

Final Project 1

Code: MGM 496-PMG496

Project Template

Faculty of Management
Midocean University

Midocean University

Faculty of Management

Department/ Specialization



Project title
Bug Tracking System

Student name

.....

Graduation Project – 1
Faculty of Management - Midocean University

Supervisor

Dr.

Faculty member- Faculty of management- Midocean University

Date of submission – number of Semester

Table of Contents

1. Chapter 1: Introduction	3
1.1. Research Background.....	4
1.2. Research Problem.....	6
1.2.1. Theoretical Gaps	6
1.2.2. Practical Gaps	7
1.3. Research Significance	8
1.3.1. At the theoretical level.....	8
1.3.2. At the practical level	9
1.4. Research Questions	Error! Bookmark not defined.
1.5. Research Objectives	Error! Bookmark not defined.
1.6. Research Hypotheses.....	Error! Bookmark not defined.
1.7. Research Model.....	Error! Bookmark not defined.
1.8. Research Methodology.....	Error! Bookmark not defined.
1.9. Scope of The Study	Error! Bookmark not defined.
1.10. The Suggested Project Structure.....	Error! Bookmark not defined.
2. Chapter 2: Literature Review	Error! Bookmark not defined.
2.1. Introduction	Error! Bookmark not defined.
2.2. The independent variable	Error! Bookmark not defined.
2.3. The dependent variable	Error! Bookmark not defined.
2.4. The relationship between the independent and the dependent variables	Error! Bookmark not defined.
2.5. Conclusion.....	Error! Bookmark not defined.
References.....	10

1. Chapter 1: Introduction

1.1. Research Background

- The bug tracking application carries great theoretical significance, as it embodies software engineering concepts such as requirements management, traceability, documentation, and impact analysis, enhancing the quality of development and deepening the academic understanding of the software lifecycle.
- With the increasing reliance of organizations on software, the development of their software products and their delivery to customers has become an indispensable priority, so these organizations rely on software project management as a basic practice, through which software development, starting from the initial preparation of requirements to the final testing and maintenance, is implemented over a predetermined period to produce the agreed software, and in the following lines, we explain the definition of software project management, its activities, and the most important challenges it faces.
- Software projects can be described as the activities of designing and organizing software development according to several factors, most notably time, cost, quality, and scale, which is a subcategory of the software development process.
- Thus, software project management is defined as the processes and control practices for planning, scheduling, and supervising software projects, as it works to allocate resources to develop computer programs that meet the predetermined requirements, and in this program, it is necessary to clarify the length, duration, and cost of the project for the client and developers.
- In a software project management system, project managers, stakeholders and users monitor costs, manage budget and quality, and can use this system as a management system.

- Software project management is the process of planning, organizing, and supervising software development projects to ensure their success. It aims to deliver the project on time, within the specified budget, and with the required quality, while controlling the scope of the project and risks. It includes key activities such as scoping and resource allocation, progress tracking, and communication and risk management.

Main activities of software project management:

- **Planning and scheduling:** Develop a detailed work plan and timeline for project implementation.
- **Scoping:** Accurately define project boundaries and requirements to avoid unwanted changes.
- **Resource Management:** Effectively allocating human, physical, and financial resources.
- **Cost and budget management:** Monitor costs and ensure that the project stays within budget.
- **Quality Management:** Ensuring that project deliverables meet the set quality standards.
- **Risk management:** Identifying potential risks and developing plans to deal with them.
- **Progress tracking and reporting:** Monitor workflow, track obstacles, and periodically report on project progress.
- **Communication and Collaboration:** Facilitate effective communication between team members and stakeholders.
- **Document Management:** Organize and document all documents related to the project.

1.2. Research Problem

- Difficulty in accurately and effectively managing and tracking bugs in a dynamic, multi-team development environment, resulting in processing delays, repeat errors, and low quality of the final product.
- "Software developers suffer from the absence of a standardized conceptual bug tracking model that connects requirements, code, and test cases, leading to poor impact analysis and difficulty in quality management."
- "Current systems lack a flexible theoretical framework that allows for the classification and prioritization of bugs in a way that is adaptable to modern development methodologies."

1.2.1. Theoretical Gaps

The theoretical problem of the bug tracking application project lies in designing a system that is able to accurately represent software bugs and link them to the various components of the system, while maintaining scalability, traceability, and analysis without complexity or ambiguity.

- **Structural representation of the bug** Specifying the characteristics that each bug record should contain (e.g., status, priority, source, impact, date) in a way that allows for analysis without excessive complexity.
- **Linking the bug to the components of the system** such as requirements, code, and test cases, which requires a Traceability Model that illustrates the relationships between the elements.
- **Classification and Prioritization of Ills** Developing a theoretical framework for classifying bugs according to impact, frequency, or source, which helps in making effective treatment decisions.
- **Bug Lifecycle Management** Design a model that shows the bug stages from detection to closure, specifying actions at each stage.

1.2.2. Practical Gaps

The practical problem of the bug tracking application project is that it is difficult to ensure documentation accuracy, traceability, and usability within a fast-paced and changing development environment, which affects product quality and speed of response to errors.

- Inaccurate or incomplete data input Developers may neglect to adequately document the bug, making it difficult to reproduce or analyze it later.
- Poor adherence to the use of the system Some teams do not commit to updating or recording the status of the bug, which leads to its loss of tracking or recurrence.
- Lack of integration with other development tools such as Git, Jenkins, or CI/CD tools, which makes the tracking manual and increases the effort required.
- User interface complexity or poor user experience If the application is not intuitive or requires many steps, it is less used and the possibility of error increases.
- Difficulty in classifying and prioritizing the illnesses: The absence of a clear mechanism to determine the seriousness of the disease or its impact on the system.
- Poor statistical analysis and reports The lack of in-app analysis tools prevents teams from understanding fault patterns or evaluating performance.
- Performance problems or expansion of large projects, the application may experience slowness or difficulty in dealing with a large number of bugs.

1.3. Research Significance

The bug tracking application project is one of the vital projects in the field of software development, as it has a direct impact on improving software quality, speeding up maintenance processes, and enhancing cooperation between teams.

1.3.1. At the theoretical level

The importance of the software bug tracking project at the theoretical level lies in consolidating software engineering concepts, achieving integration between the stages of the development lifecycle, and enhancing the quality of software through systematic tracking and structural analysis of bugs.

- The project practically embodies concepts such as requirements management, traceability, and bug lifecycle, enhancing theoretical understanding among students and developers.
- It helps understand the relationship between software bugs and other system components such as code, design, and test cases
- The project provides a model for classifying bugs by type, source, and impact, making it easier to study recurring patterns and identify weaknesses in the system.
- Enables testing theories related to software quality such as "early defects are less expensive"
- It is an ideal example of project-based learning (PBL), as it integrates theoretical knowledge with practical application, enhancing students' deep understanding and analytical skills

1.3.2. At the practical level

The importance of a bug tracking application project at the application level lies in improving software quality, speeding up maintenance processes, and enhancing coordination between development and testing teams. This type of application is a vital tool in modern development environments, especially in complex or multi-team projects.

- Improve the quality of the finished product by tracking the bugs accurately, which can be addressed before the product is released, which reduces errors and increases users' satisfaction.
- Accelerate maintenance and technical support processes Having a detailed bug log makes it easy for support teams to quickly identify and address issues, especially in future releases.
- Enhance collaboration between teams The app provides a unified platform for exchanging information between developers, testers, and project managers, reducing misunderstandings and increasing productivity.
- Organize work and prioritize By categorizing bugs by severity or impact, the team can focus on critical issues first, which improves time and resource management.

References

- Foster, G. (2025). Building an Effective Bug Tracker Project with GitHub. Graphite. Link → A practical guide to creating and managing an efficient bug tracking system using GitHub.
- Khamar, J. (2025). Top Bug Reporting and Tracking Tools: Eliminate Project Bottlenecks. DhiWise. Link → A review of the best bug tracking tools in 2025, including features and comparisons.
- Peham, T. (2025). 17 Excellent Open Source Bug Tracking Tools. User snap Blog. Link → Analysis of open-source bug tracking tools and their role in improving user experience.
- LibHunt. (2025). Top 14 Bug-Tracker Open-Source Projects. Link → A curated list of open-source bug tracking projects, including tools like Open Project and RestlessIDE.
- Vander Heyden, N. (2024). 16 Best Open-Source Bug Tracking Tools: Comparison Guide. Marker.io Blog. Link → A detailed comparison of open-source bug tracking tools such as BugZilla and MantisBT.