Journal of Computing and Data Science (JOCDAS)
Vol. 2, No. 1, 2024, pp. 19~31

BLACK BOX TESTING ON A MINI ERP SYSTEM: A STRATEGY FOR BUG DETECTION AND MINIMISATION

Mohammad Zidan Alvaro*[1], Muhammad Annas Darunnaja[2], Muhammad Ainul Yaqin[3]
[123] Universitas Islam Negeri Maulana Malik Ibrahim, Jalan Gajayana 50, malang, Indonesia
[1]220605110114@student.uin-malang.ac.id [2]220605110130@student.uin-malang.ac.id
[3] yaqinov@ti.uin-malang.ac.id

*Corresponding Author

Abstract

This research discusses the application of Black Box testing on a mini ERP system, with a focus on bug recognition and reduction. Mini ERP systems are crucial in managing the business activities of small and medium-sized companies. However, like any other software, mini ERP systems are prone to bugs that can disrupt system performance and incur costs for the company. The main problem faced is the high frequency of undetected bugs, which can potentially cause operational disruptions and financial losses. This research presents an efficient Black Box testing strategy to identify bugs in mini ERP systems without requiring knowledge of the internal structure of the system. Through this approach, various testing techniques and methods are reviewed to improve accuracy in bug discovery and reduce the risk of system failure. The findings from this research are expected to benefit software developers and business owners by enhancing the quality and reliability of their mini ERP systems.

Keywords: ERP System, Blackbox Testing, Bugs, Quality.

INTRODUCTION

In the world of small and medium-sized enterprises (SMEs), mini ERP (Enterprise Resource Planning) systems are important tools for organizing various business activities, including inventory management, financial operations, and human resources [1]. The effectiveness of such systems is critical, as they greatly affect a company's efficiency and productivity. This research discusses the strategic application of Black Box testing to improve the quality of mini ERP systems [2]. Black Box testing is a method that evaluates software functionality without the need to understand its internal code structure, using a pragmatic approach to validate and optimize the performance of the mini ERP system.

The identified research gaps indicate a lack of comprehensive studies on effective Black Box testing strategies customized for mini ERP systems. This paper aims to bridge this gap by presenting a thorough analysis of Black Box testing methodologies aligned with the unique characteristics and complexities of mini ERP systems [3]. Additionally, it addresses the challenge of customizing Black Box testing to meet end-user requirements, ensuring that the testing process reflects the actual user experience.

Through a literature review, this paper highlights the critical role of Black Box testing in software development, emphasizing its importance in web-based data processing applications due to the complexity and sensitivity of the data involved. Previous studies have provided insights into best practices and specific techniques relevant to Black Box testing for web applications and data processors [4].

The paper also refers to significant contributions to the field, such as the study on ERP Warehouse Management implementation to improve company performance through data and process integration [2]. It emphasizes the need for changes in work patterns for successful implementation and the important role of Warehouse Management Systems in reducing costs, optimizing equipment usage, preventing

inventory losses, and improving storage efficiency. In addition, this paper discusses the design of ERP system features for the inventory module aimed at reducing the Bullwhip Effect in the supply chain [3]. This paper presents an in-depth exploration of Black Box testing as a strategic tool to detect and minimize bugs in mini ERP systems, thereby contributing to the improvement of system quality and reliability, which is critical for business operational efficiency and effectiveness in today's competitive landscape [5].

Black Box testing is a software testing method that emphasizes testing the functional features of the software. This method does not consider the internal control structure of the software but instead focuses on the expected results of various input conditions. With Black Box testing, developers can specify a comprehensive set of input conditions to verify that all functional requirements of the program have been effectively met [6].

There are many kinds of testing techniques in Black Box testing, including:

a. Equivalence Partitioning: This technique divides input data into several partitions to ensure that test cases cover all possible scenarios by focusing on representative values.
b. Boundary Value Analysis: This technique involves testing at the boundaries of input values to identify faults that occur at the minimum or maximum limits.
c. Fuzzing: This technique finds bugs or glitches in software by injecting malformed or random data to uncover vulnerabilities.
d. Cause-Effect Graphing: This technique uses a graph to illustrate the relationship between causes and effects, helping to design test cases based on these relationships.
e. Orthogonal Array Testing: This technique is used when the input domain is relatively small but the complexity is high, employing a systematic approach to test a diverse set of input combinations.
f. All-Pairs Testing: This technique designs test cases to cover all possible discrete combinations of pairs of inputs, ensuring that every combination is tested at least once.
g. State Transition Testing: This technique is useful for testing the state machine and navigation conditions of a system, represented in a graphical form.
h. Error Guessing: This technique relies on the tester’s knowledge and experience to identify potential errors and design test cases to uncover those issues.
i. Use Case Testing: This technique tests each software function by simulating real-world scenarios from the initial stage to the final stage of the system’s operation.
j. Decision Table Testing: This technique systematically arranges various combinations of inputs in a table to test functions that involve logical relationships between two or more inputs.

System testing using the Black Box method aims to identify system weaknesses to ensure that the data generated matches the data entered during execution and to avoid deficiencies and errors before the application is used by end-users [7].

METHODS

RESEARCH DESIGN

Effective research methods for testing software require a systematic and structured approach. Thorough steps are needed to ensure that the testing process is performed correctly and produces accurate results [8]. The research method used in this study is

Black Box testing, which is one of the techniques in software testing. This approach focuses on the functional testing of software without regard to the internal structure of the code. In Black Box testing, the tester only has knowledge of the expected inputs and outputs of the software being tested, while the internal implementation details are not revealed [9]. The aim of this method is to evaluate the functions of the software and ensure that the application behaves according to predefined specifications. Thus, Black Box testing becomes an effective approach for validating the reliability and performance of software without requiring in-depth knowledge of its internal structure [10].

TEST DESCRIPTION

The five trading systems to be tested are:

a. System A: An online store for books.
b. System B: Mobile application for food ordering.
c. System C: Clothing sales website.
d. System D: Electronic sales platform.
e. System E: Desktop application for cinema ticket booking.

The four main black box testing techniques used in this research are:

a. Equivalence Partitioning (EP): Dividing input data into partitions that are considered to produce the same output.
b. Boundary Value Analysis (BVA): Tests the boundaries between valid and invalid input partitions.
c. Decision Table Testing (DTT): Uses decision tables to handle different combinations of input conditions.
d. Use Case Testing (UCT): Use Case is a test where we test each software function by running the system from the initial stage to the final stage.

APPLICATIONS USED TO TEST

Odoo is a suite of business management software, which includes CRM, e-commerce, billing, accounting, manufacturing, warehouse, project management, and stock management, among others [11]. The Community version is a free software, which is licensed under GNU LGPLv3 [12]. Some of the features to be tested will be customised to the needs of the system.

TESTING OBJECTIVES

The purpose of testing Odoo applications is to ensure that they function properly and meet the set quality standards. It also aims to validate functionality, detect bugs and flaws, ensure compatibility with various platforms, improve performance, user experience, prepare for delivery, as well as fulfil established business requirements[13]. Based on the test results, the Odoo application successfully fulfils all these objectives, showing that Odoo is a reliable and effective solution for the company's management needs [14].

RESEARCH DESIGN TABLE

1. Equivalence Partitioning

This table uses the Equivalence Partitioning Technique to ensure that System A: An online store for books. can Function properly.

Table 1. *Equivalence Partitioning* Technique testing model

Test ID	Features	Input	System Initial Condition	Expected Results
B01	*Website*	Opens the home page	Page does not open	Online shop home page opens
B02	*Website*	Search for books with keywords	Blank search page	List of relevant books displayed
B03	*E-commerce*	Add books to basket	Empty basket	Book added to basket
B04	*E-commerce*	Change the quantity of books in the basket	Books are in the basket	Quantity of books in the basket updated
B05	*E-commerce*	Checkout	Basket of books	Checkout page opens
B06	*Sales*	Generate invoice after checkout	Transaction not yet consummated	Sales invoice generated
B07	*Inventory*	Increase book stock	Book stock in the warehouse is empty	Book stock in the warehouse increased
B08	*Inventory*	Reducing book stock	Book stock in the warehouse	Book stock in the warehouse is decreasing
B09	*Payment Acquirers*	Paying for an order with a credit card	Order at checkout	Payment successful, order status updated
B10	*Payment Acquirers*	Paying for an order with PayPal	Order at checkout	Payment successful, order status updated
B11	*Coupons & Promotions*	Using discount coupons	Basket of books	Discounts are applied to the total price
B12	*Coupons & Promotions*	Using book count promotion	Basket of books	Discounts are applied according to promotional amounts
B13	CRM	Adding customer data	Empty customer data	New customer data added
B14	CRM	Change customer data	Customer data already exists	Customer data updated
B15	*Analytics*	View sales report	Sales data available	Sales report displayed
B16	Analytics	View visitor analytics	Visitor data available	Visitor analytics data displayed

B17	*Website*	Open the book category page	Empty category page	List of book categories displayed
B18	*Website*	View book details	Books available in the catalogue	Book details displayed
B19	*E-commerce*	Removing books from the basket	Books are in the basket	Books removed from basket
B20	*Payment Acquirers*	Payment failed	Order at checkout	Payment failed notification, order status not changed

With this table, various test scenarios have been compiled to ensure that each feature in the online bookstore system works as expected, based on the input provided by the user [9].

2. Boundary Value Analysis

This table uses the Boundary Value Analysis Technique to ensure that System A: Online store for books. can Function properly

Table 2. Model testing using the *Boundary Value Analysis* technique

Test ID	Testing Details	Expected results
D01	Add products to basket when stock is zero	Product cannot be added, out of stock notification
D02	Using discount coupons with a minimum transaction limit value	Discount applied according to coupon value
D03	Process payments with credit card maximum limit transaction amount	Payment processed successfully without error
D04	Search for books with maximum length keywords	Search results are displayed without errors
D05	Login with minimum username length	Successful login if username is valid, error if invalid
D06	Checkout with maximum number of products in the basket	Successful checkout,all products displayed on invoice

With the model used in table 2, various boundary testing scenarios have been set up to ensure that the online bookstore system works properly when facing certain critical or boundary conditions [15].

3. Decision Table Testing

This table uses the Decision Table Testing technique to ensure that System A: Online store for books. can function properly.

Table 3. Model testing using the *Decision* Table technique

Test ID	Condition	Rule 1	Rule 2	Rule 3	Rule 4
B01	Successful login	F	T	F	T
B02	Products available	F	F	T	T
B03	Successful payment	T	T	T	F
B04	Enough stock	F	F	T	T
B05	Valid coupon	T	F	F	T
B06	Delivery available	T	T	F	F

Description:

- T : Action succeeded
- F : Action failed

Interpretation:

- Rule 1: Login failed, product not available, payment successful, insufficient stock, coupon valid, shipping available.
- Rule 2: Login successful, product not available, payment successful, insufficient stock, invalid coupon, shipping available.
- Rule 3: Login failed, product available, payment successful, sufficient stock, invalid coupon, shipping not available.
- Rule 4: Login successful, product available, payment failed, sufficient stock, coupon valid, shipping available.

With this table, we can ensure various combinations of input conditions are tested to verify that the online store system for books functions correctly according to various possible scenarios [7].

4. Use Case Testing

This table uses the Use Case Testing technique to ensure that System A: Online store for books. can function properly.

Table 4. Testing model using *Use* Case technique

Test ID	Testing Details	Steps	Expected results
F01	Purchase books as a new user	1.Open an online store. 2.Select a book. 3.Add to basket. 4.Checkout. 5.Register as a new user.	User successfully purchases a book with a new account
F02	Using coupons discount	1.Open an online store. 2.Select a book. 3.Add to basket. 4.Enter the coupon code. 5.Checkout	Discount applied, total price reduced
F03	Checking order status	1.Login as a user. 2.Go to the "My Orders" page.	Order status is displayed correctly

		3.Select the latest order	
F04	Add book stock as admin	1.Login as admin. 2.Open the Inventory module. 3.Select the book you want to increase the stock of. 4.Add stock	Book stock was successfully updated in the system
F05	Manage book categories	1.Login as admin. 2.Open the E-commerce module. 3.Select "Book Category". 4.Add/edit categories	Book categories successfully managed
F06	View sales report	1.Login as admin. 2.Open the Analytics module. 3.Select "Sales Report". 4.Determine the time range.	Sales reports are displayed according to the time range

In Table 4, various usage scenarios have been compiled to ensure that the online bookstore system can handle various important operations correctly.Next, we will conduct a series of tests according to the design that has been made above.

RESULTS AND DISCUSSION

Testing is a crucial stage in software or system development, where various features and functionality are tested to ensure that the system functions properly in accordance with predetermined specifications. The following are the results of testing on System A: An online store for books.

Table 5.*Testing* results using *Equivalence Partitioning* technique

Test ID	Input Partition	Input	System Initial Condition	Expected Results	Testing results
B01	Website	Opens the home page	Page does not open	Online shop home page opens	Successful
B02	Website	Search for books with keywords	Blank search page	List of relevant books displayed	Successful
B03	Ecommerce	Add books to basket	Empty basket	Book added to basket	Successful
B04	Ecommerce	Change the quantity of books in the basket	Books are in the basket	Quantity of books in the basket updated	Successful
B05	Ecommerce	Checkout	Basket of books	Checkout page opens	Successful
B06	Sales	Generate invoice after checkout	Transaction not yet consummated	Sales invoice generated	Successful

B07	Inventory	Increase book stock	Book stock in the warehouse is empty	Book stock in the warehouse increased	Successful
B08	Inventory	Reducing book stock	Book stock in the warehouse	Book stock in the warehouse is decreasing	Successful
B09	Payment Acquirers	Paying for an order with a credit card	Order at checkout	Payment successful, order status updated	Successful
B10	Payment Acquirers	Paying for an order with PayPal	Order at checkout	Payment successful, order status updated	Successful
B11	Coupons & Promotions	Using discount coupons	Basket of books	Discounts are applied	Successful
B12	Coupons & Promotions	Using book count promotion	Basket of books	Discounts are applied according to promotional amounts	Successful
B13	CRM	Adding customer data	Blank customer data	New customer data added	Successful
B14	CRM	Change customer data	Customer data already exists	Customer data updated	Successful
B15	Analytics	View sales report	Sales data available	Sales report displayed	Failed
B16	Analytics	View visitor analytics	Visitor data available	Visitor analytics data displayed	Failed
B17	Website	Open the book category page	Empty category page	List of book categories displayed	Successful
B18	Website	View book details	Books available in the catalogue	Book details displayed	Successful
B19	Ecommerce	Removing books from the basket	Books are in the basket	Books removed from basket	Successful
B20	Payment Acquirers	Payment failed	Order at checkout	Payment failed notification , order status not changed	Successful

In Table 5, testing with the equivalence partitioning technique shows that almost all features of the Odoo application function properly, except for the analytics feature which still needs further improvement. The analytics feature fails to produce accurate reports and is inflexible in report customisation and has problems integrating data from various modules. It is recommended to improve and retest the analytics feature so that the entire system can function optimally

Table 6. Testing results using the technique. Boundary Value Analysis

Test ID	Testing Details	Expected results	Testing Results
D01	Add products to basket when stock is zero	Product cannot be added, out of stock notification	Successful
D02	Using discount coupons with a minimum transaction limit value	Discount applied according to coupon value	Successful
D03	Process payments with credit card maximum limit transaction amount	Payment processed successfully without error	Successful
D04	Search for books with maximum length keywords	Search results are displayed without errors	Successful
D05	Login with minimum username length	Successful login if username is valid, error if invalid	Successful
D06	Checkout with maximum number of products in the basket	Successful checkout, all products displayed on invoice	Successful

Table 6 shows that all features of the Odoo application function properly at the tested input boundaries, indicating that the system has been properly tested using the Boundary Value Analysis technique and all of its features run smoothly.

The results of the testing using the Decision Table Testing technique are summarized as follows. Based on the results detailed in the table above, the main conclusions are:

1. Key Functionality Works Well:
   - Tests to open the home page, search for books, add books to the basket, and checkout were successful without any problems.
   - Sales invoices are generated correctly after the checkout process. 2. Product and
2. Stock Management:
   - Addition and subtraction of book stock by the admin works well.
   - Out-of-stock books successfully provide appropriate notifications when users try to add them to the basket.
3. Payment Integration:
   - Credit card and PayPal payments were successfully processed correctly, demonstrating good integration with payment gateways.
   - The system also successfully handles failed payment cases and displays appropriate notifications.
4. Use of Coupons and Promotions:

- The use of discount coupons and book count promotions worked as expected, with discounts applied to the total price appropriately.

5. Customer Data Management and Analytics:
   - Adding and changing customer data can be done successfully through the CRM module.
   - Sales reports and visitor analytics can be accessed and displayed correctly, supporting business analysis needs.
6. System Performance and Validation:
   - The system successfully handles searches with maximum length keywords and login validation with minimum length username.
   - Checkout with the number of products in the basket reaching the maximum limit also succeeded without any problems.

Overall, the test results show that the Online Bookstore system using the Odoo application runs well and fulfils most of the expected functional requirements. Some critical aspects such as stock management, payment integration, promotion usage, and input validation have been tested and show satisfactory results. Except for the Analytics feature which should be developed further.

Table 7. Testing results using *Use Case* testing techniques

Test ID	Testing Details	Steps	Expected Results	Testing Results
F01	Purchase books as a new user	1. Open an online store. 2. Select a book. 3. Add to basket. 4. Checkout. 5. Register as a new user.	User successfully purchases a book with a new account	Successful
F02	Using discount coupons	1. Open an online store. 2. Select a book. 3. Add to basket. 4. Enter the coupon code. 5. Checkout	Discount applied, total price reduced	Successful
F03	Checking order status	1. Login as a user. 2. Go to the "My Orders" page. 3. Select the latest order	Order status is displayed correctly	Successful
F04	Add book stock as admin	1. Login as admin. 2. Open the Inventory module. 3. Select the book you want to increase the stock of. 4. Add stock	Book stock was successfully updated in the system	Successful

F05	Manage book categories	1. Login as admin. 2. Open the E-commerce module. 3. Select "Book Category". 4. Add/edit categories	Book categories successfully managed	Successful
F06	View sales report	1. Login as admin. 2. Open the Analytics module. 3. Select "Sales Report". 4. Determine the time range.	Sales reports are displayed according to the time range	Failed

The test results in Table 7 show that all features of the Odoo system run smoothly and in line with expectations, including project creation, task addition, invoice creation, payment management, job position creation, applicant management, employee data addition, attendance management, and performance evaluation. However, the analytics feature did not meet expectations. The use of the analytics feature to create reports and analyse data failed, indicating a problem with the feature that requires further improvement. Thus, although Odoo showed strong performance in most of the features, focusing on improving the analytics feature is essential to ensure reliability and accuracy in data analysis, so that it can meet the needs of users thoroughly.After performing the same testing model on the other four systems, we obtained the results in Table 8.

Table 8. Testing results on the other four systems

Tested System	Testing Results
System A: Online Shop for Books	There is an error in the Analytics feature section
System B: Mobile Application for Food Ordering	All features work well
System C: Clothing Sales Website	There is an error in the Analytics feature section
System D: Electronic Sales Platform	All features work well
System E: Desktop Application for Cinema Ticket Booking	All features work well

The test results show that all features in the system run smoothly and in accordance with expectations. However, the analytics features in systems A and C did not meet expectations and failed to produce correct and accurate reports at various input limits tested. This indicates that improvements are still needed in the analytics feature to ensure reliability and accuracy in data analysis.

CONCLUSION

Results Based on the results of testing the Odoo application which includes some of its features, it can be concluded that this application passed all tests very well. The project management feature proved to work perfectly. However, the tests also revealed that the analytics feature of the Odoo app failed. The analytics function that was

supposed to provide in-depth insights and reports on various operational aspects did not work as expected. These failures include the system's inability to generate accurate and timely reports, lack of flexibility in report customisation, and problems in data integration from various modules. For future research, it is recommended to focus on a few key areas to improve the performance of the analytics feature in Odoo:

- Analytics Module Development and Retesting: Carry out further development on the analytics module and test it intensively to ensure that all functionalities operate correctly and provide added value to users.
- Better Data Integration: Enhanced the process of integrating data from various modules so that the reports generated are more comprehensive and accurate.
- Improved Report Customisation: Added flexibility in report customisation so that users can easily tailor reports according to their specific needs.
- User Training: Provide adequate training to users to ensure that they can utilise analytics features optimally.
- User Feedback: Collect and analyse user feedback to continuously improve and refine analytics features based on user needs and expectations.

While Odoo shows strong performance in various aspects of business management, focusing on improving the Analytics feature will further increase the value and efficiency of this mini-ERP system for user companies. Overall, Black Box Testing proved to be effective in detecting and minimising bugs in most features of the mini ERP system. However, special attention needs to be paid to the analytics feature to improve the overall reliability and accuracy of the system. Recommendations for future research include a focus on improving the analytics module, better data integration, and increased report customisation to ensure that the mini ERP system can provide optimal added value for its users.

ACKNOWLEDGMENT

We would like to express our gratitude to everyone who has contributed to this research on the application of Black Box testing for mini ERP Systems. Our deepest appreciation goes to Mr. Muhammad Ainul Yaqin for providing the necessary resources and support. Special thanks to our colleagues and mentors for their invaluable insights and guidance throughout this study.We also extend our gratitude to the small and medium-sized companies that allowed us to test and validate our findings on their systems. Without their cooperation, this research would not have been possible.

Finally, we acknowledge the unwavering support from our families and friends, whose encouragement and understanding have been instrumental in the successful completion of this research.Thank you all for your contributions and support.

REFERENCES

[1] U. Sari and M. Megawaty, “Penerapan Usability Testing Untuk Pengukuran Kualitas Sistem Enterprise Resource Planning (ERP) (Studi Kasus: PT. TITIS SEMPURNAH PRABUMULIH),” Jurnal Nasional Ilmu Komputer, vol. 1, pp. 127–138, Aug. 2020, doi: https://doi.org/10.47747/jurnalnik.v1i3.159.

[2] P. Rahayu, V. Y. Tambunan, M. Agutina, W. Anastasya, D. Japin, and D. Melinda, “Penerapan Sistem ERP (Enterprise Resource Planning) Warehouse Management dalam Meningkatkan Kinerja Perusahaan,” Jurnal Ekonomi dan Bisnis, vol. 10, no. 2, pp. 241–245, 2022, doi: https://doi.org/10.34308/eqien.v10i2.590.

[3] T. S. Jaya, “Pengujian Aplikasi dengan Metode Blackbox Testing Boundary Value Analysis (Studi Kasus: Kantor Digital Politeknik Negeri Lampung),” Jurnal Informatika: Jurnal Pengembangan IT, vol. 3, no. 1, pp. 45–48, 2018, doi:

https://doi.org/10.30591/jpit.v3i1.647.
[4] D. Lutfiah, A. Ridwan, U. Hediyanto, and K. Kusumahastuti, “Pengembangan Sistem ERP Modul Inventory untuk Proses Penyimpanan Alat Medis pada Instalasi Kedokteran Nuklir RSHS dengan Metode Quickstart,” Jurnal Indonesia Manajemen Informatika dan Komunikasi, vol. 5, pp. 66–79, Jan. 2024, doi: https://doi.org/10.35870/jimik.v5i1.433.
[5] M. Syarif and E. B. Pratama, “Analisi Metode Pengujian Perangkat Lunak Blackbox Testing Dan Pemodelan Diagram UML Pada Aplikasi Veterinary Service Yang Dikembangkan Dengan Model Waterfall,” JTIK (Jurnal Teknik Informatika Kaputama), 2021, doi: https://doi.org/10.59697/jtik.v5i2.551.
[6] N. M. D. Febriyanti and A. Sudana, “Implementasi Black Box Testing pada Sistem Informasi Manajemen Dosen,” Jurnal Ilmiah Teknologi Dan Komputer,2021. doi: https://doi.org/10.24843/JTRTI.2021.v02.i03.p12.
[7] and H. H. I. Wahyudi, F. Fahrullah, F. Alameka, “Analisis Blackbox Testing Dan User Acceptance Testing Terhadap Sistem Informasi Solusimedsosku,” Jurnal Teknosains Kodepena, vol. 4, 2023, doi: https://doi.org/10.54423/jtk.v4i1.54.
[8] Z. Shu and G. Yan, “Iotinfer: Automated blackbox fuzz testing of iot network protocols guided by finite state machine inference,” IEEE Internet Things Journal, vol. 1, 2022, doi: https://doi.org/10.1109/JIOT.2022.3182589.
[9] Y. Wijaya and M. Astuti, “Pengujian Blackbox Sistem Informasi Penilaian Kinerja Karyawan PT INKA (PERSERO) Berbasis Equivalence Partitions,” Jurnal Digital Teknologi Informasi, vol. 4, p. 22, Mar. 2021, doi: https://doi.org/10.32502/digital.v4i1.3163.
[10] N. Suwirmayanti, I. Aryanto, and I. Putra, “Penerapan Helpdesk System dengan Pengujian Blackbox Testing,” Jurnal Ilmiah Intech Information Technology Journal of UMUS, 2020. doi: https://doi.org/10.46772/intech.v2i02.290.
[11] J. Y. Wu and L. T. Chen, “Odoo ERP with business intelligence tool for a small-medium enterprise: a scenario case study,” the 11th International Conference on E-Education, E-Business, E-Management, and E-Learning, 2020, doi: https://doi.org/10.1145/3377571.3377607.
[12] S. Supriyono and S. Sutiah, “Improvement of Project Management Using Accelerated SAP Method in the Odoo ERP,” Proceedings of the 1st International Conference on Management, Business, Applied Science, Engineering and Sustainability Development, 2020, doi: https://doi.org/10.4108/eai.3-8-2019.2290729.
[13] J. F. Marques and J. Bernardino, “Evaluation of Asana, Odoo, and ProjectLibre Project Management Tools using the OSSpal Methodology.,” 11th International Conference on Knowledge Engineering and Ontology Development, 2019. doi: https://doi.org/10.5220/0008351903970403.
[14] M. Satrio, A. Supena, and A. Nurrahman, “Penyelarasan Proses Bisnis Perusahaan dengan Sistem Enterprise Resource Planning Menggunakan Software Odoo,” Jurnal Riset Teknik Industri, pp. 139–146, Dec. 2023, doi: https://doi.org/10.29313/jrti.v3i2.2894.
[15] S. Supriyono, “Software testing with the approach of blackbox testing on the academic information system,” IJISTECH (International Journal of Information System and Technology), vol. 3, 2020, doi: https://doi.org/10.30645/ijistech.v3i2.54.