SMART BUG TRACKING SYSTEM

Navya R, Dr. Charles Arockiaraj M

Student, Professor, Department of Computer Application

A.M.C. Engineering College, Bengaluru, India

ABSTRACT

The paper explores the implementation of a "Bug Tracking System" intended to overcome the limitations of the manual approach previously employed. The purpose of this system is to address and mitigate the challenges encountered in the existing system. To ensure seamless and efficient operation, the development process considered the specific requirements of the organization. The user interface of the software has been designed to be intuitive and straightforward, minimizing the likelihood of data input errors. Additionally, the system provides error messages to alert users if incorrect data is entered. The system does not necessitate any specialized or academic knowledge, making it accessible to a wide range of users. This clearly demonstrates that user convenience was a key consideration during the design phase. As previously discussed, the adoption of a defect tracking system can contribute to the establishment of a streamlined management system that is characterized by speed, accuracy, reliability, and trustworthiness. By leveraging this tool, users can devote more time to their core responsibilities instead of focusing on tedious record-keeping tasks. Consequently, businesses can optimize the utilization of available resources, thanks to this development.

INTRODUCTION:

Bug tracking and management play a crucial role in the software development lifecycle, not limited to the testing stage but throughout all stages. Issues can arise at any stage of development, including requirements gathering (such as missing or conflicting requirements), design (such as lost functionality), implementation (such as incorrect usage), and usage (such as unexpected outcomes or utility loss). Having an effective bug description and tracking system is essential for any software project.In the context of natural language processing, identifying typical error signals can be challenging due to the diverse applications of natural language data. While the execution results might be more reliable when considering data related to the execution, there are limitations. The relevance of the data, whether in standard language or related to usage, may vary depending on the specific circumstances and context in which it is used.When incorporating both types of data, several points of interest can be derived. Firstly, the natural language data derived from error descriptions often represents the behavior of external errors observed by the error reporting professional. On the other hand, execution data may capture the internal anomalies occurring within the system. By utilizing both types of data when composing an error report, it becomes possible to consider both the external and internal behavior of the system. This holistic approach allows for a comprehensive understanding of the issues and facilitates effective bug resolution.In summary, incorporating a robust bug tracking and management system throughout the software development lifecycle is essential. It helps identify and address issues at various stages, ensuring the development of high-quality and reliable software. Utilizing both natural language and execution data in bug reporting enhances the understanding of system behavior and aids in effective issue resolution.

LITERATURE SURVEY:

1.An eye tracking research on debugging strategies towards different types of bugs

AUTHORS: Fei Peng; Chunyu Li; Xiaohan Song;
Wei Hu; Guihuan Feng

The process of debugging in software development is crucial for ensuring the quality of the software. However, individuals who employ different debugging approaches often experience varying levels of success when it comes to detecting and fixing defects. While some studies have investigated alternative debugging procedures, there is limited research on the impact of these strategies on different types of errors.This article presents research conducted on twenty participants, revealing differences in eye movement data between successful and unsuccessful debugging attempts. The analysis of the research results led to these findings. When dealing with issues related to data flow, it is beneficial to focus on monitoring variable changes. On the other hand, when addressing control flow errors, it is essential to carefully examine the code and fully comprehend its logical structure.By combining these conclusions with the error messages provided by the compiler, programmers can more efficiently detect faults, saving time and resources.

2.Bug Tracking Process Smells In Practice

AUTHORS: Erdem Tuna; Vladimir Kovalenko; Eray Tuzun

Bug tracking (BT) technologies are employed by software development teams to report and manage issues. Each record in a bug tracking system, also known as a BTS, consists of multiple information fields. Different tracking methods generate reports with comparable but not identical information. Since team workflows vary, there is no single universally recommended approach for bug tracking; however, best practices suggested in both white and grey literature can serve as guidelines. It is possible that development teams may not fully adhere to industry best practices in their bug tracking process.This study examines the non-compliance of developers with recommended practices, often referred to as "smells," within the bug tracking process. To investigate the occurrence of BT smells in an industrial context, we analyze bug reports from four different projects stored in JetBrains' Bug Tracking System (BTS). Additionally, we conduct a survey among developers to assess (1) their awareness of these smells, (2) their perception of their seriousness, and (3) the potential benefits of a BT process smell detection tool. Our findings are as follows: (1) Smells can indeed be present, but their detection requires a strong understanding of the project's BT practices. (2) The perceived severity of a smell varies depending on the specific type of smell. (3) Developers believe that a smell detection tool would be valuable for six out of the 12 different smell categories.

3.Feature Ranking and Aggregation for Bug Triaging in open source issue Tracking System

AUTHORS: Anjali Goyal; Neetu Sardana

The increasing complexity of team projects has led to the proliferation of diverse tools and methodologies for project management. In managing open-source projects, the utilization of bug tracking tools is a crucial aspect. In recent decades, software development projects have witnessed a significant rise in the number of defect reports, a trend that is expected to continue. Managing these incoming issues, particularly the process of bug triage, poses a significant challenge. Bug triage involves sifting through numerous bug reports to select a suitable software developer responsible for addressing the reported defect promptly. Automated bug triaging strategies, either in part or in their entirety, can be found in existing research. These algorithms typically consider various bug characteristics to select well-suited developers.Previous research has identified several distinct qualities that are fundamental in the bug triaging process, although there is currently no consistent ranking scale to indicate the importance of specific issue characteristics. This research introduces a system for ranking non-textual bug parameters, providing a method for feature ranking and data aggregation. The proposed technique has been validated using various open-source software
projects, including Mozilla Firefox, Eclipse, GNOME, and OpenOffice.The experimental evaluation concluded that the ranking of bug parameters remains consistent across different open-source projects within the Bugzilla repository. This finding demonstrates the applicability and effectiveness of the proposed methodology.

4.An Expert System Framework for Bug Tracking and Management

AUTHORS: Abdul Wahab Khan; Sanjay Kumar

The use of issue tracking systems extends beyond open-source software projects and encompasses corporations and closed-source software practices as well. Understanding and working with issue tracking systems is vital not only for developing effective plans but also for a wide range of research activities. In the case of initiatives involving open-source software, a significant amount of information is publicly available, often with minimal barriers to access. This data, when carefully analyzed, can provide valuable insights into various aspects of Open Source projects, such as project location and progression.By leveraging the knowledge accumulated from diverse issue tracking systems, it is possible to make informed decisions regarding the development practices of open-source software as a whole. However, effectively accessing and utilizing this wealth of information for

exploratory research can present significant challenges. This study aims to demonstrate an application that standardizes and harnesses the data from a substantial number of issue tracking systems, leveraging the insights gained from this research.Through the findings of this investigation, we will showcase the development of an application that effectively utilizes and presents the information gathered from various issue tracking systems. The results of this study will contribute to overcoming the challenges associated with accessing and making use of the vast potential of issue tracking data for exploratory research purposes.

5.Bugtrac- A New Improved Bug Tracking System

AUTHORS:Madhwaraj Kango Gopal; Govindaraj M; Paramita Chandra; Prathiksha Shetty; Sunny Raj

Software testing involves the identification and discovery of bugs and other issues in computer code. The majority of software defects are caused by errors and omissions made by individual programmers in the source code or technical design of a program. These errors can manifest in various ways when using software applications, making it challenging for users to accurately detect and manage all types of bugs. To address this challenge, bug tracking systems have been developed to monitor and track reported bugs in applications. These systems aim to provide a solution to the problem at hand.Currently, there are various proprietary and open-source bug tracking systems available in the market. New systems are also being developed to accommodate the evolving needs of software and the wide range of emerging faults. Given the continuous evolution of software applications, there is a demand for a tool that can assist in bug correction and tracking the status of bug fixes.This article presents the findings of a comparative study conducted on five different defect tracking systems, including both open-source and proprietary software. The research aimed to determine the superiority among these systems. As a result, a new bug tracking system called "Bugtrac" is proposed. Bugtrac is designed to address the diverse types of software applications and the numerous bugs that can arise from using these programs. It is expected that the proposed system will evolve into a more promising defect tracking solution compared to existing systems.

SYSTEM ANALYSIS:

EXISTING SYSTEM:

Testers play a crucial role in the software validation process by identifying and reporting bugs to the Project Manager and developers using communication channels such as shared email lists and documents. However, relying solely on these manual channels for bug reporting is prone to errors and may result in some issues being overlooked or disregarded. The lack of a specialized tracking system further exacerbates this problem.In the current setup, the responsibility of maintaining a record of reported issues falls on the software development team participating in the software development life cycle. However, the limitations of existing technology hinder their ability to fulfill this responsibility effectively. This limitation negatively impacts productivity and accountability within the team.To address these challenges, there is a need for an improved bug tracking system that can streamline the bug reporting process, ensure accurate recording of issues, and facilitate efficient issue resolution. By implementing a specialized tracking system, teams can enhance productivity, improve issue management, and foster a greater sense of accountability throughout the software development life cycle.

PROPOSED SYSTEM:

The creation of the new system encompasses the methodologies outlined in the subsequent paragraphs. These steps aim to automate the entire process, considering the approach to integrating the database.

1. The implementation of the new system incorporates the steps outlined in the following paragraphs. These procedures aim to automate the entire process, with careful consideration given to the strategy for integrating the database.

2. The establishment of the new system involves following the procedures described in the subsequent paragraphs. These steps aim to automate the entire process, while also taking into consideration the approach to integrating the database.

3. One of the objectives during the development of the application was to enhance user friendliness by incorporating a diverse range of controls.

4. Improving user-friendliness was a key focus during the application's development, achieved through the inclusion of various controls.

5. Enhancing user experience was a primary aim throughout the development of the application, achieved by incorporating an assortment of user-friendly controls.

6. The management of the entire project can be optimized and customized with the assistance of the system.

7. The system facilitates the efficient and adaptable administration of the entire project.

8. The system enables streamlined project management and adaptability across the entire project.

9. Throughout the project's development stages, there will be no instances where data management is carried out in an inappropriate manner.

10. At no point during the project's development, there will be any possibility of data being mishandled or managed in an unsuitable way.

11. During every phase of the project's development, the data will be consistently and appropriately managed, leaving no room for any mishandling.

12. The system provides robust security measures and offers users a diverse range of authentication methods to choose from, ensuring a wide array of options.

13. The system ensures a high level of security and presents users with multiple authentication methods to choose from, offering a comprehensive selection of options.

14. With a strong emphasis on security, the system offers users a variety of authentication methods to cater to their preferences, delivering a wide range of choices.

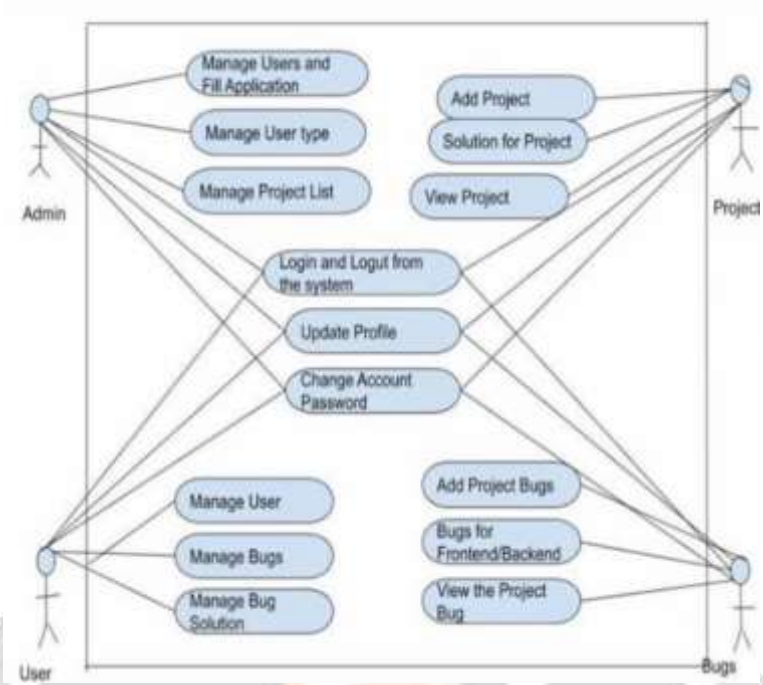
IMPLEMENTATION:

Admin: This module exercises complete control over all other modules within the system, possessing comprehensive authority over them. The administrator takes the lead in initiating project creation and subsequently delegates managerial responsibilities to the newly appointed managers. Subsequently, the administrator assigns bugs to managers in a prioritized manner, based on their level of importance, and populates the managers with team members.

Manager: The manager assumes responsibility for resolving the assigned issue and enjoys unrestricted access to the project they are overseeing. The bug list managed by the manager is accessible for developers to review.

Tester: As a tester, you will be granted project access and have visibility into the assigned issues managed by the management team. You can thoroughly analyze the project, report bugs to management, and contribute new bugs to the list. Upon logging into the system, testers will have immediate access to the list of projects assigned to them.

Reports: Both the manager and the administrator possess access to this module, enabling them to generate reports based on relevant information specific to their roles. They have the capability to extract and analyze data according to their respective responsibilities.



SYSTEM ARCHITECTURE:

This project endeavors to construct an innovative web-based Bug Tracking System, which offers a unique and optimal solution for recording and managing faults present in various solutions, products, or applications. The primary objective of this project is to develop an efficient bug tracking system that effectively addresses this issue. By utilizing this bug tracking system, either an individual developer or a team of developers can seamlessly track and prioritize unresolved issues within their product. The system maintains a chronological order of defects, facilitating efficient issue management.By incorporating a bug tracking system, organizations can significantly enhance individual workers' accountability and productivity levels. This software provides valuable feedback, restoring workflow and allowing for higher levels of productivity. Whether utilized by individual testers or testing teams, the bug tracking software enables effective tracking of unresolved issues within their programs. The system facilitates code modifications, seamless communication among team members, bug tracking, submission and evaluation of connections, as well as enforcing quality assurance standards.The scope of this project is broad, as it is not limited to any specific institution. The aim is to develop a generic software solution applicable to a wide range of businesses and organizations operating in the global marketplace. Additionally, this software offers a convenient alternative for individuals who utilize it. Furthermore, the bug tracking software will generate a significant number of data summaries, providing valuable insights for users.

CONCLUSION:

The adoption of a bug tracking system simplifies the process of identifying and resolving errors that may have been introduced into a software product promptly. The Bug Tracking System (BTS) implemented in this project serves as a valuable tool for tracking faults in project modules and providing support for error troubleshooting during both testing and development phases. The project ensures the elimination of any potential delays in problem reporting across all levels and modules within the software industry.However, it is important to note that the cost increases significantly when deploying the software on an existing company-owned server.

REFERENCES:

[1] Sultan, Torky, Khedr, Ayman E., Sayed, Mostafa. "A Proposed Bug Tracking Model for Classifying Embedded Defect Reports to Enhance Program Quality Control," ACTA Informatica MEDICA, vol. 21, no. 2, 2013, pp. 103-108.

[2] Wang, Yajie, Jiang, Ming, Wei, Yueming. "A Software Quality Framework for Mobile Application Testing," Proceedings of the Fourth International Conference on Advances in System Testing and Validation Lifecycle, 2012.

[3] Chandani, Priyanka, Gupta, Chetna. "An Investigation on Effective Defect Prevention - 3T Approach," MECS Journal, vol. 1, 2014, pp. 32-41. (Distributed online in February 2014).

[4] Salger, Frank, Sauer, Stefan, Engels, Gregor."A Coordinated Quality Assurance System for Detecting Trade Data Systems," Proceedings of the CAiSE Conference, 2009.

[5] Yau, Stephen S., Wang, Yeou-Wei, Huang, Jules G., Lee, Jinshuan E. "A Coordinated Master Framework System for Software Quality Assurance," IEEE Transactions on Software Engineering, vol. 16, no. 2, 1990, pp. 161-171.