



CS5709 - SOFTWARE ENGINEERING EVOLUTION  
Student: SHUBHAM DIGAMBER BIRADAR  
Student ID: 25322206

## Department of SOFTWARE ENGINEERING

### Module: CS5709 - SOFTWARE ENGINEERING EVOLUTION 2025/6 SEM1

#### **Assignment 5: Project Peer Review**

Student: SHUBHAM DIGAMBER BIRADAR

Student ID: 25322206

E-MAIL : 25322206@studentmail.ul.ie

## Table of Contents

1. Abstract.....	3
2. Introduction .....	4
3. Research Topic: Documentation for Software Evolution....	5
3.1 Software Documentation and Software Evolution .....	5
3.2 Documentation Quality and Maintainability .....	6
3.4 Tools and Technologies for Software Documentation ....	7
4. Application of Research Findings.....	7
5. Conclusion.....	8
6. References .....	9

## 1. Abstract

This peer review critically evaluates a digital portfolio website developed and deployed as part of the CS5709 Software Engineering Evolution module. The project investigates the role of software documentation in supporting software evolution, maintainability, and long-term adaptability. Beyond the core academic requirements, the project demonstrates added professional value through the acquisition of a custom domain and public deployment, reflecting real-world engineering practices.

The review combines independent literature research with a technical walkthrough of the portfolio system. The research focuses on software documentation quality, documentation for software evolution, and documentation practices in modern DevOps-driven environments. Academic sources highlight that documentation is not a static artefact but an evolving component that supports architectural understanding, onboarding, maintenance, and iterative refinement.

The findings are applied directly to the digital portfolio project, examining how documentation decisions influenced modularity, refactoring, CI/CD integration, and long-term maintainability. Particular attention is paid to code-level documentation, architectural clarity, and traceability between design decisions and implementation. The review also reflects on peer feedback and iterative improvements made throughout the module.

Overall, this review concludes that high-quality, continuously evolving documentation significantly enhances software sustainability. The portfolio project demonstrates a clear progression from minimal documentation to a more disciplined, professional approach aligned with software evolution principles. The outcomes reinforce the importance of documentation as a first-class engineering activity rather than an afterthought.

## 2. Introduction

Software systems rarely remain static after initial development. As requirements change, technologies evolve, and user expectations grow, software must continuously adapt. This principle of software evolution forms the foundation of the CS5709 module and directly motivated the development of the digital portfolio website.

The portfolio project was designed not only as a personal showcase but also as an evolving software artefact that undergoes iterative refinement. Built using modern web technologies such as React, Tailwind CSS, Firebase, and CI/CD pipelines, the project reflects contemporary software engineering practices. Additionally, acquiring and hosting the website under a personal domain extends the project beyond an academic exercise into a publicly accessible, real-world system.

The primary research motivation for this review is to examine how software documentation supports software evolution. While documentation is often perceived as secondary to code, research

consistently shows that inadequate documentation leads to increased technical debt, reduced maintainability, and higher onboarding costs.

The research method adopted in this review is a literature-based study combined with technical analysis and peer review. Academic sources on documentation quality, evolutionary software development, and DevOps documentation practices are synthesised and applied to the portfolio project. This approach enables a structured evaluation of how research findings translate into practical software engineering decisions.

### 3. Research Topic: Documentation for Software Evolution

#### 3.1 Software Documentation and Software Evolution

Software documentation plays a critical role in enabling systems to evolve over time. As systems grow in complexity, documentation acts as the collective memory of architectural decisions, design rationale, and system behaviour. Without proper documentation, evolutionary changes become risky and error-prone.

Research indicates that evolving systems require documentation that adapts alongside the codebase. Static documents quickly become outdated, leading to misinformation and reduced trust in documentation artefacts. Therefore, modern software evolution emphasises living documentation that evolves with each iteration.

## 3.2 Documentation Quality and Maintainability

Documentation quality directly impacts software maintainability. High-quality documentation is accurate, concise, structured, and aligned with the current system state. Studies show that poor documentation increases maintenance effort and slows down refactoring activities.

Key quality attributes include clarity, traceability, consistency, and accessibility. Code-level documentation (comments, JSDoc, README files) complements higher-level architectural documents by explaining intent rather than implementation details alone.

## 3.3 Documentation in Agile and DevOps Environments

Agile and DevOps practices often challenge traditional documentation approaches due to rapid release cycles. However, research highlights that lightweight but continuous documentation is essential in these environments.

Practices such as README-driven development, automated documentation generation, and CI/CD documentation pipelines ensure that documentation remains synchronised with evolving systems. This approach reduces friction between speed and maintainability.

### 3.4 Tools and Technologies for Software Documentation

Modern documentation tools support automation and integration with development workflows. Examples include Markdown-based documentation, JSDoc, static site generators, and documentation pipelines integrated with GitHub Actions.

These tools align documentation updates with code commits, improving traceability and reducing documentation decay. Research emphasises that tooling is a key enabler of sustainable documentation practices in evolving systems.

## 4. Application of Research Findings

The research findings were directly applied to the digital portfolio project. Initially, documentation was minimal and primarily descriptive. As the project evolved, several improvements were introduced based on best practices identified in the literature.

Architectural clarity improved through better folder organisation and separation of concerns. Code-level documentation was enhanced using structured comments and clearer naming conventions. The README was expanded to document the technology stack, deployment process, and CI/CD workflow.

The acquisition of a custom domain and public deployment reinforced the importance of accurate documentation, as the project became a real-

world system rather than a closed academic artefact. Iterative refinement, informed by peer feedback, further improved traceability between design decisions and implementation.

Overall, applying research-driven documentation principles improved maintainability, readability, and long-term sustainability of the portfolio system.

## 5. Conclusion

This peer review demonstrates that software documentation is a fundamental enabler of software evolution rather than a supplementary activity. Through independent research and practical application, the review highlights how evolving documentation supports maintainability, architectural clarity, and iterative refinement.

The digital portfolio project illustrates a clear progression toward professional software engineering practice. By integrating research insights into documentation and development workflows, the project aligns strongly with the learning objectives of CS5709 and prepares the system for continued evolution beyond the module.



## 6. References

Bass, L., Weber, I., & Zhu, L. (2015). DevOps: A software architect's perspective. Addison-Wesley Professional.

Bennett, K. H., & Rajlich, V. T. (2000). Software maintenance and evolution: A roadmap. Proceedings of the Conference on the Future of Software Engineering, 73–87. <https://doi.org/10.1145/336512.336534>

Brown, N., Cai, Y., Guo, Y., Kazman, R., Kim, M., Kruchten, P., Lim, E., MacCormack, A., Nord, R., Ozkaya, I., Sangwan, R., & Seaman, C. (2010). Managing technical debt in software-reliant systems. Proceedings of the FSE/SDP Workshop on Future of Software Engineering Research, 47–52. <https://doi.org/10.1145/1882362.1882373>

Forward, A., & Lethbridge, T. C. (2002). The relevance of software documentation, tools, and technologies: A survey. Proceedings of the 2002 ACM Symposium on Document Engineering, 26–33. <https://doi.org/10.1145/585058.585065>

Fowler, M. (2019). Refactoring: Improving the design of existing code (2nd ed.). Addison-Wesley.

Humble, J., & Farley, D. (2010). Continuous delivery: Reliable software releases through build, test, and deployment automation. Addison-Wesley Professional.

IEEE Computer Society. (2018). ISO/IEC/IEEE 12207:2017 systems and software engineering—software life cycle processes. IEEE.

Kitchenham, B., Pfleeger, S. L., Pickard, L., Jones, P. W., Hoaglin, D. C., Emam, K. E., & Rosenberg, J. (2002). Preliminary guidelines for empirical research in software engineering. *IEEE Transactions on Software Engineering*, 28(8), 721–734.  
<https://doi.org/10.1109/TSE.2002.1027796>

Kruchten, P. (1995). The 4+1 view model of architecture. *IEEE Software*, 12(6), 42–50. <https://doi.org/10.1109/52.469759>

McConnell, S. (2004). *Code complete* (2nd ed.). Microsoft Press.

Pérez, J., Díaz, J., & Gutiérrez, F. (2014). Documentation in agile software development: A systematic literature review. *Information and Software Technology*, 56(7), 722–734.  
<https://doi.org/10.1016/j.infsof.2014.02.002>

Spinellis, D. (2006). Code quality: The open source perspective. Addison-Wesley Professional.

Storey, M. A., Zagalsky, A., Filho, F., Singer, L., & German, D. M. (2017). How social and communication channels shape and challenge a participatory culture in software development. IEEE Transactions on Software Engineering, 43(2), 185–204.  
<https://doi.org/10.1109/TSE.2016.2584053>

Sommerville, I. (2016). Software engineering (10th ed.). Pearson Education.