id⁶

ABSTRACT

Software development projects are exposed to various risks related to requirements, people, strategy, framework, quality, deployment, and changes during development. Although previous studies have examined individual software development models and their associated risks, evidence remains fragmented and there is limited systematic comparison of Waterfall, Prototyping, and Incremental models from a risk-management perspective. This study aims to systematically compare the risk characteristics and mitigation approaches associated with these three software development models. A Systematic Literature Review (SLR) was conducted following a structured study identification, screening, eligibility assessment, quality assessment, and data synthesis process. Sixteen studies were included as the primary studies for analysis. The selected studies were mapped according to software development model, risk category, risk characteristics, and mitigation strategy. The analysis focused on five major risk categories: deployment, people, strategy, framework, and quality. The findings indicate that Waterfall is frequently associated with people, strategy, framework, and quality-related risks, particularly because of its sequential structure and dependence on requirements defined at an early stage. Prototyping provides greater support for early requirement validation and stakeholder feedback but introduces risks related to quality, scope changes, and prototype management. Incremental development provides greater adaptability to changing requirements and enables early delivery, but introduces risks associated with module integration, documentation, coordination, and additional development effort. Therefore, no single development model can be considered universally superior. Instead, the appropriate model depends on project requirements, uncertainty, stakeholder involvement, delivery strategy, and the organization's ability to manage the associated risks. This study contributes a risk-oriented comparison of Waterfall, Prototyping, and Incremental development models and provides a structured basis for selecting a software development approach according to project-specific risk conditions.

Keywords : Risk Analysis; Software Development Life Cycle; Waterfall; Prototype; Incremental.

Article Info :

Submitted:

Reviewed:

Accepted :

Copyright © 2026 – Journal. All rights reserved.

1. INTRODUCTION

Recent studies have demonstrated that software risk management has been examined from different perspectives. Masso et al. conducted a systematic literature review involving 45 studies and identified risk-management activities throughout the software life cycle, while also highlighting limitations in the rigor of empirical validation and the use of standardized risk-management frameworks. More recently, Adriani et al. conducted an SLR on risk-management practices throughout the software development life cycle and mapped risk-management activities to traditional and agile development processes. Other recent research has focused specifically on risk-management tools and techniques used in software projects.

However, these studies primarily examine software risk management at the general software life-cycle level or focus on particular risk-management activities and techniques. They do not specifically provide a comparative synthesis of the risk profiles of Waterfall, Prototyping, and Incremental development using a common risk-classification framework. Existing studies also tend to examine individual development models or compare development models based on general characteristics such as flexibility, cost, development time, and requirements rather than systematically comparing the types of risks and corresponding mitigation strategies associated with each model.

This creates an important research gap. Waterfall, Prototyping, and Incremental development differ fundamentally in how they handle requirements, stakeholder feedback, development cycles, delivery, and change. These differences can influence the type and distribution of risks encountered throughout software development. Therefore, simply identifying the advantages and disadvantages of each model is insufficient for supporting risk-oriented model selection. A focused SLR is necessary to consolidate fragmented evidence and provide a consistent comparison of risk characteristics across the three models.

Accordingly, this study specifically focuses on comparing Waterfall, Prototyping, and Incremental development from a risk-management perspective. The review identifies the major risk categories reported in previous studies, compares their occurrence and characteristics across the three models, and examines the mitigation strategies associated with those risks. The resulting comparison is intended to provide evidence-based guidance for selecting a software development model according to project-specific risk conditions.

2. METHOD

This study employed a Systematic Literature Review (SLR) to identify, evaluate, classify, and synthesize existing research concerning risk management in Waterfall, Prototyping, and Incremental software development models. The review methodology was based on systematic review guidelines for software engineering, while PRISMA 2020 was adopted to improve the transparency and completeness of the study identification, selection, and reporting process. PRISMA 2020 provides updated guidance for reporting the identification, selection, appraisal, and synthesis of studies in systematic reviews (Page et al., 2021). The SLR process consisted of five main stages: (1) formulation of research questions, (2) literature search, (3) study selection, (4) quality assessment and data extraction, and (5) data synthesis and comparative analysis.

A. Research Question

The research questions were formulated based on the research gap identified in the Introduction. The questions were designed to investigate the risk characteristics and mitigation strategies associated with Waterfall, Prototyping, and Incremental development. RQ1: What types of risks are reported in Waterfall, Prototyping, and Incremental software development models?

1. RQ1: What types of risks are reported in Waterfall, Prototyping, and Incremental software development models?
2. RQ2: How do the risk characteristics and mitigation strategies differ among Waterfall, Prototyping, and Incremental development models?
3. RQ3: Under what project conditions does each development model provide a more appropriate risk-management profile?

These research questions guided the search strategy, study selection, data extraction, and comparative synthesis.

B. Search Strategy

The literature search was conducted using academic databases and indexing services, including Google Scholar, ScienceDirect, Semantic Scholar, and IEEE Xplore. The search focused on studies concerning software development models and software risk management.

The primary search string was formulated using Boolean operators as follows: ("software development" OR "software development model" OR "software development life cycle" OR SDLC) AND ("risk management" OR "risk analysis" OR "software risk") AND ("waterfall" OR "prototyping" OR "prototype" OR "incremental").

Additional search combinations were applied when necessary to identify studies that discussed one of the three target development models and its associated risk characteristics. The search was conducted using the title, abstract, and keyword fields where supported by the respective databases.

C. Study Selection

Following the recommendations offered by Kitchenham y Charters (Kitchenham et al., 2007), the selection of the papers was completed in three stages:

- a. **Phase 1:** Selection of Potential Studies Using Search Strings The first step in the process is to select potential studies by using search strings, locating each study, and assessing it based on the title, abstract, and keywords.
- b. **Phase 2:** Selection of Primary Studies from Among Candidate Papers collected in Phase 1 Performing a Full Text Analysis and Applying Inclusion and Exclusion Criteria This phase involves the selection of primary studies from among candidate papers collected in the previous phase.
- c. **Phase 3:** The application of quality assessment to the primary research identified in Phase 2 brings us to the third phase.

D. Inclusion and Exclusion Criteria

To ensure consistency during study selection, predefined inclusion and exclusion criteria were applied.

a. Inclusion Criteria

1. IC1: The study discusses software development or software development life-cycle models.
2. IC2: The study discusses Waterfall, Prototyping, or Incremental development.
3. IC3: The study contains information related to software development risks or risk management.
4. IC4: The full text of the study is available.
5. IC5: The study contains sufficient information for risk identification and classification.

b. Exclusion Criteria

1. EC1: Duplicate publications.
2. EC2: Studies unrelated to software development.
3. EC3: Studies that do not discuss software development risks or risk management.
4. EC4: Studies without sufficient information for data extraction.
5. EC5: Non-scholarly materials such as blogs, commercial websites, or informal web content.

E. Quality Assessment

The quality of each included study was assessed using five quality criteria adapted to the objectives of this review.

1. QC1: Does the study provide a clear description of the risk-management activity or activities?
2. QC2: Does the study identify the software development process or life-cycle stage to which the risk applies?
3. QC3: Is the proposed approach supported by empirical evidence such as a case study, survey, experiment, or other empirical investigation?
4. QC4: Does the study describe validity, reliability, limitations, or potential sources of bias?
5. QC5: Is the study published in a recognized academic journal or conference?

Each criterion was scored as 1 (Yes) or 0 (No). The resulting quality score was used to assess the relevance and strength of the evidence during the synthesis process. Quality assessment was not used as the sole basis for excluding a study; instead, the scores were used to interpret the strength of the evidence and identify potential limitations.

F. Data Source Process

The search stage as it relates to risk management is used when looking at these various software development approaches. The information gathered comes from secondary data, or information that was previously public. These were electronic (online) databases. Electronic databases are as follows:

- a. Google Scholar (<https://scholar.google.com/>)
- b. Elsevier (<https://www.sciencedirect.com/>)
- c. ResearchGate (<https://www.researchgate.net/>)
- d. Semantic Scholar (<https://www.semanticscholar.org/>)
- e. IEEExplore (<https://ieeexplore.ieee.org/>)

The fact that all that is needed to access an article on this website <https://sci-hub.se/> is the DOI for the piece enables it to be accessed quickly and easily. After that, we are able to continue the process of our study by downloading it or seeing it immediately on the internet.

G. Data Synthesis and Analysis

The extracted data were analyzed using qualitative thematic synthesis and comparative analysis. Risk findings were grouped into five categories identified from the reviewed studies:

1. **Deployment**
2. **People**
3. **Strategy**
4. **Framework**
5. **Quality**

The frequency of risk-category occurrences was calculated for each software development model. Because one study may report more than one risk category and may discuss more than one development model, the frequency values represent coded occurrences rather than unique study counts. The synthesis then compared the risk characteristics, causes, and mitigation strategies of Waterfall, Prototyping, and Incremental development. The comparison was used to identify the risk-management profile of each model and the project conditions under which a particular model may be more appropriate.

3. RESULT AND DISCUSSION

After conducting research on the literature obtained from online repositories, we used 16 articles that met the criteria, namely articles that discussed risk management in software development models. A list of titles, software development models, and risk management can be seen in Table 1. We focus on the waterfall, prototype, and incremental software development methods of study. This means the other software development models are not covered in our discussion and we indicate their existence with a (X).

Table 1 Mapping of Research Focus

No	Title	Development Models	Risk Management
1	Critical evaluation of waterfall project management methodology: A case study of digital management conference project (Aina et al., 2022)	Waterfall	People
2	Agile Software Requirements Engineering Challenges-Solutions - A Conceptual Framework from Systematic Literature Review (Hoy, Z., & Xu, M. 2023)	Agile	X
3	Implementation of Prototyping Process Model in Task Management Application and Objective Key Result Case Study PT.XYZ (Samihardjo et al., 2020)	Prototype	Quality
4	Formalization of the prediction and ranking of software development life cycle models (Almazaydeh et al., 2022)	Waterfall	Strategy
		Incremental, Iterative	Strategy
		Agile, Spiral, Scrum	X
5	Waterfall Process Operations in the Fast-paced World: Project Management Exploratory Analysis (Aroral, 2021)	Waterfall	Framework
6	Analisis Pengembangan Sistem Informasi Dengan Metode Waterfall Systematic Literature Review (Qolbi, R. M., Putra, R. M., & Nugroho, W. 2024)	Waterfall	People and Tools
7	Application of the Waterfall Method on a Web-Based Job Training Management Information System at Trunojoyo University Madura (Herawati et al., 2021)	Waterfall	Framework
8	Agile versus Waterfall: Project Management: Decision Model for Selecting the Appropriate Approach to a Project (Thesing et al., 2021)	Waterfall	People and Strategy
		Agile	X
9	An empirically based model of software prototyping: A mapping study and a multi-case study (Bjarnason et al., 2023)	Prototype	Quality
10		Waterfall	Strategy

	A Survey of Software Development Process Models in Software Engineering (Sarker et al, 2015)	Incremental	Strategy
		V-shaped, RAD, Iterative Spiral, Agile	X
11	A Review of Risk Management in Different Software Development Methodologies (Evans, 2020)	Waterfall	Deployment, People, Quality
		Iterative and incremental	Deployment
		V-model, Spiral, Scrum.	X
12	Software Development Life Cycle Models-Comparison, Consequences (Rastogi, 2014)	Waterfall	Deployment and Quality
		Iterative, V-shaped, Spiral, Agile	X
13	The incremental model in software development: A structured and interactive way to deliver quality products (Lima et al. 2023)	Incremental	Quality
14	A Survey of Software Development Process Models in Software Engineering (Jovliyevich, 2022)	Waterfall	Strategy, Framework and Quality
		Incremental	Strategy
		V-shaped, RAD, Iterative Spiral and Agile	X
15	A Comparative Study on Software Development Life Cycle Models (Salve et al., 2018)	Waterfall	Strategy and Quality
		Prototype	People, Quality and Strategy
		Incremental	Strategy and Quality
		Spiral	X
16	Analisis Resiko Pada Pengembangan Perangkat Lunak yang Menggunakan Metode Waterfall dan Prototyping (Supandi & Sudir, 2019)	Waterfall	People, Framework, and Quality
		Prototype	People, Framework, and Quality

Figure 1 shows the articles Risk Management targeted based on the focus of the model to be researched, namely waterfall, prototype and incremental. There are 16 articles, 12 of which use the waterfall software development methodology, four of which use the prototype model, and six of which use the incremental model.

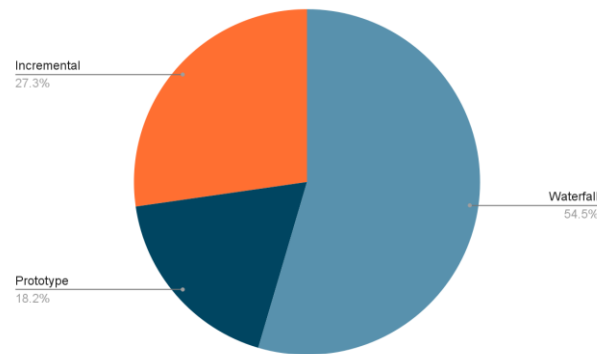


Fig. 1 Prototype-Oriented Software Life Cycle.

A. Comparison of Exact Conditions of Application

The individual aspects of the software development project have a significant role in determining which software development methodologies, such as waterfall, prototype, or incremental, should be used. The following gives a more in-depth description of when each approach should be used as well as the reasons why it should be used (Adriani et al., 2024; Luthfiansyah et al., 2024; Page et al., 2021; Masso et al., 2020).

Table 2 Comparison of Exact Conditions of Application

No	Waterfall Models	Prototype Models	Incremental Models
1	Suitable for converting existing products to new platforms	Effective in the analysis and design of online systems	Suitable for products that are released multiple times based on their functionality
2	Suitable for projects that have clear requirements and don't change much during the development process	Great for projects where requirements are unclear or not fully understood at first	Suitable for complex and large projects that require sufficient time and resources for development
3	When have a lot of cost	When want to save money	When have a lot of costs, because need to repeat product releases
4	Systems that have a lot of user interaction	A system with little user interaction	Suitable for applications that have a lot of interaction or high complexity
5	Stable product definition and requirements	Suitable for finely designing human-computer interfaces that are engineering the needs of repetitive users	Key requirements must be specified, however, some details may evolve over time
6	Suitable for projects with well-structured and well-organized development teams that require complete documentation of each stage	Suitable for products that require user validation and feedback from the very beginning of development	Suitable for projects that require feedback and validation from users during development

B. Comparison of Risk Management

This section will provide an overview of three different techniques in various software development, along with a tabular representation of the risk characteristics exhibited by each methodology which is neatly packaged in Table 3 (Adriani et al., 2024; Luthfiansyah et al., 2024; Meckenstock et al., 2024; Azanedo & Dávila, 2025; Nwekori, 2026).

Table 3 Risk Analysis Results

Models	Risk Management	
	Risk analysis	Solution Strategy
Waterfall	It's hard to change mistakes	Identification of software requirements is determined and ensured from the early stages of planning
	Development can not be done repeatedly before the occurrence of a before step	Good management is required before the development process
	Clients find it difficult to express their needs explicitly (clearly, decisively, frankly) so that they cannot accommodate (provide something to meet their needs) uncertainty at the beginning of development.	It is necessary to carry out an evaluation at each stage and ensure that it is clear before entering the next stage, this is to minimize the occurrence of errors
	Client are impatient and change specifications so that the project can be completed quickly	Do not apply to software development rush
	Lack of Client involvement so that the resulting product is not fully in accordance with the business strategy desired by the client	Client are involved in the planning process, namely in determining the identification of software requirements
	The possibility of repeating the stages from the beginning is high because the client does not know the desired product clearly and often changes specifications	In order not to have to repeat the steps from the beginning, complete documentation is needed.
Prototype	The rapid pace of development can create quality problems, which are not discovered until the system is operational.	Conduct training and skills development of the development team in terms of the latest technology and better software development methodologies, so as to increase the ability and expertise in developing quality software
	Increases overall system complexity	Clear segregation of tasks between work groups can help reduce complexity and improve development efficiency
	Experienced HR leaves the project before the project time is finished. The new HR could not stand the pressure of the project so they left the project before the project was finished.	MOU is needed
Incremental	Lack of sufficient documentation	Ensuring that the documentation created is always updated at each stage
	Difficulty in integrating modules, Because each module is developed separately, there can be difficulties in integrating all modules into one unified system	Perform testing on each module and carry out the integration in stages

Additional costs and time	Estimate costs and time accurately at each stage and conduct periodic evaluations to ensure the project is progressing according to plan
There is a developer staff who left	There is a need for daily reports/documentation, so that if a developer leaves, the software development process can continue without starting from scratch

Table 3 above shows that the choice of software development methods such as waterfall, prototype, or incremental is highly dependent on the specific characteristics of the software development project. In practice, software development methods can be used flexibly and often depend on the specific characteristics of the software development project in question. In addition, some software development teams may choose to combine various software development methods, such as combining waterfall and incremental or prototyping and incremental methods to achieve better development goals. However, the application of this methodology depends on the aspect of your software project whether your software is large or not. It is very difficult to choose the right methodology for developing software that fits the project, but using tables 2 and 3 is very helpful in choosing the right methodology by looking at the risks that will be faced in the future and avoiding risks before they arise. Therefore, the selection of a software development model should consider the specific risk characteristics identified in the project. Because risk management has a relationship between choosing the right personnel to analyze risks in software projects.

Table 4 Risk Focus Result

Risk Focus	Models		
	Waterfall	Prototype	Incremental
Deployment	2	0	1
People	5	2	0
Strategy	5	1	4
Framework	4	1	0
Quality	5	4	2

Table 4 presents the distribution of reported risk categories based on the studies mapped in Table 1. Waterfall appears in 12 studies, Prototype in 4 studies, and Incremental in 6 studies. The risk-category frequencies indicate that Waterfall has the broadest distribution of reported risk characteristics, while Prototype is more concentrated on quality-related risks and Incremental is more concentrated on strategy-related risks. Visualization of the results of table 4 will be presented in figure 2.

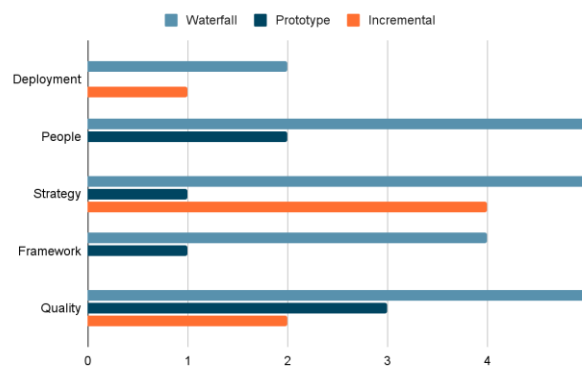


Fig. 2 Risk Focus Result

Figure 2 shows the results of a model's analysis of the Risk Management categories. Deployment risk categories for each waterfall, prototype and incremental software development model are 2, 0, and 1. People risk categories are 5, 2, and 0. Strategy risk categories are 5, 1, and 4. Framework risk categories are total 4, 1, and 0. and the quality risk categories are 5, 4, and 2.

The comparison indicates that the selection of Waterfall, Prototype, or Incremental development should consider the specific risk characteristics of the project. Waterfall presents a broader distribution of reported risks, Prototype is more concentrated on quality and people-related risks, and Incremental is primarily associated with strategy and quality-related risks. Therefore, development-model selection should be considered as a risk-management decision rather than solely a technical or process decision.

4. CONCLUSION

This systematic literature review examined the risk-management characteristics of Waterfall, Prototyping, and Incremental software development models based on 16 primary studies. The review identified five major risk categories: deployment, people, strategy, framework, and quality.

In response to RQ1, the findings show that the three development models exhibit different risk profiles. Waterfall is frequently associated with people, strategy, framework, and quality-related risks. Prototyping is strongly associated with quality and stakeholder-related risks, while Incremental development is particularly associated with strategy, integration, documentation, and coordination-related challenges.

In response to RQ2, the findings demonstrate that the differences in risk characteristics are closely related to the structural characteristics of each development model. Waterfall relies on early planning and relatively stable requirements, making it more sensitive to requirement and planning errors. Prototyping reduces requirement uncertainty through early stakeholder feedback but introduces risks related to prototype quality, scope changes, and stakeholder expectations. Incremental development enables progressive delivery and adaptation to changing requirements but requires effective integration, documentation, and coordination across development increments.

In response to RQ3, no single development model can be considered universally superior for risk management. Waterfall provides a more appropriate risk-management profile for projects with stable and well-defined requirements. Prototyping is more appropriate when requirements are uncertain and early stakeholder validation is required. Incremental development is more appropriate when requirements evolve and functionality needs to be delivered progressively.

The main contribution of this study is a risk-oriented comparative framework for understanding the relationship between software development models, risk characteristics, and mitigation strategies. The framework can support project managers in selecting a development approach according to the uncertainty, requirements, stakeholder involvement, and delivery conditions of a particular project. Despite these contributions, the study has limitations. The evidence base consists of a relatively small number of primary studies, and the distribution of studies across the three development models is not balanced. In addition, the synthesis is qualitative and based on the risk categories reported in the selected literature. Future research should therefore expand the search to additional academic databases, include a larger number of recent empirical studies, and investigate quantitative or mixed-method approaches for comparing risk severity and mitigation effectiveness across development models.

ACKNOWLEDGEMENT

The authors would like to express their gratitude to God Almighty for His blessings and grace, which have enabled the successful completion of this scientific article. The authors would also like to extend their sincere appreciation to colleagues at Sekolah Tinggi Ilmu Ekonomi (STIE) IBMT Surabaya for their valuable time and collaboration throughout this research. The authors also express their gratitude to Politeknik Pariwisata Batam and Institut Teknologi Batam for providing academic support, a conducive scholarly environment, and opportunities to develop the authors' competencies and research experience.

REFERENCES

- [1] Adriani, Z. A., Raharjo, T., & Trisnawaty, N. W. (2024). A Comprehensive Examination of Risk Management Practices Throughout the Software Development Life Cycle (SDLC): A Systematic Literature Review. *The Indonesian Journal of Computer Science*, 13(3). DOI: 10.33022/ijcs.v13i3.401.
- [2] Aina, T. S., Akinte, O. O., Awelewa, A. J., & Adelakun, D. O. (2022). Critical evaluation of waterfall project management methodology: A case study of digital management conference project. *International Journal of Advanced Multidisciplinary Research and Studies*. <https://www.researchgate.net/publication/361092440>
- [3] Almazaydeh, L., Alsafasfeh, M., Alsalamdeen, R., & Alsharari, S. (2022). Formalization of the prediction and ranking of software development life cycle models. *International Journal of Electrical and Computer Engineering*, 12(1), 534–540. <https://doi.org/10.11591/ijece.v12i1.pp534-540>
- [4] Andryadi, A. A., Zulkifli, R., Amrullah, F., Puspita, N. N., Pratama, F. P., Amalia, D. A., Aulya, S., Agustina, N. S., & Nuraeni, M. (2026). Identifikasi Best Practice Pengembangan Sistem Informasi Melalui Perbandingan Model SDLC: Tinjauan Literatur. *Jurnal Nasional Komputasi dan Teknologi Informasi*, 9(1).
- [5] Aroral, H. K. (2021). Waterfall Process Operations in the Fast-paced World: Project Management Exploratory Analysis. *International Journal of Applied Business and Management Studies*, 6(1).
- [6] Azanedo, D., & Dávila, A. (2025). Risk Management in Software Development Using Scrum: A Systematic Literature Review. *IEEE*.
- [7] Camburn, B., Viswanathan, V., Linsey, J., Anderson, D., Jensen, D., Crawford, R., Otto, K., & Wood, K. (2017). Design prototyping methods: State of the art in strategies, techniques, and guidelines. *Design Science*, 3. <https://doi.org/10.1017/dsj.2017.10>
- [8] Dhiman, K. L. (2024). Reassessment of Software Development Life Cycle Models. *International Journal of Research in Engineering, Science and Management*, 7(2), 4–10.
- [9] Gowtham, V., Manoj, Y., Pooventhiran, G., Praveen, A., Shivaram, R., & Kathiresan, A. (2017). Evolutionary Models in Software Engineering. *International Journal of New Technology and Research (IJNTR)*, 3(5), 30–33.
- [10] Haniva, D. T., Ramadhan, J. A., & Suharso, A. (2023). Systematic Literature Review Penggunaan Metodologi Pengembangan Sistem Informasi Waterfall, Agile, dan Hybrid. *Journal of Information Engineering and Educational Technology*.
- [11] Herawati, S., Negara, Y. D. P., Febriansyah, H. F., & Fatah, D. A. (2021). Application of the Waterfall Method on a Web-Based Job Training Management Information System at Trunojoyo University Madura. *E3S Web of Conferences*, 328. <https://doi.org/10.1051/e3sconf/202132804026>
- [12] Jason Henri, E., & Henri, J. (2020a). A Review of Risk Management in Different Software Development Methodologies. <https://www.researchgate.net/publication/343584508>

- [13] Jovliyevich, K. B. (2022). A Survey of Software Development Process Models in Software Engineering. *Eurasian Scientific Herald*, 8, 69–72.
- [14] Kitchenham. (2007). *Guidelines for Performing Systematic Literature Reviews in Software Engineering*.
- [15] Kramer, M. (2018). *Best Practices in Systems Development Lifecycle: An Analyses Based on the Waterfall Model*. <https://www.researchgate.net/publication/328770195>
- [16] Luthfiansyah, F., Prasetyo, A., & Raharjo, T. (2024). A Systematic Review of Risk Management Tools and Techniques in Software Projects. *The Indonesian Journal of Computer Science*, 13(1). DOI: 10.33022/ijcs.v13i1.3694
- [17] Madhukar Salve, S., Neha Samreen, S., Khatri-Valmik, N., & Professor, A. (2018). A Comparative Study on Software Development Life Cycle Models. *International Research Journal of Engineering and Technology*, 5(2), 696–700. www.irjet.net
- [18] Martins, H. F., de Oliveira, A. C., Canedo, E. D., Kosloski, R. A. D., Paldês, R. Á., & Oliveira, E. C. (2019). Design thinking: Challenges for software requirements elicitation. *Information*, 10(12). <https://doi.org/10.3390/info10120371>
- [19] Meckenstock, J.-N., et al. (2024). Shedding light on the dark side – A systematic literature review of the issues in agile software development methodology use. *Journal of Systems and Software*, 211, 111966. DOI: 10.1016/j.jss.2024.111966
- [20] Nwekori, F. E. (2026). Comparative Analysis of Agile and Traditional Software Development Methodologies: A Systematic Review of Project Success, Flexibility, and Organizational Performance.
- [21] Page, M. J., McKenzie, J. E., Bossuyt, P. M., et al. (2021). The PRISMA 2020 statement: An updated guideline for reporting systematic reviews. *BMJ*, 372, n71. <https://doi.org/10.1136/bmj.n71>
- [22] Pressman, R. S. (2010). *Software Engineering: A Practitioner's Approach*. 7th Edition. McGraw Hill, New York.
- [23] Qolbi, R. M., Putra, R. M., & Nugroho, W. (2024). Analisis Pengembangan Sistem Informasi Dengan Metode Waterfall Systematic Literature Review. *Infoman's: Jurnal Ilmu-ilmu Informatika dan Manajemen*.
- [24] Rastogi, V., & Professor, A. (2015). Software Development Life Cycle Models-Comparison, Consequences. *International Journal of Computer Science and Information Technologies*, 6(1).
- [25] Salve, S. M., Samreen, S. N., & Khatri-Valmik, N. (2018). A Comparative Study on Software Development Life Cycle Models. *International Research Journal of Engineering and Technology (IRJET)*, 5(2), 696–700.
- [26] Samihardjo, R., Anggadita, Y., & Nugraha, E. (2020). Implementation of Prototyping Process Model in Task Management Application and Objective Key Result Case Study PT. XYZ-Palarch's. *Journal of Archaeology of Egypt/Egyptology*, 17(5).
- [27] Sarker, I. H., Faruque, F., Hossen, U., & Rahman, A. (2015). A survey of software development process models in software engineering. *International Journal of Software Engineering and Its Applications*, 9(11), 55–70. <https://doi.org/10.14257/ijseia.2015.9.11.05>
- [28] Sitepu, A. I. B., & Tanjung, D. Y. H. (2020). Rancang Bangun Aplikasi Pemesanan dan Penjualan Berbasis Web dan Android pada Toko YT. Wall Interior. *Jurnal Mahasiswa Fakultas Teknik dan Ilmu Komputer*, 1(1), 816–828.
- [29] Sommerville, I. (2011). *Software Engineering*. Pearson.
- [30] Supandi, F., & Sudir, M. (2019). Analisis Resiko Pada Pengembangan Perangkat Lunak Yang Menggunakan Metode Waterfall dan Prototyping. *Seri Prosiding Seminar Nasional Dinamika Informatika*, 2(1).

-
- [31] Thakur, A., Singh, D., & Maurya, H. C. (2013). A Survey on Incremental Software Development Life Cycle Model. *International Journal of Engineering Technology and Computer Research (IJETCR)*, 3(2), 102–106.
- [32] Thesing, T., Feldmann, C., & Burchardt, M. (2021). Agile versus Waterfall Project Management: Decision model for selecting the appropriate approach to a project. *Procedia Computer Science*, 181, 746–756. <https://doi.org/10.1016/j.procs.2021.01.227>
- [33] Tilley, S., & Rosenblatt, H. (2017). *Systems Analysis and Design*. 11th Edition. Cengage Learning Asia Pte Ltd.
- [34] Wahyuni, et al. (2025). Analisis Komparatif Metode Spiral dan Incremental Berdasarkan Manajemen Resiko dan Fokus Utama Pengembangan. *JEKIN – Jurnal Teknik Informatika*, 5(2). DOI: 10.58794/jekin.v5i2.1520
- [35] Yas, Q. M., ALazzawi, A., & Rahmatullah, B. (2023). A Comprehensive Review of Software Development Life Cycle Methodologies: Pros, Cons, and Future Directions. *Iraqi Journal for Computer Science and Mathematics*, 4(4), Article 14.