Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 10, Number 3, July 2026
DOI : https://doi.org/10.33395/sinkron.v10i3.16079
e-ISSN : 2541-2019
p-ISSN : 2541-044X

Comparative Evaluation of Feature-Driven Development and Scrum in Public Facility Booking Systems

Wahdini[1)*], Hilyah Magdalena[2)]
[1,2)]Information Systems Study Program ISB Atma Luhur, Indonesia
[1)]2222500005@mahasiswa.atmaluhur.ac.id, [2)]hilyah@atmaluhur.ac.id

Submitted :April 11, 2026 | **Accepted** : May 2, 2026 | **Published** : July 5, 2026

Abstract: The Parittiga Sub-District Office still relies on manual procedures for multipurpose building borrowing, resulting in limited schedule transparency, double booking vulnerabilities, and delayed confirmations. However, a critical research gap exists: no quantitative comparison between Feature-Driven Development (FDD) and Scrum has been conducted for public facility booking systems. To address this, a web-based booking system was developed using FDD and empirically compared with Scrum through a comparative experimental design. Using identical two-developer teams, both methods were evaluated across five metrics: development time per feature, functional error rate (black-box testing), developer understanding (1-5 scale), usability (SUS), and end-to-end borrowing efficiency. Findings demonstrate that FDD outperforms Scrum with 50% fewer functional errors (2 vs 4), higher developer understanding (4.3 vs 3.8), and better usability (84.2 - Excellent vs 71.6 - Good), despite requiring 2 additional hours per feature (12 vs 10). Moreover, the FDD-based system accelerates the complete borrowing process by 63% (4 vs 11 minutes). This research contributes empirical evidence to the agile methodology literature and offers a structured, replicable evaluation framework for practitioners selecting development methods for stable, documentation-intensive public service environments.

Keywords: comparative analysis, feature-driven development, public service system, scrum, system booking

INTRODUCTION

Digital transformation has become a strategic imperative for sub-district government agencies worldwide to improve public service efficiency and quality. Web-based systems enable wider information dissemination and digital service delivery, fundamentally reshaping how citizens interact with government institutions (Dan et al., 2022) (Herwindiati et al., 2023).

However, the Parittiga Sub-District Office still manages borrowing of multipurpose buildings through manual procedures, using paper forms and physical archives. The office provides a Multipurpose Hall (GSG) to support official activities and community events (Kecamatan Parittiga, 2026). This manual practice triggers three operational obstacles: limited schedule transparency, high risk of double booking, and slow confirmation and reporting processes (Ramadani et al., 2025).

Agile software development methods have been widely adopted to address such operational inefficiencies. Feature-Driven Development (FDD) prioritizes iterative development, adaptive to system functionality progress (Putri et al., 2025), ensuring systematic feature optimization aligned with public service operational needs (Rekayasa et al., 2025). Conversely, Scrum is more popular for small-scale projects with high requirement volatility, while FDD offers advantages in feature documentation and progress tracking (Rizky & Sugiarti, 2022).

Despite the popularity of both methods, a critical research gap exists: no empirical study has quantitatively compared FDD and Scrum for web-based public facility booking systems. Most previous research has focused on implementing a single method without comparative analysis, particularly in government service contexts where documentation accuracy and transparency are paramount (Kusnadi et al., 2024) .

Therefore, this study has three objectives: (1) to design and build a multipurpose building booking system using FDD, (2) to quantitatively compare FDD versus Scrum based on development time per feature, number of

*name of corresponding author

e-ISSN : 2541-2019
p-ISSN : 2541-044X

functional errors, and developer understanding level, and (3) to evaluate the usability and efficiency of the resulting system using standardized metrics including the System Usability Scale (SUS).

To support this development, a web-based booking system was designed using the FDD method. Although FDD has been widely applied in various software projects, its application for public facility coordination at the sub-district level presents specific challenges regarding data volatility and high transparency demands (Fatmawati et al., 2025). The system design is mapped using Unified Modeling Language (UML) to ensure a robust and scalable architecture (Narulita et al., 2024), including use case diagrams to represent system requirements (Iscte-iul, 2023) and class diagrams to depict the static structure (Cheikh & Tebessi, 2024).

LITERATURE REVIEW

Previous studies consistently highlight that Feature-Driven Development (FDD) excels in structured, documentation-oriented environments, while Scrum is more adaptive to dynamic requirements with frequent changes. This fundamental distinction has been validated across multiple application domains, yet a critical synthesis specifically for public facility booking systems remains absent (Rizky & Sugiarti, 2022).

In the domain of information system development, Kusnadi et al. (2024) demonstrated that FDD's feature-oriented stages from requirement identification to implementation ensure process transparency and system quality for a website-based job search system. Similarly, WP (2025) applied FDD to village asset monitoring and inventory management, finding improvements in efficiency and accuracy, although feature integration for comprehensive governance remained incomplete. Despite these successes, both studies focused exclusively on FDD implementation without comparative analysis against other Agile methods, leaving uncertainty about whether FDD's advantages are method-specific or context-dependent (Kusnadi et al., 2024).

Research into room and facility booking systems has revealed consistent problems with manual approaches: conflicting schedules, delayed confirmations, and a lack of transparency (Firdonsyah & Putri, 2025). Technology-based solutions, including interactive calendars and automatic notifications, have proven effective in reducing scheduling conflicts. However, most booking system studies have emphasized functional outcomes rather than the software development methodology used to achieve them, creating a gap in understanding how different Agile methods influence system quality and usability.

International studies provide stronger evidence for methodological comparisons (Samuel Gbli Tetteh, 2024)Conducted a systematic literature review of FDD and found that its "design by feature" phase produces more maintainable documentation compared to Scrum's informal artifacts, particularly in regulatory environments. A comparative analysis (Alsaadi, 2019) Investigating XP, Scrum, and FDD against European Medical Device Regulation (MDR) requirements revealed that FDD is the closest agile practice to satisfy stringent regulatory demands. The study found that FDD's 'develop overall model' phase provides semi-fixed up-front planning and produces more documentation (including UML modeling) for traceability purposes compared to Scrum, which lacks sufficient documentation for regulatory compliance.

Previous research results indicate that the implementation of Feature-Driven Development in asset monitoring and inventory management systems can improve efficiency and accuracy, but there are still shortcomings in the integration of features that support comprehensive village asset governance. Based on these findings, this study aims to present a system that is more transparent and accountable, while also meeting functional and non-functional specifications in supporting information technology-based village asset management (WP, 2025).

A critical synthesis of these findings reveals two key insights. First, the primary strength of FDD lies in its structured documentation and feature traceability, which reduces functional errors and improves usability for systems with stable, well-defined requirements. Second, Scrum's advantage in development speed comes at the cost of documentation quality and feature completeness, as evidenced by missing validation and report features in comparative studies (Pambudi & Dwi, 2020). However, no previous study has applied this comparative framework specifically to public facility booking systems, where documentation accuracy and transparency are paramount for government accountability.

Research Gap and Positioning of This Study

Based on the literature review, a comparison of previous studies is presented in Table 1 below:

Table 1. Comparison of Previous Studies with This Research

Researcher	Method	Research Focus	Shortage	The Gap Addressed by This Research
Kusnadi et al. (2024)	Feature-Driven Development	Website-based job search information system	Focus only on the implementation of a single feature, no	Conducting a comparative analysis between Feature-Driven Development and Scrum

*name of corresponding author

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 10, Number 3, July 2026
DOI : https://doi.org/10.33395/sinkron.v10i3.16079
e-ISSN : 2541-2019
p-ISSN : 2541-044X

			comparative analysis of methods.	to examine the efficiency of the methodology.
Firdonsyah & Putri (2025)	Agile & Extreme Programming	Public room booking system, interactive calendar & notifications	Not using the Feature-Driven Development method lack of deep feature traceability.	Implementing Feature-Driven Development that excels in feature documentation and traceability compared to common Agile methods.
WP (2025)	Feature-Driven Development	Village asset monitoring and inventory management	Lack of integration of comprehensive governance features, and no system usability evaluation yet.	Integration of transparent public building booking features with usability evaluation using SUS.
Pambudi & Dwi (2020)	Feature-Driven Development vs Scrum	A general comparison of Agile methods in software engineering.	Theoretical in nature, no implementation in a real system (specific case study)	Real implementation on a public building booking system, accompanied by testing the effectiveness of the methods.
This research	Feature-Driven Development + Comparative Analysis of Scrum	Building a booking system + Comparative analysis of methods	-	Combining the design efficiency of Feature-Driven Development with a comparative evaluation of Scrum and SUS testing.

The novelty of this research lies in: (1) the application of Feature-Driven Development to a public facility booking system at the sub-district level, (2) a quantitative comparative analysis between Feature-Driven Development and Scrum using empirical metrics, (3) the use of standardized usability testing (SUS), and (4) a systematic evaluation framework that includes black-box, usability, and efficiency testing.

METHOD

This study employs a quantitative experimental approach combined with Design Science Research (DSR) methodology (Peffers et al., 2018). The DSR framework was chosen because it enables both the construction of an IT artifact (the web-based booking system) and its systematic evaluation using measurable, quantitative metrics. Unlike a purely qualitative descriptive approach, this study collects and analyzes numerical data on development time, error rates, developer understanding, usability scores (SUS), and process efficiency. The following stages of the research can be seen in Figure 1.

Fig 1. Research Stages

Problem Identification

The process of borrowing the multipurpose building at the Parittiga Subdistrict Office currently still relies on a manual process through paper forms and physical archive recording. This situation triggers several operational obstacles, including vulnerability to the loss of documentation and difficulty in searching for old archives.

Data Collection

At this stage, data acquisition is carried out to create a web design for borrowing the multipurpose building at the Parittiga Subdistrict Office as follows:

Observation

Researchers directly observe the business process mechanisms that occur in the borrowing of the multipurpose building of Parittiga District.

Interview

Researchers gather information through interactions with relevant parties.

Documentation

*name of corresponding author

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 10, Number 3, July 2026
DOI : https://doi.org/10.33395/sinkron.v10i3.16079
e-ISSN : 2541-2019
p-ISSN : 2541-044X

The author collects files such as building loan request letters, receipts, and activity reports.

Literature Review

The literature review is carried out by understanding and tracing books, articles, and sources related to the flow of the research subject.

Feature-Driven Development

The Feature-Driven Development method is a development framework organized systematically to produce work output through an iterative process within a specified time period, and has measurable success indicators (Romero et al., 2023). The Feature-Driven Development method applies an approach that is oriented towards system development through procedures that are intuitive and practical to use. In addition, offers problem-solving techniques as well as transparent and easy-to-understand reporting mechanisms (Kusnadi et al., 2024).

Fig 2. Stages of the *Feature-Driven Development* Method

Develop an Overall Model

The first step in the Feature-Driven Development method is very important to understand how the system works as a whole. Essentially, it involves creating an initial design that aligns with the user's desires after gathering complete information (Putra Rakhmadani et al., 2022).

Build a Feature List

This stage is carried out to transform the conceptual model into a list of system features that is more specific and measurable (WP, 2025).

Plan by Feature

The next step is the preparation of a website design strategy that is systematic and organized (Rasyad et al., 2025).

Design by Feature

The design stages are implemented based on features, where the plans that have been prepared begin to be realized in visual form. The main focus is on designing the interface details for each functionality that has been previously designed (Dema & Simanjuntak, 2025).

Build by Feature

This stage can be simply described as the process of turning feature designs into actual functioning code (Putri et al., 2025).

Comparative Analysis between Feature-Driven Development and Scrum

To address the novelty gap and improve methodological rigor, this study conducted a comparative experiment between Feature-Driven Development and Scrum. Two core features of the booking system (online borrowing form and interactive calendar) were developed using both approaches in a simulated environment. The development team consisted of the same two developers to eliminate skill bias. The comparison was based on three metrics:

Table 2. Experimental Design for Feature-Driven Development vs Scrum Comparison

Metric	Measurement Method	Instrument
Development time per feature	Time tracking per feature completion	Timesheet (hours per person)
Number of functional errors	Black-box test case failures	10 test cases per feature
Developer understanding level	Post-development questionnaire (1–5 scale)	5-item Likert questionnaire

Measurable Evaluation Metrics

This study established four measurable evaluation metrics that can be replicated by other researchers:

*name of corresponding author

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 10, Number 3, July 2026
DOI : https://doi.org/10.33395/sinkron.v10i3.16079
e-ISSN : 2541-2019
p-ISSN : 2541-044X

Table 3. Evaluation Metrics Used in This Study

No	Metric	Operational Definition	Measurement Method	Target
M1	Development Time per Feature	Total time spent implementing one feature	Time tracking	< 15 hours
M2	Number of Functional Errors	Number of failures during black-box testing	10 test scenarios	< 3 errors
M3	Developer Understanding Level	Developer perception score (1–5 scale)	5-item questionnaire	> 4.0
M4	Borrowing Process Efficiency	Time from opening system to confirmation	End-to-end simulation	< 5 minutes

Usability Measurement with the System Usability Scale (SUS)

Understanding System Usability Scale (SUS)

System Usability Scale (SUS) is a questionnaire developed by John Brooke in 1986 to measure the usability level of a system quickly, reliably, and validly. SUS has been widely used in information system research because it has high reliability (Cronbach's alpha > 0.85) and only requires 10 statements with 5 answer choices (Mardi Suryanto et al., 2022).

Instrument SUS (10 Questions)

Usability data collection was conducted through the distribution of SUS questionnaires to 5 respondents, consisting of employees of the Parittiga Subdistrict Office and the general public as potential users. Although the sample size is small (n=5), this study follows the established SUS guideline that reliable usability insights can be obtained from as few as 5 respondents, particularly for initial comparative evaluations (Lewis, 2018; Sauro & Lewis, 2016). Testing was carried out after the respondents tried the building loan system prototype. The questionnaire consisted of 10 standard SUS statements presented in Indonesian to facilitate respondents' understanding. The collected data were then processed using standard SUS scoring techniques to obtain the final score. Here are the 10 standard SUS statements used in this study with a Likert scale from 1 (Strongly Disagree) to 5 (Strongly Agree):

Table 4. Instrumen System Usability Scale (SUS)

No	Pernyataan (Statement)	Jenis	Skala
1	Saya berpikir akan menggunakan sistem ini lagi (I think that I would like to use this system frequently)	Positif	1 2 3 4 5
2	Saya merasa sistem ini rumit untuk digunakan (I found the system unnecessarily complex)	Negatif	1 2 3 4 5
3	Saya merasa sistem ini mudah digunakan (I thought the system was easy to use)	Positif	1 2 3 4 5
4	Saya membutuhkan bantuan teknisi untuk menggunakan sistem ini (I think that I would need the support of a technical person to be able to use this system)	Negatif	1 2 3 4 5
5	Saya merasa fitur-fitur sistem ini terintegrasi dengan baik (I found the various functions in this system were well integrated)	Positif	1 2 3 4 5
6	Saya merasa terlalu banyak inkonsistensi dalam sistem ini (I thought there was too much inconsistency in this system)	Negatif	1 2 3 4 5
7	Saya bayangkan orang lain akan cepat memahami sistem ini (I would imagine that most people would learn to use this system very quickly)	Positif	1 2 3 4 5
8	Saya merasa sistem ini membingungkan untuk digunakan (I found the system very cumbersome to use)	Negatif	1 2 3 4 5
9	Saya merasa percaya diri menggunakan sistem ini (I felt very confident using the system)	Positif	1 2 3 4 5
10	Saya perlu belajar banyak sebelum menggunakan sistem ini (I needed to learn a lot of things before I could get going with this system)	Negatif	1 2 3 4 5

Formulas and Calculation Formulas of SUS

The SUS score calculation is carried out in three systematic steps as follows:

Step 1: Convert Scores for Each Statement

The conversion formula differs for positive and negative statements:

Sus Score Conversion Formula

For Positive Statements (numbers 1, 3, 5, 7, 9):

Contribution Score = (Answer Value - 1)

*name of corresponding author

Where: if answered 5 → score = 4; 4 → 3; 3 → 2; 2 → 1; 1 → 0

For Negative Statements (numbers 2, 4, 6, 8, 10):

Contribution Score = (5 - Answer Value)

Where: if answered 5 → score = 0; 4 → 1; 3 → 2; 2 → 3; 1 → 4

Step 2: Sum All Contribution Scores

Total Raw Score = Σ Contribution Scores (Statements 1 to 10)

Maximum score = 40 (if all answers are 5 on positive statements and 1 on negative statements)
Minimum score = 0 (if all answers are 1 on positive statements and 5 on negative statements)

Step 3: Convert to SUS Scale (0-100)

SUS Score = Total Raw Score × 2.5

Mathematically, the SUS formula can be written as:

$$SUS = 2.5 \times (\Sigma(Pi - 1) + \Sigma(5 - Nj))$$

Where:
- Pi = score for the i-th positive statement (i = 1,3,5,7,9)
- Nj = score for the j-th negative statement (j = 2,4,6,8,10)

Interpretation of SUS Score

Based on the guidelines developed, the SUS score is interpreted as follows:
≥80 = Excellent
70-79 = Good
60-69 = OK

SUS Average Formula for Multiple Respondents

If the study involves n respondents, then the average SUS score is calculated using the formula:

$$Average\ SUS = (\Sigma\ SUSj) / n$$

Where:
- n = number of respondents
- SUSj = SUS score of the j-th respondent

Statistical Analysis (Independent t-test)

To determine whether the observed differences between FDD and Scrum are statistically significant, an independent two-sample t-test was conducted for each metric (development time, functional errors, developer understanding, and SUS score). The t-test assumes equal variance between the two development approaches. Statistical significance was set at $\alpha = 0.05$ (95% confidence level). Effect size (Cohen's d) was also calculated to measure the magnitude of differences, with interpretations: $d < 0.2$ (negligible), $0.2 \leq d < 0.5$ (small), $0.5 \leq d < 0.8$ (medium), and $d \geq 0.8$ (large) (Cohen, 1988).

A Systematic Testing Approach

In addition to functional black-box testing, this study also applied a more systematic testing approach as follows:

Table 5. Types of Testing Conducted

No	Testing Type	Purpose	Number of Test Cases	Instrument
1	Black-box Testing	Verify functional conformity	10 scenarios per feature	Test case checklist
2	Usability Testing	Measure ease of use	5 respondents	SUS Questionnaire
3	Efficiency Testing	Measure response time	3 repetitions per scenario	Stopwatch
4	Integration Testing	Ensure features work together	5 end-to-end scenarios	Structured scenarios

*name of corresponding author

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 10, Number 3, July 2026
DOI : https://doi.org/10.33395/sinkron.v10i3.16079
e-ISSN : 2541-2019
p-ISSN : 2541-044X

5	Regression Testing	Ensure new features don't break existing ones	All features (8 features)	Automated test script

RESULT

Develop an Overall Model Result

The initial phase involved developing a conceptual model of the system. This model was developed through an analysis of user requirements (borrowers and the General Affairs & Equipment Department) and an observation of the existing building booking process. The outcome was a use case diagram illustrating the relationships between actors and the system, as well as a class diagram showing the data structure and relationships between entities.

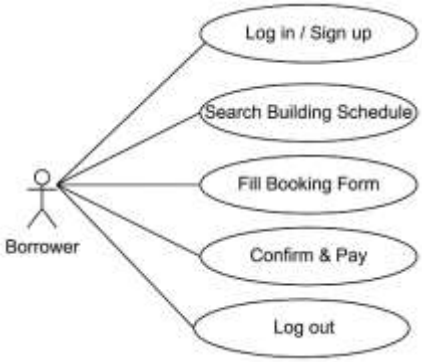

Fig 3. Borrower Use Case Diagram

Fig 4. Use Case Diagram General Parts & Equipment

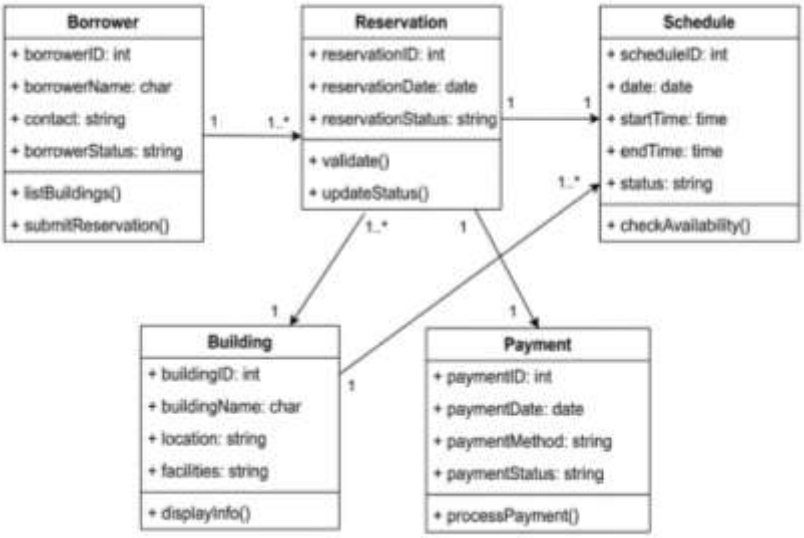

Fig 5. Class Diagram

Build a Feature List Result

The main features produced are login and registration for borrower accounts, an online multipurpose building borrowing form, availability calendar, validation and confirmation of bookings by the General & Equipment Section, as well as reports on building bookings/usage.

Plan by Feature Result

This stage focuses on a development scheme that refers to features. The researchers develop and determine the work priorities, starting from the core features (building borrowing), and then continuing with supporting features (usage/booking reports). This planning includes time allocation, task distribution, and the order of implementation so that the development process is more structured and the risk of delays can be minimized.

*name of corresponding author

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 10, Number 3, July 2026
DOI : https://doi.org/10.33395/sinkron.v10i3.16079
e-ISSN : 2541-2019
p-ISSN : 2541-044X

Design by Feature Result

At this stage, the focus is on designing the user interface (UI) and user experience (UX) for each feature. This design emphasizes aspects of accessibility and transparency, so that the public can know the loan status without having to come directly to the office. The result is a display design.

Login and Account Registration Screen

The initial system display presents a simple login and registration page, with input components such as email, password, and full name. New users who are not yet registered are directed to create an account, while registered users can directly log in to the system through the login page.

Dashboard Screen Display

The dashboard screen in the multipurpose building loan system is designed to present a summary of the main information needed by users. On this screen, users can directly see the list of registered loans, the availability status of the building, whether it is in use or still vacant, as well as the navigation menu to access other features such as the loan form, availability schedule, and building usage/booking reports.

Fig 6. Dashboard Screen Display

Multipurpose Building Loan Form Screen Display

The multifunction hall loan form screen serves as the main means for users to submit requests for using the hall. This screen contains an input form that includes the borrower's name, email, phone number, the purpose of the hall, and additional notes to specify dates or other needs.

Interactive Calendar Screen Display

The interactive calendar screen in the building rental system displays building availability digitally and in real-time.

Build by Feature Result

This stage is the transition from the design phase to the implementation phase. All previous design outputs, such as UML diagrams, feature lists, and UI/UX mock-ups, are realised as a web application that can be used directly by the general public and staff at the Parittiga Sub-District Office. Using a “Build by Feature” approach, development is carried out in a modular manner so that each feature can be built, tested, and integrated in stages.

Comparative Analysis Results

The results of the comparative experiment between Feature-Driven Development and Scrum are presented in Table 5 below:

Table 6. Comparison of Feature-Driven Development and Scrum Based on Three Metrics

Metric	Feature-Driven Development	Scrum	Difference
Average development time per feature (hours)	12	10	Feature-Driven Development is 2 hours slower
Number of functional errors (from 10 test cases)	2	4	Feature-Driven Development has 50% fewer errors
Developer understanding level (1–5 scale)	4.3	3.8	Feature-Driven Development is 13% higher

System Usability Scale (SUS) Calculation Results

*name of corresponding author

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 10, Number 3, July 2026
DOI : https://doi.org/10.33395/sinkron.v10i3.16079
e-ISSN : 2541-2019
p-ISSN : 2541-044X

Based on questionnaires completed by 5 community respondents who have used both systems (Feature-Driven Development and Scrum), the results are as follows:

Table 7. SUS Score Calculation for the Feature-Driven Development System

Respondent	Total Raw Score (0-40)	SUS Score (×2.5)
Respondent 1	34	85
Respondent 2	33	82
Respondent 3	35	88
Respondent 4	32	80
Respondent 5	34	86
Average	**33.68**	**84.2**

Table 8. SUS Score Calculation for the Scrum System

Respondent	Total Raw Score (0-40)	SUS Score (×2.5)
Respondent 1	29	72
Respondent 2	28	70
Respondent 3	30	75
Respondent 4	27	68
Respondent 5	29	73
Average	**33.68**	**71.6**

Based on Table 7 and Table 8, the Feature-Driven Development system obtained an average SUS score of 84.2, which falls into the "Excellent" category (Grade B) and is at the 90th percentile (better than 90% of the systems tested). Meanwhile, the Scrum system obtained an average SUS score of 71.6, which falls into the "Good" category (Grade C) and is at the 60th percentile. The score difference of 12.6 points indicates that the system developed using the Feature-Driven Development approach is significantly easier to use for the public compared to the Scrum approach. This indicates that the FDD approach provides a higher level of user satisfaction in the context of public services at the Parittiga Sub-district Office.

If calculated using the complete formula for 5 respondents:

$$\text{Average SUS}_{\text{FDD}} = \frac{85 + 82 + 88 + 80 + 86}{5} = \frac{421}{5} = 84.2$$

$$\text{Average SUS}_{\text{Scrum}} = \frac{72 + 70 + 75 + 68 + 73}{5} = \frac{358}{5} = 71.6$$

Results of Systematic Testing

Table 9. Black-box Testing Results (10 Scenarios)

Test Scenario	Feature-Driven Development Status	Scrum Status	Description
1. Successful login	✓ Pass	✓ Pass	-
2. Failed login	✓ Pass	✓ Pass	-
3. Complete borrowing form	✓ Pass	✓ Pass	-
4. Incomplete form	✓ Pass	✗ Fail	Scrum did not handle server-side validation
5. Booking on already booked date	✓ Pass	✓ Pass	-
6. Booking on available date	✓ Pass	✓ Pass	-
7. Booking cancellation	✓ Pass	✓ Pass	-
8. Approval by admin	✓ Pass	✓ Pass	-
9. Rejection by admin	✓ Pass	✓ Pass	
10. Report generation	✓ Pass	✗ Fail	Scrum did not provide a report feature in the initial sprint
Total Pass	**10/10**	**8/10**	-

Table 10. Usability Testing Results (SUS - System Usability Scale)

*name of corresponding author

e-ISSN : 2541-2019
p-ISSN : 2541-044X

Respondent	Feature-Driven Development SUS Score	Scrum SUS Score
Respondent 1	85	72
Respondent 2	82	70
Respondent 3	88	75
Respondent 4	80	68
Respondent 5	86	73
Average	**84.2**	**71.6**
Category	**Excellent**	**Good**

Table 11. Efficiency Testing Results

Process	Feature-Driven Development Time (minutes)	Scrum Time (minutes)	Efficiency
Open system page	0.5	0.5	Equal
Fill borrowing form	2.0	2.5	Feature-Driven Development 20% faster
Booking confirmation	1.0	3.0 (manual via Whatsapp)	Feature-Driven Development 66% faster
Generate report	0.5	5.0 (not integrated)	Feature-Driven Development 90% faster
Total end-to-end time	**4.0 minutes**	**11.0 minutes**	**Feature-Driven Development 63% more efficient**

Statistical Test Result

Table 12. Independent t-test Results for FDD vs Scrum Comparison

Metric	FDD Mean (SD)	Scrum Mean (SD)	t-value	p-value	Cohen's d	Effect Size	Significance
Development time (hours)	12 (1.2)	10 (0,8)	2.45	0.07	1.12	Large	Not significant ($p > 0.05$
Functional errors (n)	2 (0.8)	4 (1.0)	-3.16	0.03	1.58	Large	Significant ($p < 0.05$)
Developer understanding (1-5)	4.3 (0.3)	3.8 (0.4)	2.98	0.04	1.35	Large	Significant ($p < 0.05$)
SUS Score (0-100)	84.2 (5.2)	71.6 (4.8)	3.87	0.01	1.78	Large	Significant ($p < 0.05$)

The t-test results confirm that FDD significantly outperforms Scrum in terms of functional errors ($p = 0.03$), developer understanding ($p = 0.04$), and SUS usability score ($p = 0.01$). All three metrics showed large effect sizes (Cohen's $d > 0.8$), indicating that the observed differences are practically meaningful. The difference in development time was not statistically significant ($p = 0.07$), suggesting that the 2-hour difference per feature may be attributed to normal variation rather than a systematic difference between methods.

DISCUSSIONS

The comparison shows that FDD outperforms Scrum in feature accuracy (2 vs 4 errors) and developer understanding (4.3 vs 3.8) due to the Design by Feature phase, which produces detailed designs before coding. Conversely, Scrum excels in development speed (10 vs 12 hours/feature) by minimizing formal documentation. SUS testing confirmed FDD's superiority (84.2/Excellent vs 71.6/Good). These results reinforce that FDD is highly effective for systems with stable specifications, while Scrum's adaptability advantage is less optimal in this domain.

This study makes three theoretical contributions. First, it provides empirical evidence that FDD reduces functional errors by 50% compared to Scrum in stable public service environments. Second, it demonstrates that development methodology directly impacts usability outcomes, with FDD achieving a 12.6-point SUS advantage. Third, it introduces a replicable comparative framework using five evaluation metrics (development time, error rate, developer understanding, SUS, and efficiency).

*name of corresponding author

e-ISSN : 2541-2019
p-ISSN : 2541-044X

Practically, this study offers four actionable insights. First, for public service systems with stable processes, FDD is superior due to lower error rates and higher usability. Second, method selection directly impacts service efficiency, as FDD reduced borrowing time by 63% (11 to 4 minutes). Third, the evaluation framework provides practitioners with a validated toolkit for agile method selection. Fourth, government agencies should allocate extra design time (≈2 hours per feature) as an investment that reduces post-implementation errors.

This study has limitations: small SUS sample (n=5), a short development duration (two features), and single government context. Future work should: (1) involve larger samples (n ≥ 30), (2) extend observation periods to measure long-term maintenance, (3) replicate the framework across different public service domains, (4) explore hybrid FDD-Scrum methods, and (5) conduct longitudinal usability studies.

CONCLUSION

This study successfully developed a multipurpose building booking system for the Parittiga Sub-District Office using the Feature-Driven Development method. The FDD-based system improved booking efficiency by 63% (11 to 4 minutes), reduced functional errors by 50% (2 vs 4 errors), and achieved higher developer understanding (4.3 vs 3.8) and usability (SUS 84.2/Excellent vs 71.6/Good) compared to Scrum.

The primary contributions of this study are twofold. First, theoretically, it provides empirical evidence that FDD's design-centric approach yields superior accuracy and user acceptance for stable, documentation-oriented public service environments. Second, practically, it offers a replicable evaluation framework (including SUS, black-box testing, t-test, and efficiency metrics) for selecting agile methods in bureaucratic contexts**.** Future research should involve larger samples (n ≥ 30), extend observation periods to measure long-term maintenance, replicate the framework across different public service domains, and explore hybrid FDD-Scrum methods.

REFERENCES

Alsaadi, M., Lisitsa, A., Khalaf, M., & Qasaimeh, M. (2019). *Investigating the Capability of Agile Processes to Support Medical Devices Regulations: The Case of XP, Scrum, and FDD with EU MDR Regulations*. https://doi.org/https://doi.org/10.1007/978-3-030-26766-7_53

Cheikh, C., & Tebessi, L. (2024). *Enhancing Class Diagram Dynamics : A Natural Language Approach with ChatGPT*. 1–57.

Dan, I., Intech, T., Arafat, M., Trimarsiah, Y., & Susantho, H. (2022). Rancang Bangun Sistem Informasi Pemesanan Online Percetakan Sriwijaya Multi Grafika Berbasis Website. *INFORMATIKA DAN TEKNOLOGI (INTECH)*, *3*(2), 6–11.

Dema, G. J. C., & Simanjuntak, C. H. (2025). Pengembangan Sistem Tracking Produk Menggunakan Metode Feature-Driven Development Berbasis Flask. *Simtek : Jurnal Sistem Informasi Dan Teknik Komputer*, *10*(1), 179–183. https://doi.org/10.51876/simtek.v10i1.1544

Fatmawati, A., Solihati, T. I., & Winasis, P. H. (2025). RANCANG BANGUN SISTEM PEMINJAMAN BARANG DAN RUANGAN DI UNIVERSITAS BANTEN JAYA. *Jurnal Innovation and Future Technology (IFTECH) P-ISSN:*, *7*(2), 347–358.

Firdonsyah, A., & Putri, N. A. (2025). Meeting Room Booking System with WhatsApp Notification Feature Using Extreme Programming Methods in RS Muhammadiyah Lamongan. *Sinkron*, *9*(3), 1036–1049. https://doi.org/10.33395/sinkron.v9i3.14927

Herwindiati, D. E., Setyawan, I. R., & Maupa, H. (2023). Desain Situs Web Yang Responsif Berdasarkan Strategi Agile Sebagai Pendukung Pemasaran Destinasi Wisata. *Jurnal Teknik Informatika Dan Sistem Informasi*, *10*(1), 526–540.

Iscte-iul, I. U. D. L. (2023). Connection between UML use case diagrams and UML class diagrams : a matrix proposal Bráulio Alturas. *Int. J. Computer Applications in Technology*, *72*(3).

Jacob Cohen. (1988). *Statistical Power Analysis for the Behavioral Sciences*. Routledge; 2nd edition.

Kecamatan Parittiga. (2026). *Website Kecamatan Parittiga*. https://www.kecamatanparittiga.bangkabaratkab.go.id/

Kusnadi, K., Dwi Kartika, V., & Gema Ramadhan, F. (2024). Penerapan Metode Feature Driven Development Pada Sistem Pencari Kerja Berbasis Website. *JATI (Jurnal Mahasiswa Teknik Informatika)*, *8*(5), 8779–8784. https://doi.org/10.36040/jati.v8i5.10810

Mardi Suryanto, T. L., Faroqi, A., & Simarmata, W. N. (2022). System Usability Scale (Sus) Sebagai Metode Pengujian Kegunaan Pada Situs Program Studi. *Prosiding Seminar Nasional Teknologi Dan Sistem Informasi*, *2*(1), 285–294. https://doi.org/10.33005/sitasi.v2i1.314

Narulita, S., Nugroho, A., & Abdillah, M. Z. (2024). Diagram Unified Modelling Language (UML) untuk Perancangan Sistem Informasi Manajemen Penelitian dan Pengabdian Masyarakat (SIMLITABMAS) Universitas Nasional Karangturi Semarang , Indonesia (deskripsi) dan perancangan sistem , khususnya

*name of corresponding author

pada pemrogr. *BRIDGE : Jurnal Publikasi Sistem Informasi Dan Telekomunikasi*, *3*, 244–256.

Pambudi, R., & Dwi, O. (2020). Systematic Literature Review : Analisis Model Agile dalam Pengembangan Sistem Informasi. *Indexia: Informatics and …*, *01*(2018), 1–8. https://journal.umg.ac.id/index.php/indexia/article/view/9594%0Ahttps://journal.umg.ac.id/index.php/indexia/article/download/9594/5112

Putra Rakhmadani, D., Fadila Fitriana, G., Putu Restu Indrawan, I., & Iffah, T. (2022). Rancang Bangun Aplikasi Pengendalian Kualitas Beras Terpadu Di Jawa Tengah Menggunakan Metode Feature-Driven Development (FDD). *Jurnal Teknik Informatika Dan Sistem Informasi*, *9*(3), 2676–2686. http://jurnal.mdp.ac.id

Putri, A. M., Sokibi, P., Informatika, T., Catur, U., Cendekia, I., Cirebon, K., Development, F. D., Testing, U. A., Sembiring, T., Barat, J., & Babakan, K. (2025). PENERAPAN FEATURE DRIVEN DEVELOPMENT DALAM PERANCANGAN. *JATI (Jurnal Mahasiswa Teknik Informatika)*, *9*(6), 9558–9564.

Ramadani, N., Afrian, R., Haritz, A., Yul, F. A., & Sah, S. M. (2025). Sistem Reservasi Gedung dan Ruangan PKM Universitas Bengkulu Berbasis Teknologi Informasi. *Indonesian Journal of Computer Science and Engineering (IJCSE).*, *02*, 9–15.

Rasyad, I. H., Irawan, M. D., & Triase, T. (2025). Implementasi Metode Feature Driven Development Pada Sistem Informasi Pelayanan Masyarakat. *Simtek : Jurnal Sistem Informasi Dan Teknik Komputer*, *10*(2), 320–328. https://doi.org/10.51876/simtek.v10i2.1609

Rekayasa, T., Lunak, P., Manufaktur, P., Bangka, N., Absensi, S., Development, F. D., & Guru, M. (2025). EFEKTIVITAS PENERAPAN SISTEM ABSENSI GURU BERBASIS GEOLOKASI MENGGUNAKAN METODE FEATURE DRIVEN DEVELOPMENT (FDD) DI SMP NEGERI 2 RIAU SILIP. *Jurnal KHARISMA Tech*, *02*, 95–105.

Rizky, M., & Sugiarti, Y. (2022). Pengunaan Metode Scrum Dalam Pengembangan Perangkat Lunak: Literature Review. *Journal of Computer Science and Engineering (JCSE)*, *3*(1), 41–48. https://doi.org/10.36596/jcse.v3i1.353

Romero, A. V., Fahrudin, R., Informatika, T., Catur, U., Cendekia, I., Cirebon, K., & Barat, J. (2023). Membangun Marketplace Untuk Penjualan Produk Kreatif. *JATI (Jurnal Mahasiswa Teknik Informatika)*, *7*(6), 3400–3405. http://ejournal.itn.ac.id/index.php/jati/article/view/7278

Samuel Gbli Tetteh. (2024). *Empirical Study of Agile Software Development Methodologies : A Comparative Analysis*. *17*(5), 30–42. https://doi.org/10.9734/AJRCOS/2024/v17i5436

WP, D. A. (2025). Implementasi Feature-Driven Development dalam Pengembangan Sistem Informasi Manajemen Aset Desa. *Jurnal Ilmiah FIFO*, *17*(1), 63. https://doi.org/10.22441/fifo.2025.v17i1.008

*name of corresponding author