Whitebox Testing of A Restaurant Reservation System Using the Basis Path Technique

Surahmat[1*], Fakhri Aryva[2], Sony Oktapriandi[3], Ravie Kurnia Laday[4], Indra Satriadi[5]
[1,2,3,4,5] *Politeknik Negeri Sriwijaya*
* *surahmat@polsri.ac.id*

Abstract
Software testing is a crucial phase in the system development process to ensure that features and functions operate according to design specifications. This study aims to test a restaurant reservation system using the whitebox testing method with the basis path technique. The testing was conducted through the analysis of the Control Flow Graph (CFG), calculation of Cyclomatic Complexity, and identification of independent paths representing various reservation scenarios, ranging from successful bookings to failure conditions such as invalid input, unavailable tables, and incomplete customer data. The test results revealed seven paths derived from the basis path technique applied to the restaurant reservation system, all of which produced outcomes that matched expectations, with every test case passing successfully. This confirms that the system's logic functions properly and reliably across different user interactions. The application of the basis path technique proved effective in validating the system's control flow and enhancing reliability prior to deployment. This research contributes to the development of more stable systems and improved user experience

Keywords: Whitebox, Basis Path, Reservation System, Cyclomatic Complexity

Introduction

The software development process is inseparable from the testing phase, which serves to ensure that the functions and features of a system operate as intended according to the design. Software testing is also crucial for evaluating user responses to the features provided by the developed system, helping to validate data, ensure feature alignment, and deliver a positive user experience.

This study is conducted as a critical phase in the development of a restaurant reservation system (Egigogo et al., 2024), employing whitebox testing with the basis path technique. This approach ensures that the internal logic and code structure of the system align with the program's control flow, supported by comprehensive test cases(Golian et al., 2022).

The testing process involves several stages, starting from analyzing the Control Flow Graph (CFG), calculating Cyclomatic Complexity, and identifying the basis paths used as part of the testing framework. Previous studies on similar testing approaches include research titled "Testing of a Web-Based Library Information System Using White Box Testing", which concluded that testing provides a deeper understanding of the system implementation phase (Rosyada et al., 2022).

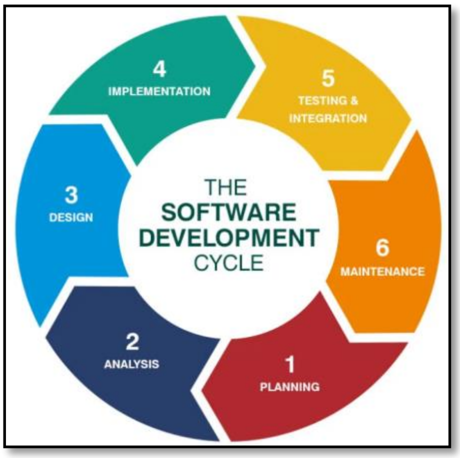
Through this research, the author aims to demonstrate how system testing can significantly enhance system reliability and improve the overall user experience. By conducting verification and validation before the system is deployed, users are more likely to have a smooth and satisfying interaction with the application

Literature Review

Software testing is a critical phase in the software development methodology. One commonly used approach is whitebox testing with the basis path technique, which aims to validate the structure of the code and assess system functionality based on the logical paths defined within the program. This method helps to trace the flow of logic in the developed system and ensure its correctness.

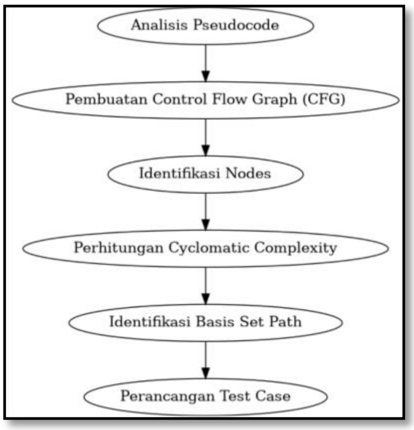
Several previous studies have explored the application of whitebox testing. For instance, the study titled "Testing the Wonokeling Village Information System Using the Whitebox Testing Method" examined whether the developed system contained bugs or errors (Paramanandi et al., 2023). Another study, "Whitebox Testing on the Android-Based Debt Manager Application", discussed the use of whitebox testing to evaluate the functionality of an Android system (Gusdevi et al., 2022).
From these studies, a research gap is identified in the application of basis path testing (Nugraha, 2022), particularly in terms of user interaction and the complexity of path selection within the code. Therefore, this research aims to address that gap by applying whitebox testing to several prepared test modules, ensuring that the logical paths within the code are properly validated.

Methods
The software development method used in this study follows the Software Development Life Cycle (SDLC) (Ganjarrintana et al., 2024), which consists of six stages: analysis, system design, implementation (coding), testing, deployment, and maintenance(Ervanisari et al., 2024).



Figures 1. Software Development Life Cycle (SDLC) Method

For the system testing method, this study employs whitebox testing with a structured sequence of steps(Pratiwi & Widianti, 2025). This method was chosen because whitebox testing using the basis path technique provides a systematic testing approach that covers the entire codebase of the system (Munaiseche & Rorimpandey, 2021)(Solissa et al., 2023). It enables the identification of logical paths within the code (Khamaeni, 2023), helping to reduce the occurrence of errors or bugs. Furthermore, this testing method is considered suitable for evaluating the restaurant reservation system developed in this research.



Figures 2. Whitebox Testing

Results and Discussion

The interface of the restaurant reservation system being tested is as follows, focusing on the table reservation menu used by customers.

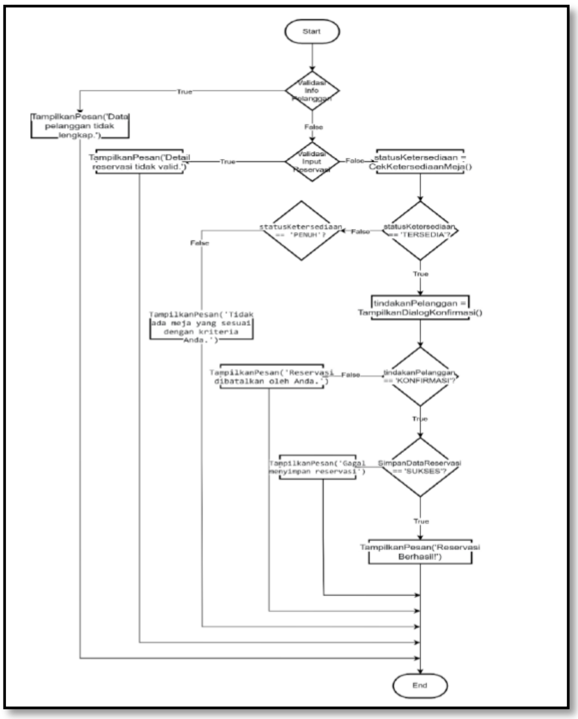
Figures 3. Application Interface

The results of the research conducted in this study are described through several stages, in accordance with the testing phases previously explained. At this stage, system testing is first carried out on the customer table reservation process by explaining the pseudocode of the 'LakukanReservasiMeja' function with the following code.

```
1. MULAI
2. IF ValidasiInfoPelanggan(userInput.infoPelanggan) ==
FALSE THEN
3. TampilkanPesan("Data pelanggan tidak lengkap atau tidak
valid.")
4. ELSE
5. IF ValidasiInputReservasi(userInput.tanggal,
userInput.jam, userInput.jumlahOrang) == FALSE THEN
6. TampilkanPesan("Detail reservasi (tanggal, jam, atau
jumlah orang) tidak valid.")
7. ELSE
8. statusKetersediaan =
SistemCekKetersediaanMeja(userInput.tanggal, userInput.jam,
userInput.jumlahOrang)
9. IF statusKetersediaan == "TERSEDIA" THEN
10. tindakanPelanggan = TampilkanDialogKonfirmasiReservasi()
11. IF tindakanPelanggan == "KONFIRMASI" THEN
12. IF SimpanDataReservasiKeDatabase(userInput) == "SUKSES"
THEN
13. TampilkanPesan("Reservasi Anda berhasil dikonfirmasi!")
14. ELSE
15. TampilkanPesan("Maaf, terjadi kesalahan saat menyimpan
reservasi Anda. Silakan coba lagi.")
16. END IF
17. ELSE
18. TampilkanPesan("Reservasi dibatalkan oleh Anda.")
19. END IF
20. ELSE IF statusKetersediaan == "PENUH" THEN
21. TampilkanPesan("Maaf, restoran sudah penuh pada tanggal
dan jam yang Anda pilih.")
22. ELSE
23. TampilkanPesan("Maaf, tidak ada meja yang tersedia
sesuai dengan kriteria Anda.") 24. END IF
25. END IF
26. END IF
27. SELESAI
END PROCEDURE
```

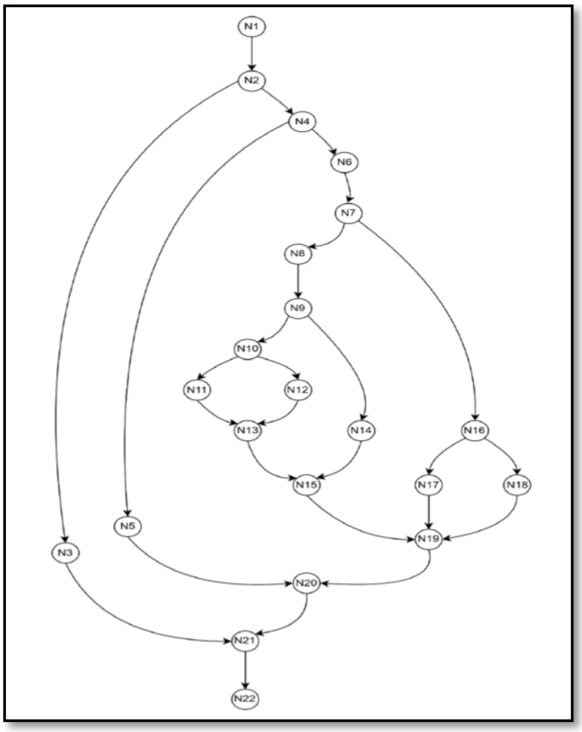
Figures 4. Pseudocode LakukanReservasiMeja

From that stage, the mapping of nodes is then carried out, followed by the creation of a diagram using a flowchart, as shown below:



Figures 5. Flowchat LakukanReservasiMeja

After that, a Control Flow Graph (CFG) is created



Figures 6. Control Flow Graph LakukanReservasiMeja

Cyclomatic Complexity (V(G)) is a software metric used to measure the logical complexity of a program or function. The value of V(G) provides an upper bound on the number of test cases required for Basis Path Testing, meaning that V(G) indicates the number of linearly independent paths in the Control Flow Graph (CFG).

To calculate the Cyclomatic Complexity (V(G)) of the LakukanReservasiMeja function, a method based on the number of Predicate Nodes (P) in the CFG is used. A Predicate Node is a node that has more than one outgoing path (a decision point)

The formula used is:

•V(G) = P + 1

Based on the analysis of the Control Flow Graph (CFG) for the LakukanReservasiMeja function

•The number of Predicate Nodes (P) identified is 6.

•These Predicate Nodes are N2, N4, N7, N9, N10, and N16.

Therefore, the Cyclomatic Complexity (V(G)) calculation is as follows:

•V(G) = 6 + 1

•V(G) = 7

A Cyclomatic Complexity (V(G)) value of 7 indicates that there are 7 independent paths within the LakukanReservasiMeja function. This number serves as the basis for identifying the set of basis paths to be used in designing test cases to achieve Basis Path coverage. After the Cyclomatic Complexity value is obtained, the process continues with the identification of each basis path, consisting of seven sets as follow

Path 1:

N1 → N2(F) → N4(F) → N6 → N7(T) → N8 → N9(T) → N10(T) → N11 → N13 → N15 → N19 → N20 → N21 → N22

Scenario Description: Customer information is valid, reservation input is valid, table is available, customer confirms, data saving is successful.
Path 2:
N1 → N2(F) → N4(F) → N6 → N7(T) → N8 → N9(T) → N10(F) → N12 → N13 → N15 → N19 → N20 → N21 → N22
Scenario Description: Customer information is valid, reservation input is valid, table is available, customer confirms, but data saving fails.
Path 3:
N1 → N2(F) → N4(F) → N6 → N7(T) → N8 → N9(F) → N14 → N15 → N19 → N20 → N21 → N22
Scenario Description: Customer information is valid, reservation input is valid, table is available, but customer cancels confirmation.
Path 4:
N1 → N2(F) → N4(F) → N6 → N7(F) → N16(T) → N17 → N19 → N20 → N21 → N22
Scenario Description: Customer information is valid, reservation input is valid, but table is not available due to full restaurant.
Path 5:
N1 → N2(F) → N4(F) → N6 → N7(F) → N16(F) → N18 → N19 → N20 → N21 → N22
Scenario Description: Customer information is valid, reservation input is valid, but table is not available due to other reasons (e.g., capacity mismatch).
Path 6:
N1 → N2(F) → N4(T) → N5 → N20 → N21 → N22
Scenario Description: Customer information is valid, but reservation details input is invalid.
Path 7:
N1 → N2(T) → N3 → N21 → N22
Scenario Description: Customer information is invalid or incomplete.

the final stage of testing, test case designs are created and testing is carried out for each path that has been previously identified

Table 1. Test Case

Test Case ID	Test Path	Expected Result	Actual Result	Pass/Fail
001	Path 1	A confirmation message stating "Your reservation has been successfully confirmed!" is displayed to the user. The reservation data is then successfully stored in the database,	The confirmation message was displayed as expected, and the reservation data was successfully saved to the database	Pass
002	Path 2	A message stating "Sorry, an error occurred while saving your reservation. Please try again." is displayed to the user. The reservation data was not saved to the database	The error message was displayed correctly, and the reservation data was not saved.	Pass

003	Path 3	A message stating "Reservation has been canceled by you." is displayed to the user. The reservation data was not saved to the database.	The cancellation message appeared as expected, and no data was saved.	Pass
004	Path 4	A message stating "Sorry, the restaurant is fully booked on the date and time you selected." is displayed.	The system displayed the full booking message correctly.	Pass
005	Path 5	A message stating "Sorry, no tables are available that match your criteria." is displayed.	The system displayed the unavailability message as expected.	Pass
006	Path 6	A message stating "The reservation details (date, time, or number of people) are invalid." is displayed.	The system correctly displayed the invalid input message.	Pass
007	Path 7	"Customer data is incomplete or invalid." is displayed.	The system displayed the message about incomplete or invalid customer data as expected.	Pass

Conclusion

This study successfully applied whitebox testing using the basis path technique to evaluate the internal logic of a restaurant reservation system. Through the construction of a Control Flow Graph (CFG) and the calculation of Cyclomatic Complexity, seven independent paths were identified and tested. Each path represented a unique scenario in the reservation process, ranging from successful bookings to various failure conditions such as invalid input, unavailable tables, and incomplete customer data.

The results of the testing showed that all expected outcomes matched the actual system behavior, with each test case passing successfully. This confirms that the system's logic is functioning correctly and reliably across different user interactions. The use of basis path testing proved effective in validating the system's control flow and ensuring robustness before deployment.Overall, the research demonstrates that structured whitebox testing enhances system reliability, reduces potential errors, and contributes to a better user experience.

References

Egigogo, R. A., Naniya, M. T., Abubakar, A. A., & Mansir, A. (2024). Design and Implementation of Computerized Restaurant Table Booking System. *Ceddi Journal of Information System and Technology (JST)*, *3*(1), 47–55.

Ervanisari, Y. P., Koyimatu, M., Simanjuntak, K. A., & Oktafiani, I. (2024). Penerapan Metode SDLC Waterfall Pada Sistem Pemesanan Makanan Menggunakan QR-Code Berbasis Website. *Prosiding SISFOTEK*, *8*(1), 38–43.

Ganjarrintana, A. R., Yusuf, A. I., Vernanda, D., & Herdiawan, T. (2024). Penerapan SDLC Agile dalam Pembuatan SIRUMA (Sistem Informasi Rumah Makan). *Inventor: Jurnal Inovasi Dan Tren Pendidikan Teknologi Informasi*, *2*(3), 116–124.

Golian, N., Golian, V., & Afanasieva, I. (2022). Black and white-box unit testing for web applications. *Bulletin of National Technical University" KhPI". Series: System Analysis, Control and Information Technologies*, *1 (7)*, 79–83.

Gusdevi, H., Kuswayati, S., Iqbal, M., Bakar, M. F. A., Novianti, N., & Ramadan, R. (2022). Pengujian White-Box Pada Aplikasi Debt Manager Berbasis Android. *Naratif: Jurnal Nasional Riset, Aplikasi Dan Teknik Informatika*, *4*(1), 11–22.

Khamaeni, M. G. A. L. (2023). Implementasi White Box Testing Berbasis Path Pada Aplikasi Berbasis Web. *Jurnal Siliwangi Seri Sains Dan Teknologi*, *9*(1).

Munaiseche, C. P. C., & Rorimpandey, G. C. (2021). Penerapan Metode Basis Path Analysis dalam Pengujian White Box Sistem Pakar. *Prosiding SISFOTEK*, *5*(1), 124–128.

Nugraha, W. A. (2022). Pengujian White Box Berbasis Path Pada Form Autentikasi Berbasis Mobile. *Jurnal Siliwangi Seri Sains Dan Teknologi*, *8*(2).

Paramanandi, H. S., Novianto, R. E., & Permatasari, H. (2023). Pengujian Sistem Informasi Desa Wonokeling Menggunakan Metode Whitebox Testing. *Prosiding Seminar Nasional Teknologi Informasi Dan Bisnis*, 507–511.

Pratiwi, Y., & Widianti, L. W. (2025). IMPLEMENTASI WHITEBOX TESTING DENGAN TEKNIK BASIS PATH PADA PENGUJIAN HALAMAN PENCARIAN PROGRAM PROMO. *Jurnal Kecerdasan Buatan Dan Teknologi Informasi*, *4*(2), 173–180.

Rosyada, A. F., Sukirman, I., Sukirman, I., Nur, M. A., Kristiawan, K., & Saifudin, A. (2022). Pengujian Sistem Informasi Aplikasi Perpustakaan basis Website Menggunakan White Box Testing. *BULLET: Jurnal Multidisiplin Ilmu*, *1*(06), 1034–1039.

Solissa, Y. J., Putra, F., Putri, A. N., & Nursari, S. R. C. (2023). Pengujian White Box Berbasis Path pada Form Daftar Jobstreet. co. id. *KONSTELASI: Konvergensi Teknologi Dan Sistem Informasi*, *3*(2), 353–362.