Research on Code Documentation Tools for Software Projects

Course: CSE327 – Software Engineering
Project: University E-Voting System
Submitted By: Abu Ahmad, 2121725042, Section 8

Abstract

Code documentation is an essential part of software engineering that ensures a program is understandable, maintainable, and reusable. This research explores popular code documentation tools Sphinx, MkDocs, and pydoc used in Python-based projects. The objective is to evaluate their usefulness, strengths, and limitations and identify the most suitable tool for documenting the University E-Voting System project. The paper concludes that MkDocs offers the best balance between simplicity, presentation, and integration with GitHub, making it the recommended tool for this project.

---

1. Introduction

In modern software development, documenting code is as important as writing the code itself. Documentation helps new developers understand the system, supports future maintenance, and demonstrates professionalism. As part of the CSE327 project *University E-Voting System*, this paper investigates various tools that automate or assist in code documentation, focusing on Python-based frameworks.

---

2. Importance of Code Documentation

Code documentation provides insight into the logic and purpose of a software system. It benefits both developers and end-users by:

- Making the code easier to understand and debug
- Supporting collaboration in team projects
- Enabling faster onboarding of new developers
- Providing formal records of system features and APIs
- Increasing the maintainability and reusability of software

- Without proper documentation, even well-written programs become difficult to extend or maintain.

---

3. Types of Code Documentation Tools

There are generally two categories of documentation tools:

1. **Automatic Documentation Tools:**
   These tools extract information from code comments or docstrings and generate HTML or PDF documents automatically (e.g., Sphinx, pydoc, Doxygen).

2. **Manual or Hybrid Documentation Tools:**
   These allow developers to combine written documentation (Markdown, reStructuredText) with code references (e.g., MkDocs, GitHub Wiki).

---

4. Popular Code Documentation Tools

4.1 Sphinx

Sphinx is an advanced, Python-based documentation generator originally created for the Python language itself. It uses **reStructuredText (.rst)** files and can automatically generate documentation from code docstrings.
Advantages:

- Highly customizable themes
- Auto-generation from Python modules (`autodoc`)
- Supports multiple output formats (HTML, PDF, ePub)
  Disadvantages:
- Steep learning curve
- `.rst` syntax is harder than Markdown

4.2 MkDocs

MkDocs is a fast, user-friendly documentation generator that uses **Markdown (.md)** files. It is ideal for project documentation and integrates easily with **GitHub Pages**.
Advantages:

- Simple syntax (Markdown)
- Beautiful “Material” theme
- Real-time preview and easy deployment

Disadvantages:

- Not as advanced for auto code documentation as Sphinx

4.3 pydoc

pydoc is a **built-in Python tool** that extracts documentation directly from code docstrings. It can generate both plain-text and HTML documentation.

Advantages:

- No installation needed (comes with Python)
- Extremely simple to use (`pydoc -w module_name`)

Disadvantages:

- Very basic formatting
- Not suitable for large-scale or multi-module projects

4.4 Doxygen

Doxygen is a cross-language documentation tool supporting **C, C++, Java, and Python**. It extracts documentation from specially formatted comments and can create detailed diagrams.

Advantages:

- Good for multi-language projects
- Generates UML-style diagrams

Disadvantages:

- Requires complex configuration
- Outdated visual style compared to MkDocs

---

5. Comparative Evaluation

Feature	Sphinx	MkDocs	pydoc	Doxygen
Language Support	Python	Any (Markdown-based)	Python	Multi-language
Ease of Use	Medium	Easy	Very Easy	Complex
Output Format	HTML, PDF, ePub	HTML	Text, HTML	HTML, PDF
Integration with GitHub	Yes	Yes	Limited	Yes
Code Auto-Documentation	Excellent	Limited	Basic	Good
UI/Design Quality	Good	Excellent	Basic	Average
Suitable for CSE327 Project	✅	✅✅✅	✅	⚠️

---

6. Recommended Tool for University E-Voting System

After analyzing all tools, **MkDocs** is identified as the most appropriate documentation solution for the University E-Voting System project.
Reasons include:

- It supports Markdown, which is easy to write and integrates naturally with GitHub Wiki.
- It produces visually appealing, professional documentation websites using the “Material” theme.
- It requires minimal setup, allowing students to focus on development rather than configuration.
- It aligns perfectly with the CSE327 assessment criteria, which emphasize code documentation, GitHub usage, and tool integration.

---

7. Conclusion

Code documentation tools play a crucial role in maintaining high-quality software. Tools such as Sphinx, MkDocs, and pydoc enhance developer productivity and ensure project maintainability. For small to medium academic projects like the University E-Voting System, **MkDocs** offers the best balance of simplicity, presentation, and functionality. Its integration

with GitHub Pages and Markdown-based workflow makes it the ideal tool for students and educational projects.

---

8. References

1. Sphinx Documentation – https://www.sphinx-doc.org
2. MkDocs Documentation – https://www.mkdocs.org
3. pydoc Documentation – https://docs.python.org/3/library/pydoc.html
4. Doxygen Official Site – https://www.doxygen.nl
5. Real Python – “Documenting Python Projects” (realpython.com)