

## 





**Development of Static Code Analysis Application for Bug and Vulnerabilities Detection**

**Bahrul Rozak11, Writer\_Name\_22**

1,2Study Program Name, Faculty Name

1,2University or Affiliation Name (no abbreviation allowed)

1.2Complete Address of University or Affiliation

E-mail: 1xxx@university.ac.id, 2xxx@university.ac.id, 3xxx@university.ac.id

|  |  |  |  |
| --- | --- | --- | --- |
| |  |  | | --- | --- | | **Article:**  Accepted: xxx xx, 20xx  Revised: xxx xx, 20xx  Issued: xxx xx, 20xx  \***Correspondence Address:**  (email) | **ABSTRACT**  Abstract is a synopsis of the work containing the problems studied, the purpose of research, information and methods used to solve problems, and conclusions. Articles must be submitted in print-ready format and are limited to a minimum of ten (10) pages and a maximum of twelve (12) pages. Abstract is a synopsis of the work that contains the issues studied, the research purpose, the information and methods used to solve the problem, and the research conclusion. Abstracts are limited to 200 words and should not contain references, mathematic equations, figures, and tables. The font size for abstracts, keywords, and body of article is 11pt. Keywords are no more than six (6) words, but the minimum is three (3) words.  **Keywords:** *Web, Asset Management, CodeIgniter, Bootstrap* | | **ABSTRACT**  Software development is a crucial activity in today's digital age. High-quality software not only guarantees good performance, but also ensures the security of user data. However, the software development process is not easy and is often faced with various challenges, one of which is the problem of bugs and security vulnerabilities in program code. Bugs are errors in program code that can cause the program to not run as expected or even fail. On the other hand, security vulnerabilities are gaps in the system that can be exploited by irresponsible parties to gain unauthorized access. Both of these issues can have serious repercussions, both for users and software providers. To address these issues, tools are needed that can assist developers in detecting and addressing bugs and security vulnerabilities in program code. One effective approach is to use static code analysis. Static code analysis is a technique used to examine program code without having to run it, with the aim of finding potential problems in the code. In this research, we developed a static code analysis application that aims to detect bugs and security vulnerabilities in program code. This application uses the Streamlit framework, a Python framework that allows developers to easily create interactive web user interfaces. By using this application, developers can quickly and efficiently detect and address issues in their program code. This research will discuss in detail about the development of a static code analysis application using Streamlit, including the analysis techniques used, the implementation of the application, and the evaluation of the effectiveness and efficiency of the application in detecting and resolving bugs and security vulnerabilities in program code. It is expected that this application can help improve the quality and security of the software developed.  **Keywords:** *Bug Detection, Security Vulnerabilities, Static Code Analysisi, Software Development* |
|  | |

1. **INTRODUCTION**

Software development plays a central role in today's digitally transformed world. High-quality software not only improves overall system performance, but also keeps sensitive data and information safe from security threats. However, in the quest to produce quality software, development teams are often faced with complex challenges that can slow down the development process.

These include technical issues, such as scalability, security, and interoperability, as well as non-technical challenges, such as project management, communication between teams, and a deep understanding of user needs. Therefore, it is important for the development team to have a strong understanding of these various aspects and collaborate effectively to overcome these challenges for successful software development.

In developing software, development teams are often faced with various complex challenges that can hinder the development process. One of the main challenges often faced is the presence of bugs or errors in the program code. Bugs can appear in many forms, ranging from simple syntax errors to complex logic issues. When bugs occur, the program does not function correctly, produces unwanted output, or even causes the program to fail.

Fixing bugs can be a complicated and time-consuming task, especially in large and complex software projects. The debugging process involves identifying the bug, understanding its cause, and implementing the right solution. Sometimes, bugs can arise as a result of complex interactions between different parts of the program, making the debugging process even more challenging.

In addition, other challenges in software development include efficient resource management, maintenance of complex code, ensuring software security, and meeting user needs appropriately.

In addition to bug issues, security vulnerabilities are also a major concern in software development. Security vulnerabilities are gaps in the system that can be exploited by irresponsible parties to gain unauthorized access. Identifying and addressing security vulnerabilities in software is crucial to maintaining the integrity and security of user data.

Security vulnerabilities can arise from various aspects, including insecure software design, the use of components or libraries that are vulnerable to attack, or a lack of understanding of security best practices in software development. Efforts to address security vulnerabilities involve using secure development practices, such as implementing strict access controls, properly validating and filtering user input, and performing regular monitoring and maintenance of the software.

In addition, it is important to keep software up-to-date with the latest security patches and keep abreast of developments in the world of information security. Information security training and awareness is also important for development team members so that they can recognize potential security risks and take the necessary preventive measures. By paying attention to security aspects from the beginning of development, development teams can reduce the risk of security vulnerabilities in the software they develop.

To overcome the complex challenges in software development, developers need effective tools and techniques. One commonly used approach is static code analysis. Static code analysis is a technique used to examine program code without having to run it. This technique can help identify bugs and security vulnerabilities in the program code before the application is run, thus allowing developers to fix them before the application is released to the public.

Static code analysis involves the use of automated code analysis tools that can examine program code for potential bugs, syntax errors, and security vulnerabilities. The tool performs a thorough scan of the program code, analyzes the code structure, and looks for patterns that could indicate the presence of bugs or security vulnerabilities. By using static code analysis, developers can identify potential issues in their code quickly and efficiently, allowing them to make improvements before the code is used in production.

In addition, static code analysis can also help developers to improve the overall quality of their code. By thoroughly analyzing the code, developers can find and remove code that is inefficient, redundant, or difficult to understand, thus making the code easier to maintain and update in the future. Thus, the use of static code analysis can help developers to overcome complex challenges in software development and improve the quality and safety of the resulting product.

In this context, this research aims to develop a static code analysis application using the Streamlit framework. Streamlit is a Python framework that allows developers to create interactive web user interfaces with ease. This application will assist developers in detecting bugs and security vulnerabilities in program code more efficiently.

The development process of this app involves steps such as designing an intuitive user interface, integrating static code analysis algorithms into the app, and testing the app to ensure it performs as expected. With this app, developers can quickly and efficiently scan their program code, identify potential bugs and security vulnerabilities, and take the necessary corrective actions before the code is used in production.

It is expected that this research can make a positive contribution in improving the quality and security of the developed software. By using the developed static code analysis application, developers can reduce the risk of bugs and security vulnerabilities in their program code, thus improving the reliability and security of the software produced. In addition, this application can also assist developers in accelerating the software development process by providing a more efficient and easy-to-use tool in analyzing program code.

* 1. **Writing Guide of Introduction Chapter**

In the introductory chapter, the author should provide adequate background, and a brief literature survey to show differences from previous research, to show what you hope to achieve, and to point out the scientific achievements or novelties of the submitted article manuscript. This section also includes the writing of relevant previous studies. Furthermore, the writer must state the purpose of the research at the end of the introductory chapter.

Please also add at the end of the introduction, a statement that this manuscript is the original result of your research or the development of previous research using your own ideas. Articles that you submit will be thoroughly checked with plagiarism checker software (Turnitin). If the results of the examination are more than **20%**, then we reserve the right to reject articles from the author. **[Font: Times New Roman 11 point, Justify]**

1. **METHODOLOGY**

The method explained in this part is scientific and shall make the readers can repeat the experiment you have conducted. Please follow the guidelines which is more detail in Author Guideline. **[Font: Timer New Roman 11 points, Justify]**

1. **RESULTS AND DISCUSSION**

The experiment result shall be explained clearly and shortly. It is expected that the discussion written in this chapter can be supported by the relevant theories. Please follow the guideline which is more detail in Author Guideline. **[Font: Times New Roman 11 point] [All paragraphs of Justify]**

1. **CONCLUSION**

The closing section should be in the form of a paragraph that addresses the research objective, telling how your work can advance current knowledge. After the closing section, the authors are expected to be able to follow the Bibliography writing format provided below, which follows the IEEE format and refers to at least 10 articles in indexed journals or national and international proceedings published in the last 5 years. In addition, the author can also include "Acknowledgments" after the Bibliography section. Please follow the more detailed instructions in the Author Guideline.

Thank you for reading this writing guide to the end. The authors of the articles are expected to send their CV containing the phone number to Open Journal System, in a Supplementary File or sent to our e-mail address, namely jurnal-ti@uinjkt.ac.id. As for article writers who do not send their CV, they will not be published in Jurnal Teknik Informatika. **[Font: Times New Roman 11 points] [All Paragraphs of Justify]**

**BIBLIOGRAPHY**

1. Successive quotes in brackets [1]. Sentence punctuation following bracket [2]. Refer only to reference numbers, as in [3] -Do not use "Ref. [3]" or "Ref. [3].
2. Minimum bibliography is **10 citations** from **journals/research**, maximum published in the **last 5 years**.
3. Mandatory to use Mendeley for citation purposes and bibliography in IEEE format) Mendeley can be downloaded at <https://www.mendeley.com/downloads>

[1][2][3][4][5][6][7][8][9][10][11][12][13][14][15][16][17][18][19][20][21][22][23][24][25]

[1] G. Padmavathi, D. Shanmugapriya, and M. Kalaivani, “A Study on Vehicle Detection and Tracking Using Wireless Sensor Networks,” *Wirel. Sens. Netw.*, vol. 02, no. 02, pp. 173–185, 2010, doi: 10.4236/wsn.2010.22023.

[2] B. A. Kitchenham and S. Charters, “Guidelines for performing Systematic Literature Reviews in Software Engineering. EBSE Technical Report EBSE-2007-01. School of Computer Science and Mathematics, Keele University,” no. January, p. 2007, 2007.

[3] C. Sharp *et al.*, “Design and implementation of a sensor network system for vehicle tracking and autonomous interception,” *Proc. Second Eur. Work. Wirel. Sens. Networks, EWSN 2005*, vol. 2005, pp. 93–107, 2005, doi: 10.1109/EWSN.2005.1462002.

[4] F. S. Cohen, S. Abushariefeh, G. Bruton, M. Matthews, and K. Varghese, “Tracking a vehicle moving in a wireless sensor network,” *Lect. Notes Inst. Comput. Sci. Soc. Telecommun. Eng.*, vol. 28 LNICST, pp. 438–449, 2010, doi: 10.1007/978-3-642-11723-7\_29.

[5] J. Li, Z. Xing, W. Zhang, Y. Lin, and F. Shu, “Vehicle tracking in wireless sensor networks via deep reinforcement learning,” *IEEE Sensors Lett.*, vol. 4, no. 3, pp. 1–4, 2020, doi: 10.1109/LSENS.2020.2976133.

[6] R. K. Megalingam, V. Mohan, A. Mohanan, P. Leons, and R. Shooja, “Wireless sensor network for vehicle speed monitoring and traffic routing system,” *ICMET 2010 - 2010 Int. Conf. Mech. Electr. Technol. Proc.*, no. Icmet, pp. 631–635, 2010, doi: 10.1109/ICMET.2010.5598438.

[7] D. Parthasarathy, R. Whiton, J. Hagerskans, and T. Gustafsson, “An in-vehicle wireless sensor network for heavy vehicles,” *IEEE Int. Conf. Emerg. Technol. Fact. Autom. ETFA*, vol. 2016-Novem, no. Figure 2, 2016, doi: 10.1109/ETFA.2016.7733554.

[8] J. Tavares, F. J. Velez, and J. M. Ferro, “Application of Wireless Sensor Networks to automobiles,” *Meas. Sci. Rev.*, vol. 8, no. 3, pp. 65–70, 2008, doi: 10.2478/v10048-008-0017-8.

[9] M. Razfar *et al.*, “Wireless network design and analysis for real time control of launch vehicles,” *IEEE Int. Conf. Wirel. Sp. Extrem. Environ. WiSEE 2013 - Conf. Proc.*, pp. 7–8, 2013, doi: 10.1109/WiSEE.2013.6737574.

[10] K. N. Qureshi, A. H. Abdullah, and R. W. Anwar, “Wireless sensor based hybrid architecture for vehicular Ad hoc networks,” *Telkomnika (Telecommunication Comput. Electron. Control.*, vol. 12, no. 4, pp. 942–949, 2014, doi: 10.12928/TELKOMNIKA.v12i4.537.

[11] M. Potdar and S. Wani, “Wireless Sensor Network in Vehicles,” *SAE Tech. Pap.*, vol. 2015-April, no. April, 2015, doi: 10.4271/2015-01-0241.

[12] V. S. R, “A Vehicular Wireless Sensor Network for Vehicle Emission Monitoring and Fuel Indication,” *Int. J. Eng. Res. Technol.*, vol. 4, no. 18, pp. 1–4, 2016.

[13] D. Khachane and A. Shrivastav, “Wireless Sensor Network and its Applications in Automobile Industry,” *Int. Res. J. Eng. Technol.*, pp. 2395–56, 2016, [Online]. Available: https://www.irjet.net/archives/V3/i5/IRJET-V3I5452.pdf.

[14] J. P. Benson *et al.*, “Car-park management using wireless sensor networks,” *Proc. - Conf. Local Comput. Networks, LCN*, pp. 588–595, 2006, doi: 10.1109/LCN.2006.322020.

[15] S. Sinthuja and S. V. Saravanan, “Vehicle-2-vehicle communication based on wireless sensor network,” *Bull. Electr. Eng. Informatics*, vol. 6, no. 4, pp. 364–366, 2017, doi: 10.11591/eei.v6i4.868.

[16] J. Alcaina, Á. Cuenca, J. Salt, M. Zheng, and M. Tomizuka, “Energy-efficient control for an unmanned ground vehicle in a wireless sensor network,” *J. Sensors*, vol. 2019, 2019, doi: 10.1155/2019/7085915.

[17] Y. S. Deraman, I. Ahmad, A. F. Ramli, and N. A. M. Mortar, “Optimization performance of unmanned aerial vehicle in wireless sensor network,” *AIP Conf. Proc.*, vol. 2291, no. November, 2020, doi: 10.1063/5.0025092.

[18] B. Etikasari, H. Husin, S. Kautsar, H. Y. Riskiawan, and D. P. S. Setyohadi, “Wireless sensor network development in unmanned aerial vehicle (uav) for water quality monitoring system,” *IOP Conf. Ser. Earth Environ. Sci.*, vol. 411, no. 1, 2020, doi: 10.1088/1755-1315/411/1/012061.

[19] J. da C. V. Rezende, R. I. da Silva, and M. J. F. Souza, “Gathering big data in wireless sensor networks by drone,” *Sensors (Switzerland)*, vol. 20, no. 23, pp. 1–27, 2020, doi: 10.3390/s20236954.

[20] A. Elmrini and A. Ghacham Amrani, “Wireless Sensors Network for Traffic Surveillance and Management in Smart Cities,” *Colloq. Inf. Sci. Technol. Cist*, vol. 2018-Octob, pp. 562–566, 2018, doi: 10.1109/CIST.2018.8596636.

[21] E. Matthews, *Python Crash Course*. 2019.

[22] A. Sweigart, *Automate the Boring Stuff with Python, 2nd Edition: Practical Programming for Total Beginners*. 2015.

[23] D. Amos, “A Practical Introduction to Web Scraping in Python.” https://realpython.com/python-web-scraping-practical-introduction/ (accessed May 19, 2021).

[24] A. Treadway, “HOW TO SCRAPE NEWS ARTICLES WITH PYTHON,” 2020. http://theautomatic.net/2020/08/05/how-to-scrape-news-articles-with-python/ (accessed May 21, 2021).

[25] C. Bajracharya, R. Grodi, and D. B. Rawat, “Performance analysis of wireless sensor networks for wind turbine monitoring systems,” *Conf. Proc. - IEEE SOUTHEASTCON*, vol. 2015-June, no. June, 2015, doi: 10.1109/SECON.2015.7133053.

**ARTICLE WRITING FORMAT OF JURNAL TEKNIK INFORMATIKA**

1. **GENERAL FORMAT**

The writing is in English in at least 10 pages, typed using A4 size paper (210 mm x 297 mm) with two columns with top and bottom margins and right and left margins, 2.5 cm. In the middle of the first page contains the title of the manuscript. Under the title presented abstract and keywords from the abstract. Abstract typed using single space, and **Times New Roman font, 11pt**. Manuscripts submitted must be in the form of a **Microsoft Word** document.

* 1. **Chapters and subchapters**

The titles of chapters and sub-chapters are capitalized at the beginning of each sentence and followed by lowercase letters in bold.

* 1. **Headers and footers**

Avoid writing headers, footers or page numbering in the submitted manuscript. Page numbering will be added at the final stage of journaling.

* 1. **Abbreviations and acronyms**

Define abbreviations and acronyms at the beginning of paragraphs when they are used in sentences. Important abbreviations have been defined in the abstract. Other abbreviations such as IEEE, SI, MKS, CGS, AC, DC, RMS and other common abbreviations need not be defined. Abbreviations must be without spaces, for example "S. T." the correct one is "S.T.". Do not use abbreviations in titles, chapter titles or subsections, unless it is unavoidable. Terms that use a foreign language are written in *italic style*.

1. **TABLES AND FIGURES**

The table displayed should not be in the form of an image (jpg or png) so that the editing process can be carried out. Figures are titled with the initial words “Figure” and “Table” and are numbered, for example: Figure 1, Figure 2, Table 1, Table 2, and so on. Not allowed to write "Table below" or "Figure below". The Figure Title is placed at the bottom center of the image. The Table Title is placed in the top center of the table. Both figures and tables are centered between the right and left margins of the page. Tables or figures must be placed on the same page as the title of the table or figure. If the table or figure exceeds 1 page, it can be reduced in size.

*Table 1. Centered, 10pt, and shows the table title (capital letters at the beginning of the sentence only)*

|  |  |  |
| --- | --- | --- |
| Description | August | September |
| Title | Organic Chemistry | Classic Physics |
| Name | Ian Mahendra | Bambang JK |
| Address | Semarang | Kendal |



Figure 1. The title of the image uses only capital letters at the beginning of the sentence and the image uses a high resolution that can be clearly seen (not broken or blurry)

1. **EQUATION**

When numbering an equation, put brackets and the number is placed to the right of the equation. Equations are mentioned in the discussion by writing Equation 1, Equation 2, and so on. Equations must be typed using Insert Equation in Microsoft Word, not images or inserts. Example:

*Z* = *c*0 + *c*1*X*1 + *c*2*X*2 + *c*3*X*3 + *c*4*X*4(1)

1. **EXCERPTS/REFERENCES**

Excerpts or references to other sources of writing put forward by the author in the body of the article must be adjusted to the references in the Bibliography, which are sorted by the appearance of the excerpts, not alphabetically. Reference sources are expected to be mostly primary sources, either in the form of journals or proceedings, with a minimum of 10 articles in indexed journals or national and international proceedings published in the last 5 years.

To make it easier, authors are required to use the facilities in the MENDELEY or ENDNOTE software as a reference manager. Authors can also use the REFERENCES menu and the BIBLIOGRAPHY submenu in Microsoft Word.

The excerpts number is placed in square brackets (IEEE format) as in the following example: [1]. The numbering follows the reference number written in the bibliography section. If you want to refer at the beginning of the sentence, please write only the number in square brackets then the statement follows, for example "[3] indicates that ...". If the statement mentioned first, then place the reference number at the end of the sentence, one space from the last sentence and end with a period without a name or anything. References referred to in the Bibliography must be more than 10 references.

Capital letters are only placed at the beginning of the sentence, except for the title of the manuscript, chapter and sub-chapter titles and symbols. For excerpts from articles in English, it can still be written in English, and written in *italic style*.