Jurnal Ilmiah Informatika dan Komputer (Informatech)
Volume 3, Number 2, December 2026, page 206 - 214
DOI : 10.69533
Website : *https://ejournal.rizaniamedia.com/index.php/informatech*

E-ISSN: 3047-4752

Quality Evaluation of an RFID Parking Desktop Application Using ISO 9126

Muhammad Fitra Abdi Tama[1*], Nadila Dewi Mustika[2], Bayu Adi Saputro[3], Dhony Manggala Putra[4]

Jember State Polytechnic PSDKU Sidoarjo, Indonesia
[*1]*email: abditamafit@gmail.com*
[2]*email : nadiladewimustika1205@gmail.com*
[3]*email : bayuadi00saputro@gmail.com*
[4]*email : dhony_manggala@polije.ac.id*

(Article Received: 19 October 2026; Article Revised: 20 November 2026; Article Published: 1 December 2026;)

ABSTRACT – *Software quality assurance is essential before an RFID-based parking application is released for daily operation. This study evaluates a Java desktop parking application that uses RFID identification and a MySQL database hosted on XAMPP, guided by the ISO 9126 quality model. The database access path was tested with Apache JMeter through a JDBC request that queries the parking history table under ten concurrent virtual users (40 samples); this test gives partial evidence on functionality, reliability, and efficiency but does not exercise the graphical interface or the RFID reader. Usability was measured with the System Usability Scale (SUS) completed by 16 respondents who all held the operator role. All 40 JMeter requests succeeded (0% error rate), with an average response time of 15 ms, a median of 5 ms, a 95th percentile of 36 ms, a maximum of 75 ms, and a throughput of 1.2 requests per second. In contrast, the mean SUS score was 49.69 (SD = 4.17), which is below the widely used benchmark of 68 and below the "OK" adjective anchor. Because this result comes from 16 operators who use the application daily, it indicates that the operator interface needs improvement before wider deployment, whereas the technical results are limited to data retrieval under light load and do not validate the application as a whole.*

Keywords – *Apache JMeter; ISO 9126; RFID parking system; software quality testing; System Usability Scale*

Evaluasi Kualitas Aplikasi Desktop Parkir Berbasis RFID Menggunakan ISO 9126

ABSTRAK – *Penjaminan kualitas perangkat lunak sangat penting sebelum aplikasi parkir berbasis RFID dirilis untuk operasional sehari-hari. Studi ini mengevaluasi aplikasi parkir berbasis desktop Java yang menggunakan identifikasi RFID dan basis data MySQL yang di-hosting di XAMPP, dengan mengacu pada model kualitas ISO 9126. Jalur akses basis data diuji menggunakan Apache JMeter melalui permintaan JDBC yang melakukan kueri pada tabel riwayat parkir dengan sepuluh pengguna virtual yang mengakses secara bersamaan (40 sampel). Pengujian ini memberikan bukti parsial mengenai fungsionalitas, keandalan, dan efisiensi, namun tidak mencakup antarmuka grafis maupun perangkat pembaca RFID. Aspek kegunaan (usability) diukur menggunakan *System Usability Scale* (SUS) yang diisi oleh 16 responden yang semuanya berperan sebagai operator. Seluruh 40 permintaan melalui JMeter berhasil (tingkat kesalahan 0%), dengan waktu respons rata-rata 15 ms, median 5 ms, persentil ke-95 sebesar 36 ms, nilai maksimum 75 ms, serta *throughput* sebesar 1,2 permintaan per detik. Sebaliknya, skor rata-rata SUS adalah 49,69 (SD = 4,17); angka ini berada di bawah tolok ukur umum sebesar 68 dan di bawah ambang batas kategori "OK". Mengingat hasil ini diperoleh dari 16 operator yang menggunakan aplikasi tersebut setiap hari, dapat disimpulkan bahwa antarmuka operator memerlukan perbaikan sebelum aplikasi diterapkan secara lebih luas. Sementara itu, hasil teknis yang diperoleh terbatas pada pengambilan data dalam kondisi beban ringan dan belum memvalidasi aplikasi secara keseluruhan.*

Keywords – *Apache JMeter; ISO 9126; sistem parkir RFID; pengujian kualitas perangkat lunak; System Usability Scale*

1. INTRODUCTION

Parking services are increasingly digitalized because manual vehicle recording is slow, prone to input errors, and produces unreliable operational data. Radio Frequency Identification

(RFID) is widely adopted to address these problems because it identifies vehicles quickly and without physical contact. Recent studies describe RFID parking prototypes built on microcontrollers and MySQL databases with web-based reservation and monitoring [1], low-cost RFID gate controllers that admit only registered cards with an average delay of about 0.3 s [2], and touchless parking payment that is integrated with mobile payment services [3].

Most of this work, however, concentrates on hardware prototypes and system design, and reports little about the software quality of the application that operators use every day. A desktop parking application built on a relational database has likewise been described mainly through its design and implementation [16]. An application that is unstable, slow, or difficult to operate can undermine the benefits of automation even when the RFID hardware works correctly, so software quality should be verified systematically before deployment. ISO/IEC 9126 provides a widely used quality model that defines functionality, reliability, usability, efficiency, maintainability, and portability [4], and it has been applied to web portals, web applications, and mobile applications [5], [6], [24].

Performance-related characteristics are commonly measured with Apache JMeter, which simulates concurrent users and records response time, throughput, and error rate. Recent studies applied JMeter to web sites and application programming interfaces (APIs) with loads from 10 to 5,000 virtual users [7]–[10]. Usability, in contrast, is often measured with the System Usability Scale (SUS), a ten-item questionnaire [11] whose scores can be interpreted with adjective ratings [12] and compared with a benchmark mean of 68 [13]. A structured comparison of related studies (Table 1 in Section 2) shows that the RFID parking studies report no quality evaluation under a recognized model, that the ISO 9126 studies concern web or mobile objects, and that the JMeter studies include no usability measure. We found no study that evaluates an RFID parking *desktop* application by combining a JMeter test of its database layer with a SUS-based usability assessment under the ISO 9126 model. This is the gap addressed here; the claim is limited to the studies compared in Table 1.

The object of this study is a Java desktop parking application that uses RFID identification and a MySQL database served through XAMPP. The purpose of the article is to (i) assess the functionality, reliability, and efficiency of the application with Apache JMeter, (ii) measure its usability with SUS, and (iii) decide whether the application meets predefined acceptance criteria and which aspects need improvement.

2. LITERATURE REVIEW

ISO 9126 Quality Model

ISO/IEC 9126-1 defines a quality model for software products with six characteristics: functionality, reliability, usability, efficiency, maintainability, and portability [4]. Functionality describes whether the software provides functions that satisfy stated needs; reliability describes its ability to maintain a level of performance under stated conditions; efficiency describes its performance relative to the resources it uses; and usability describes how easily users can understand and operate it. Studies that adopt the model usually assign a different technique to each characteristic. For example, a news portal was assessed with black-box testing for functionality, stress testing for reliability, a Likert questionnaire for usability, and an online tool for efficiency [5], while another study evaluated a mobile application with ISO/IEC 25010 and found 11 failed cases among 97 functional tests [6]. ISO/IEC 9126 has also served as a classification scheme for software defects [25], and a web monitoring application was assessed with four of its characteristics, using SUS for usability, a stress test for reliability, and a profiling tool for efficiency [24]. The same logic is followed in this study.

Performance Testing and Apache JMeter

Performance testing measures how quickly and how consistently a system responds under a defined workload. Typical indicators are response time (average, median, and percentiles), throughput, and error rate. Apache JMeter is an open-source tool that generates such workloads through thread groups and samplers, and records results with listeners such as the Summary Report and the Aggregate Report. Published studies show that it can be used for web applications with up to 5,000 virtual users, where a site remained reliable up to 200 concurrent users but exceeded 60 s response time and produced errors beyond 300 users [7]; for an API with ten endpoints under 10 to 500 users, where one endpoint failed completely because of validation conflicts [8]; and for the login, profile, upload, and question features of an information system with 50 and 100 users [9]. Further work

reports that response time and stability degrade as test duration and load increase [10]. JMeter can also test databases through its JDBC Request sampler, which executes SQL statements through a configured connection pool and therefore measures the data layer directly. Query-level timing has also long been used to compare database engines such as MySQL and PostgreSQL [21].

System Usability Scale

The SUS consists of ten statements rated on a five-point agreement scale, alternating between positive (odd) and negative (even) wording [11]. Each response is converted to a contribution of 0–4 and the sum is multiplied by 2.5 to yield a score between 0 and 100. Because a raw score is hard to interpret in isolation, adjective anchors were derived for the scale (for example, Poor = 35.7, OK = 50.9, Good = 71.4, and Excellent = 85.5) [12]. A meta-analysis found that the conventional benchmark mean of 68 (SD = 12.5) remains a reasonable reference [13]. A recent study of a Moodle learning platform with 120 students reported a mean SUS of 71.51 [14], and an enhanced SUS aligned with ISO 9126 and ISO 9241-11 has also been proposed [15]. Since its publication the scale has been examined in large data sets: one analysis covered 2,324 surveys from 206 studies [18], a later review describes it as the most widely used standardized questionnaire of perceived usability [17], and its originator has reflected on its use in a retrospective [20]. Item-level targets derived from 166 studies and 11,855 questionnaires allow single items to be compared with benchmarks, for example a mean of at least 3.39 for item 1 when the overall SUS is 68 [19]. Means below 50 are uncommon in published data (fewer than 5% of 273 studies [12]) but they do occur and are not an artifact of the scale [22]. An Indonesian adaptation of the SUS has also been published [23].

Related Work

Table 1 compares the studies that are most relevant to this article by object and method. Three patterns follow from the comparison. First, the RFID parking studies [1], [2], [16] describe prototypes or system design and report no quality evaluation under a recognized quality model. Second, the studies that apply ISO 9126 or ISO/IEC 25010 [5], [6], [24] concern web portals, a web application, and a mobile application, not desktop parking software. Third, the load-testing and database studies [7], [8], [21] address web sites, APIs, and database engines, and none of them is combined with a usability measure. Only [5], [14], and [24] collect perceived usability, and none of these concerns an RFID parking application. The comparison is selective rather than exhaustive, but it supports the gap described in the Introduction.

Table 1. Comparison of related studies by object, method, and finding

Ref.	Object and method	Key finding
[1]	RFID parking prototype; microcontroller, MySQL, web reservation	Entry, exit, fees, and slot status monitored through a website
[2]	Low-cost RFID parking gate prototype	Only registered cards opened the gate; mean delay about 0.3 s
[16]	RFID parking application; VB.NET, MySQL, desktop	Design and implementation; no quality evaluation reported
[5]	News portal; ISO 9126 (black-box, stress, Likert, online tool)	Full functionality and reliability scores; usability about 79%
[24]	Web monitoring application; ISO/IEC 9126, SUS, stress test	Usability (SUS 61.81) lowest of four characteristics
[6]	Mobile application; ISO/IEC 25010	11 of 97 functional tests failed
[7]	Journal website; JMeter, 50–5,000 users	Reliable up to 200 users; errors beyond 300 users
[8]	API; JMeter, 10–500 users	Stable at moderate load; one endpoint failed
[21]	MySQL vs PostgreSQL; query execution times	Select, insert, delete, and order-by timings compared
[14]	Moodle platform; SUS, 120 students	Mean SUS 71.51
This study	RFID parking desktop application; JMeter JDBC test and SUS under ISO 9126	Data path fast and error-free; SUS 49.69

3. RESEARCH METHODS

Research Design and Object

This study uses a descriptive quantitative design. The object is a Java desktop parking application that identifies vehicles through RFID

tags, records entry and exit times, calculates parking fees, and stores transactions in a MySQL database managed through XAMPP. Because the test tooling reaches the application only through its database, the JMeter results are treated as evidence about the data-access path and not about the complete desktop application (see the JMeter Test Configuration subsection). The research stages are shown in Fig. 1: problem identification and literature study, definition of the object, test design based on ISO 9126, a JMeter load test and a SUS questionnaire as parallel data collection activities, data analysis, and conclusion.

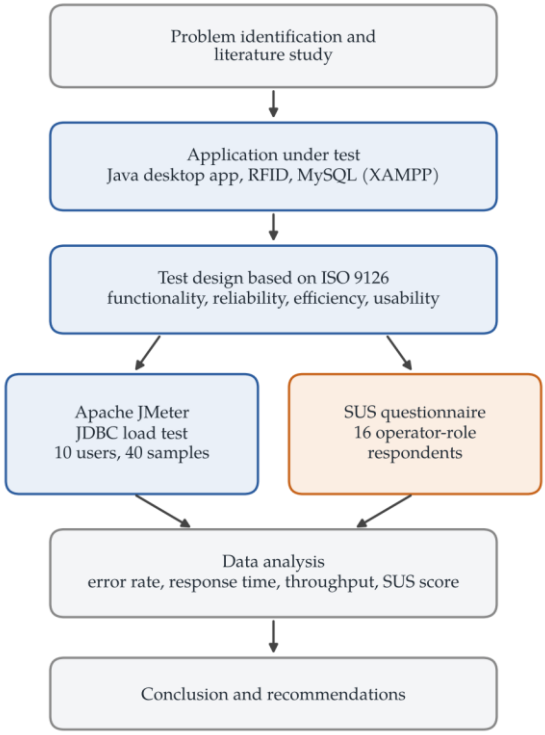


Figure 1. Research stages

Test Design Based on ISO 9126

Four ISO 9126 characteristics are examined; maintainability and portability are outside the scope of this article. Table 2 gives an operational definition of each characteristic based on the sub-characteristics in ISO/IEC 9126-1 [4], the evidence collected, and how far that evidence reaches. The JMeter test covers only the database access path (JDBC), not the graphical interface, the RFID reader, or the fee calculation logic, so the first three characteristics are reported as partial, data-layer evidence. Functionality and reliability are derived from the same 40 samples and are not independent measurements.

Table 2. Operational definitions of ISO 9126 characteristics, evidence, and coverage

Aspect	Operational definition	Evidence and metric	Scope
Functionality	Suitability: the parking history can be read through the database connection	Success of 40 JDBC samples; success rate (%)	Partial (data retrieval only)
Reliability	Maturity: no failures during a run under light load	Error rate (%) and SD of response time; same 40 samples	Partial (no recovery test)
Efficiency	Time behavior of the database access path	Average, median, percentiles (ms); throughput (requests/s)	Partial (time behavior only)
Usability	Understandability and operability as perceived by operators	SUS score (0–100), 16 operators	Perceived usability only

JMeter Test Configuration

The test was executed with Apache JMeter 5.6.3. A JDBC Connection Configuration element (pool variable *jdbcConfig*) connects JMeter to the MySQL server in XAMPP. A Thread Group runs a JDBC Request sampler of type *Select Statement* that executes the following query on the parking history table:

```
SELECT id_riwayat, id_pengunjung,
waktu_masuk, waktu_keluar, total_bayar
FROM riwayat_parkir;
```

The results are collected by a View Results Tree, a Summary Report, and an Aggregate Report listener, as shown in Fig. 2. The test used ten concurrent virtual users, and the run produced 40 samples in total, which is four samples per virtual user on average (Table 3); at the reported throughput of 1.2 requests/s this corresponds to a run of roughly 33 s. The query reads transaction records that include the entry time, exit time, and total fee, so it exercises the data layer on which the RFID entry, exit, and fee functions depend. The JDBC sampler was used because the application keeps its transactions in MySQL, which makes the database the one component that JMeter can drive without modifying the application; the same sampler cannot operate the graphical interface or the RFID reader.

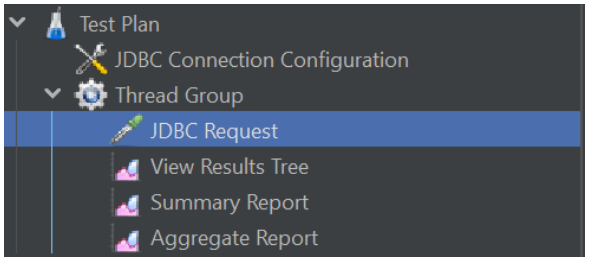


Figure 2. JMeter test plan structure

The configuration has three further limits: only one read-only query was executed, so write operations such as recording an entry or exit were not loaded; the load was not derived from measured operational traffic; and the database ran on a local XAMPP server, so network latency is not represented. The test is therefore a light check of the data-access path, not a load test of the whole application.

Table 3. JMeter test configuration

Parameter	Value
Tool	Apache JMeter 5.6.3
Sampler	JDBC Request (Select Statement)
Connection pool	jdbcConfig (MySQL, XAMPP)
Target table	riwayat_parkir
Virtual users	10
Total samples	40
Listeners	View Results Tree, Summary Report, Aggregate Report

Usability Questionnaire and Data Analysis

Usability was measured with the ten-item SUS questionnaire, completed through an online form on 11 June 2025 by 16 respondents who all had the *Operator* role and had used the application. Responses use a 1–5 scale. For odd-numbered items the contribution is

$$s_i = x_i - 1 \quad (1)$$

and for even-numbered items

$$s_i = 5 - x_i \quad (2)$$

where x is the response to item i. The SUS score of respondent j and the mean score of n respondents are

$$SUS_j = 2.5 \times \Sigma\, s_i \quad (i = 1,\ldots,10) \quad (3)$$

$$\text{Mean SUS} = (1/n) \times \Sigma\, SUS_j \quad (j = 1,\ldots,n) \quad (4)$$

For the JMeter data, the error rate is the percentage of failed samples among all samples, and throughput is the number of samples N divided by the test duration T:

$$\text{Error rate} = \text{failed} / \text{total} \times 100\% \quad (5)$$

$$\text{Throughput} = N / T \quad (6)$$

The application is considered acceptable when (i) all executed functional requests succeed, (ii) the error rate does not exceed 1%, (iii) the average response time does not exceed 1,000 ms, and (iv) the mean SUS score reaches at least the benchmark of 68 [13]. ISO 9126 does not prescribe numeric thresholds, so these values are pragmatic acceptance criteria for an interactively used desktop application; they were defined by the authors before the analysis. All calculations were repeated from the raw questionnaire sheet and the JMeter reports.

4. Results

Functionality

Every JDBC Request sample was marked successful in the JMeter View Results Tree. The query returned a result set with an average response size of 2,234 bytes, which indicates that the parking history table, including entry time, exit time, and total fee columns, could be read from the database through the application's connection settings. The functional success rate is therefore 40 of 40 samples (100%). This result applies to the transaction data retrieval function only: the RFID reader hardware and the fee calculation logic were not instrumented by this test and need separate black-box test cases.

Reliability

As Table 4 shows, the 40 samples produced a 0.00% error rate. The standard deviation of the response time is 15.97 ms and the maximum is 75 ms, so no time-outs or extreme outliers occurred, and the server processed ten concurrent users without failures. This indicator is computed from the same 40 samples as the functionality indicator and is therefore not independent evidence.

Table 4. JMeter aggregate results (40 samples)

Metric	Value
Average / Median	15 ms / 5 ms
90% / 95% / 99% line	30 ms / 36 ms / 75 ms
Minimum / Maximum	1 ms / 75 ms
Standard deviation	15.97 ms
Error rate	0.00%
Throughput	1.2 requests/s
Received / Sent data rate	2.53 / 0.00 KB/s
Average response size	2,234.0 bytes

Efficiency

The average response time is 15 ms and the median is 5 ms (Fig. 3). Because the average is three times the median, the distribution is right-skewed: most requests complete within a few milliseconds, and 95% complete within 36 ms.

The slowest request (75 ms) may reflect connection initialization, although this cannot be confirmed from aggregate data. Even this worst case is more than 13 times faster than the 1,000 ms criterion. The received data rate of 2.53 KB/s and the throughput of 1.2 requests/s describe a light workload and should not be read as the maximum capability of the system.

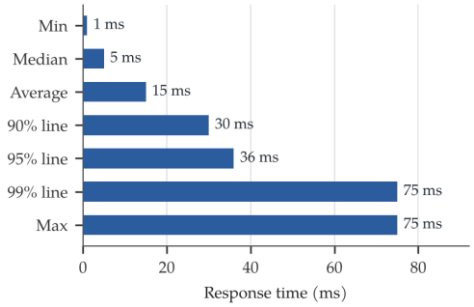


Figure 3. Response time statistics of the 40 JDBC samples (ms)

Usability

Table 5 lists the SUS scores computed from the questionnaire sheet, and Fig. 4 compares them with the benchmark. The mean score is 49.69 (SD = 4.17, median 50.0, range 42.5–57.5); the 95% confidence interval based on the t-distribution (n = 16) is 47.47–51.91. The sum of all individual scores (795) divided by 16 reproduces the same value. No respondent reached the benchmark of 68, and the highest individual score was 57.5.

Table 5. SUS score of each respondent (mean = 49.69, SD = 4.17)

Resp.	Σ s	SUS	Resp.	Σ s	SUS
R1	20	50.0	R9	17	42.5
R2	21	52.5	R10	18	45.0
R3	19	47.5	R11	18	45.0
R4	22	55.0	R12	23	57.5
R5	20	50.0	R13	20	50.0
R6	20	50.0	R14	21	52.5
R7	19	47.5	R15	20	50.0
R8	18	45.0	R16	22	55.0

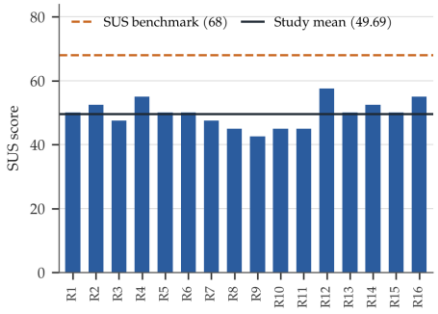


Figure 4. SUS score per respondent compared with the benchmark

The score is 18.3 points below the benchmark [13] and slightly below the "OK" anchor of 50.9, which lies between the "Poor" and "OK" descriptors [12]; because the upper confidence limit (51.91) marginally exceeds that anchor, usability is classified as *marginal to poor* (Fig. 5). The usability indicator is therefore not met.

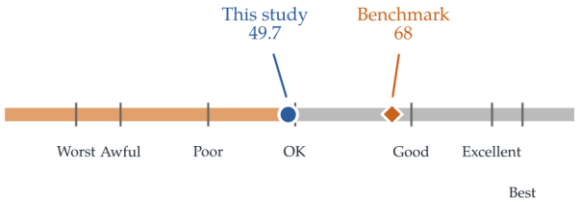


Figure 5. Position of the mean SUS score on the adjective anchors [12]

The item-level distribution (Fig. 6) shows where the score is lost. Of all 160 answers, 59.4% were neutral, 20.6% unfavorable, and only 20.0% favorable. The weakest item is Q1 (willingness to use the system frequently; mean contribution 1.63, 37.5% unfavorable), followed by Q8 (cumbersomeness) and Q6 (inconsistency), each with 31.3% unfavorable answers. The strongest items are Q5 (integration of functions; 2.19) and Q2, Q3, and Q9 (2.13 each). The concentration of answers at the midpoint limits score variance, which is why the standard deviation is small, so the item-level reading is indicative only.

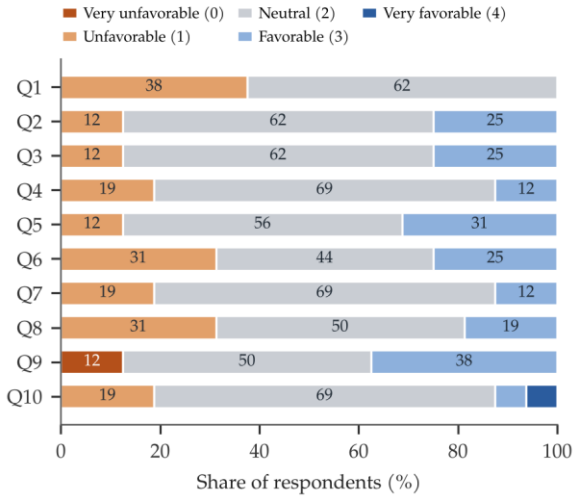


Figure 6. Distribution of SUS item contributions (0–4); Q1–Q10 follow the standard SUS order

Summary Against Acceptance Criteria

Table 6 compares the results with the acceptance criteria. The three technical indicators are met for the data-access path, but the usability indicator is not met. Functionality and reliability are computed from the same 40 JDBC samples and efficiency from the same run, so the technical

results form one body of evidence about data retrieval under light load rather than three independent confirmations.

Table 6. Acceptance criteria and results

Aspect	Criterion	Result	Met?
Functionality	All requests succeed	40/40 (100%)	Met*
Reliability	Error rate ≤ 1%	0.00%	Met*
Efficiency	Average ≤ 1,000 ms	15 ms	Met*
Usability	SUS ≥ 68	49.69	Not met

Note: *Met for the database access path only (partial evidence); the graphical interface, RFID reading, and fee calculation were not tested.*

5. DISCUSSION

Technical Performance of the Data-Access Path

The JDBC test shows that reading the parking history is fast and failure-free for ten concurrent users on a local XAMPP server. Studies of web applications report errors only when the load reaches hundreds of concurrent users [7], [8], and the single failing API endpoint in [8] was caused by validation conflicts rather than by load. The present load is far below those levels, so the result confirms stable behavior under *light* load but does not establish the capacity limit of the system. With an average of 15 ms against a 1,000 ms criterion, this single read query leaves wide headroom, and such timings mainly describe the database engine and its configuration [21]. Because functionality, reliability, and efficiency rest on the same 40 samples, they should not be presented as comprehensive validation: they say nothing about RFID read accuracy, fee calculation, or the behavior of the graphical interface.

Interpretation of the Usability Result

A mean SUS of 49.69 from 16 operators, none above 57.5, falls in the lowest grade band (below 51.7) of the curved grading scale reported in [19], although the upper confidence limit of 51.91 touches the next band. Means below 50 are uncommon in published data [12] but are not an artifact of the scale [22], so the result is better read as a real signal about the interface. At the item level, even the best-rated items fall short of the targets that correspond to an overall SUS of 68 [19]. Converted back to the 1–5 response scale, item 3 (ease of use) averages 3.13 against a target of at least 3.67, item 5 (integration) 3.19 against 3.55, and item 1 (frequent use) 2.63 against 3.39. The pattern suggests that operators find the functions integrated but the workflow cumbersome and inconsistent. Item 1 should be read with care, because operators use the application as part of their work and their willingness to use it frequently may reflect obligation rather than preference. The large share of neutral answers (59.4%) indicates ambivalence rather than strong rejection, which is why confirmation with a larger sample is needed.

Comparison with Previous Research

The combination of acceptable technical indicators and weak usability resembles the web monitoring application evaluated with ISO/IEC 9126 in [24], where usability, measured with SUS (mean 61.81, 29 respondents), was the lowest-scoring of four characteristics. The present mean is about 12 points lower than that value and about 22 points lower than the 71.51 reported for a Moodle platform with 120 students [14]. The news portal in [5] reached a usability level of about 79% with a Likert questionnaire, but a different instrument was used, so the two figures are not directly comparable. On the technical side, the zero error rate agrees with the stable behavior reported at low and moderate loads in [7] and [8]. The RFID parking studies [1], [2], [16] show that parking functions can be realized, yet they do not report operator usability; the present result indicates that a working design and a fast data layer do not by themselves ensure an interface that operators rate favorably.

Implications

For the development team, the results point to the operator interface rather than the database as the priority for improvement: the workflow should be simplified and made consistent, using the weakest items (Q1, Q6, and Q8) as starting points, and the SUS should then be repeated. For the organization that operates the application, the results argue against basing a deployment decision on technical tests alone, because the application would appear ready if only the JMeter indicators were considered. For evaluation practice, the study shows that pairing a back-end load test with a perceived-usability measure can reveal a quality risk in a different layer from the one tested, and the JMeter plan can be reused as a regression check after the interface is redesigned.

Limitations

First, JMeter exercised the database layer through JDBC, not the graphical interface or the RFID reader, so RFID read accuracy, fee calculation correctness, and interface behavior

remain untested and the technical results are partial. Second, functionality and reliability were derived from the same 40 samples and are therefore not independent. Third, the load of ten users and 40 samples of one read-only query against a local XAMPP server is light, and no write, stress, spike, or endurance scenario was run. Fourth, the SUS sample is small (n = 16), homogeneous (all operators), concentrated at the scale midpoint, collected on a single date, and not accompanied by task-based observation. Fifth, the acceptance thresholds for the technical metrics were defined by the authors rather than taken from a standard.

Future Research

Future research should (a) add black-box and hardware-in-the-loop test cases for RFID reading, fee calculation, and the graphical interface; (b) extend the JMeter plan with write operations, ramp-up, and increasing numbers of virtual users, together with endurance runs, to locate the capacity limit; (c) redesign the operator workflow and repeat the SUS with a larger and more varied sample, reporting the language version used, such as the validated Indonesian adaptation [23], and adding task-based measures such as completion time and error counts; and (d) extend the evaluation to maintainability and portability or to ISO/IEC 25010.

6. CONCLUSION

This study evaluated an RFID-based parking desktop application with Apache JMeter and SUS under the ISO 9126 model. The results are as follows. (1) Functionality: all 40 JDBC requests that read the parking history succeeded (100%), which is partial evidence limited to data retrieval. (2) Reliability: the error rate was 0.00% with a standard deviation of 15.97 ms, based on the same 40 samples. (3) Efficiency: the average response time was 15 ms and the 95th percentile 36 ms, far below the 1,000 ms criterion for this read query. (4) Usability: the mean SUS score from 16 operators was 49.69, which is below the benchmark of 68 and the "OK" anchor, so usability is not yet acceptable. The database access path therefore performed well under light load, but this evidence is limited to data retrieval and is not a comprehensive validation of the software: the interface, the RFID reader, and the fee calculation were not tested. Given the low usability score reported by the operators, the application should not be considered ready for wider deployment until its interface is improved and re-evaluated. Future work should (a) redesign the operator workflow and interface for consistency and fewer steps, then repeat the SUS with a larger sample, (b) add black-box test cases for RFID reading and fee calculation, and (c) run JMeter scenarios with write operations and increasing numbers of users to find the capacity limit of the system.

BIBLIOGRAPHY

[1] F. Tannujaya, F. Hasanah, and A. T. Nugroho, "Design and realization of parking system prototype using RFID based microcontroller with reservation service via website for four-wheel vehicle," in *Proc. 2nd Int. Seminar Sci. Appl. Technol. (ISSAT 2021)*, Adv. Eng. Res., vol. 207, Atlantis Press, 2021.

[2] M. Hanief, L. H. Nugroho, and M. Galina, "Low cost RFID-based security system for parking lots," *Jurnal Media Elektro*, vol. 11, no. 2, 2022, doi: 10.35508/jme.v0i0.8126.

[3] T. J. Eing and I. F. Kamsin, "Touchless parking payment system using RFID technology integrated FinTech payment solutions," *J. Appl. Technol. Innov.*, vol. 7, no. 3, 2023.

[4] *Software Engineering – Product Quality – Part 1: Quality Model*, ISO/IEC 9126-1:2001, International Organization for Standardization, Geneva, Switzerland, 2001.

[5] M. R. Firdaus, Y. Alkhalifi, D. Ismunandar, and O. Kurniawan, "Analisis kualitas website portal media online Milenianews.com menggunakan standar ISO 9126," *INTI Nusa Mandiri*, vol. 19, no. 2, 2025, doi: 10.33480/inti.v19i2.6218.

[6] D. A. Suryadi and E. Sulistiyani, "Evaluation of information quality using ISO/IEC 25010:2011 (case research: Menu Harianku application)," *Int. J. Innov. Enterp. Syst.*, vol. 6, no. 2, pp. 143–156, Jul. 2022, doi: 10.25124/ijies.v6i02.167.

[7] A. Annisa, M. Ardiana, P. N. Rahayu, T. Brian, and M. A. Jami'in, "Performance evaluation of web applications using JMeter load testing for server capacity and response efficiency," *J. Appl. Informatics Comput.*, vol. 10, no. 1, 2026, doi: 10.30871/jaic.v10i1.11840.

[8] A. N. Rachman *et al.*, "Load testing-based performance evaluation of the SiUKT API system," *J. Intell. Syst. Technol. Informatics*, vol. 2, no. 1, pp. 22–29, Mar. 2026, doi: 10.64878/jistics.v2i1.89.

[9] I. Indrianto, "Performance testing on web information system using Apache JMeter

and BlazeMeter," *Jurnal Ilmiah Ilmu Terapan Universitas Jambi*, vol. 7, no. 2, pp. 138–149, Dec. 2023, doi: 10.22437/jiituj.v7i2.28440.

[10] H. Alnuhait *et al.*, "Web application performance assessment: A study of responsiveness, throughput, and scalability," *Int. J. Adv. Appl. Sci.*, vol. 11, no. 9, pp. 214–226, 2024, doi: 10.21833/ijaas.2024.09.023.

[11] J. Brooke, "SUS: A 'quick and dirty' usability scale," in *Usability Evaluation in Industry*, P. W. Jordan, B. Thomas, B. A. Weerdmeester, and I. L. McClelland, Eds. London, U.K.: Taylor & Francis, 1996, pp. 189–194.

[12] A. Bangor, P. Kortum, and J. Miller, "Determining what individual SUS scores mean: Adding an adjective rating scale," *J. Usability Stud.*, vol. 4, no. 3, pp. 114–123, May 2009.

[13] M. Hyzy *et al.*, "System Usability Scale benchmarking for digital health apps: Meta-analysis," *JMIR mHealth uHealth*, vol. 10, no. 8, p. e37290, Aug. 2022, doi: 10.2196/37290.

[14] O. Suria, "A statistical analysis of System Usability Scale (SUS) evaluations in online learning platform," *J. Inf. Syst. Informatics*, vol. 6, no. 2, pp. 992–1007, Jun. 2024, doi: 10.51519/journalisi.v6i2.750.

[15] Y. Thamilarasan *et al.*, "Enhanced System Usability Scale using the software quality standard approach," *Eng. Technol. Appl. Sci. Res.*, vol. 13, no. 5, pp. 11779–11784, Oct. 2023, doi: 10.48084/etasr.5971.

[16] A. F. Cobantoro and F. G. Anugra, "Designing parking system-based VB.Net and MySQL using Radio Frequency and Identification (RFID)," *Volt: J. Ilm. Pendidik. Tek. Elektro*, vol. 2, no. 2, pp. 109–116, Oct. 2017, doi: 10.30870/volt.v2i2.2009.

[17] J. R. Lewis, "The System Usability Scale: Past, present, and future," *Int. J. Hum.–Comput. Interact.*, vol. 34, no. 7, pp. 577–590, 2018, doi: 10.1080/10447318.2018.1455307.

[18] A. Bangor, P. T. Kortum, and J. T. Miller, "An empirical evaluation of the System Usability Scale," *Int. J. Hum.–Comput. Interact.*, vol. 24, no. 6, pp. 574–594, 2008, doi: 10.1080/10447310802205776.

[19] J. R. Lewis and J. Sauro, "Item benchmarks for the System Usability Scale," *J. Usability Stud.*, vol. 13, no. 3, pp. 158–167, May 2018. [Online]. Available: https://uxpajournal.org/wp-content/uploads/sites/7/pdf/JUS_Lewis_May2018.pdf

[20] J. Brooke, "SUS: A retrospective," *J. Usability Stud.*, vol. 8, no. 2, pp. 29–40, Feb. 2013.

[21] S. Andjelic, S. Obradovic, and B. Gacesa, "A performance analysis of the DBMS – MySQL vs PostgreSQL," *Commun. – Sci. Lett. Univ. Zilina*, vol. 10, no. 4, pp. 53–57, 2008, doi: 10.26552/com.C.2008.4.53-57.

[22] P. Kortum and C. Z. Acemyan, "How low can you go? Is the System Usability Scale range restricted?," *J. Usability Stud.*, vol. 9, no. 1, pp. 14–24, 2013.

[23] Z. Sharfina and H. B. Santoso, "An Indonesian adaptation of the System Usability Scale (SUS)," in *Proc. Int. Conf. Adv. Comput. Sci. Inf. Syst. (ICACSIS)*, 2016, pp. 145–148.

[24] C. Kartiko, "Evaluasi kualitas aplikasi web pemantau menggunakan model pengujian perangkat lunak ISO/IEC 9126," *J. Nas. Tek. Elektro Teknol. Inf. (JNTETI)*, vol. 8, no. 1, pp. 16–23, Feb. 2019. [Online]. Available: https://jurnal.ugm.ac.id/v3/JNTETI/article/download/2611/732

[25] A. Vetro, N. Zazworka, C. Seaman, and F. Shull, "Using the ISO/IEC 9126 product quality model to classify defects: A controlled experiment," in *Proc. 16th Int. Conf. Eval. Assess. Softw. Eng. (EASE 2012)*, 2012, pp. 187–196.