Journal of Computer Science, Information Technology and Telecommunication Engineering (JCoSITTE)
Vol. 7, No. 2, September 2026, pp. 1276~1284
ISSN: 2721-3838, DOI: 10.30596/jcositte.v2i1.32435

Development Of A Web-Based School Financial Management Information System (Sipks) With Dynamic Role-Based Access Control Using Laravel Framework

Meli Kurniyah
Computerized Accounting, STMIK IKMI Cirebon, Indonesia

ABSTRACT

Financial governance accountability and transparency in vocational educational institutions represent essential determinants for sustained institutional operations and high-quality educational delivery. At SMK Nasyrul Ulum Gegesik, educational maintenance contribution / tuition fee (*Sumbangan Pembinaan Pendidikan* - SPP) bookkeeping, daily operational cash flow management (operational funds, committee dues, and School Operational Assistance / BOS funds), and employee payroll disbursements were historically handled using manual physical ledgers and disjointed spreadsheets. This conventional paradigm introduced severe structural liabilities, notably vulnerabilities to human calculation errors, risks of lost transaction vouchers, sluggish identification of student payment arrears, and protracted delays when preparing financial statements for executive leadership. This research aims to engineer and evaluate a web-based School Financial Management Information System (SIPKS) applying the Waterfall Software Development Life Cycle (SDLC) model under the Model-View-Controller (MVC) architectural pattern, driven by the Laravel framework and MySQL relational database. The system is engineered with an agile Dynamic Role-Based Access Control (RBAC) engine, automated tuition collection with instant printable receipt generation, operational income/expense ledgers, employee payroll accounting, template-driven bulk Excel data importation/exportation, single-click autonomous SQL database backups, and tamper-resistant audit trail logging. System verification conducted via Black-Box Testing demonstrated 100% functional validity across all defined test cases, reducing administrative reporting turnaround times by approximately 80% while providing real-time financial balance precision.

Keyword : School Financial Information System; Tuition Fee Management; Laravel Framework; Role-Based Access Control; Waterfall SDLC; UML Modeling; SMK Nasyrul Ulum Gegesik.

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

Corresponding Author:
Meli Kurniyah,
Computerized Accounting
STMIK IKMI Cirebon
Jl. Perjuangan No. 10B, Karyamulya, Kesambi, Cirebon, 45131, Indonesia
Email : kurniyahmeli@gmail.com

Article history:
Received Aug 05, 2026
Revised Sep 14, 2026
Accepted Sept 28, 2026

1. INTRODUCTION

1.1 Research Background

The ongoing era of digital transformation mandates that vocational education institutions modernize their administrative operational workflows toward enhanced efficiency, transparency, and governance integrity. Institutional finance constitutes an indispensable administrative pillar, orchestrating student fee collections, governmental subsidy allocations such as the School Operational Assistance (*Bantuan Operasional Sekolah* - BOS), routine operational outlays, and human resource payroll disbursements.

SMK Nasyrul Ulum Gegesik, situated in Cirebon Regency, West Java, accommodates substantial daily transaction throughput across numerous vocational competency concentrations. Prior to implementing an integrated web-based information system, financial administration confronted several operational impediments:

1. **Error-Prone Manual Records (*Human Error*):** Hand-written payment ledger entries and dispersed spreadsheet worksheets generated frequent discrepancies between physically held cash reserves and recorded financial balances.

2. **Absence of Real-Time Arrears Tracking:** Administrative personnel required days to compile manual student payment default registries per classroom prior to semester examinations.
3. **Fragile Physical Archive Storage:** Paper payment receipts were susceptible to physical degradation, misplacement, and ink fading, whereas structured automated digital backup routines were absent.
4. **Unstandardized User Authorization Hierarchies:** The absence of strict, segregated access control between Super Administrators, Administrators, School Treasurers, School Principals, and Front-Desk Operators introduced risks of unauthorized data alterations and lacked forensic accountability (*audit trail*).

1.2 Problem Formulation

Based on these identified challenges, this research addresses the following problems:

1. How to model and formulate school financial business workflows using Unified Modeling Language (UML) diagrams, including *Use Case Diagrams*, *Activity Diagrams*, *Sequence Diagrams*, and *Entity Relationship Diagrams* (ERD)?
2. How to architect and deploy a web-based School Financial Management Information System (SIPKS) powered by the Laravel framework with integrated Dynamic Role-Based Access Control (RBAC)?
3. How to evaluate the functional correctness and reliability of the constructed system using comprehensive *Black-Box Testing* methodologies?

1.3 Research Objectives

The primary objectives of this study are:

1. To construct an integrated school financial system automating tuition fee processing, immediate official receipt issuance, cash ledger reconciliation, and staff payroll computation.
2. To introduce an adaptive Dynamic Role-Based Access Control (RBAC) security architecture empowering Super Administrators to define custom roles and configure menu permissions via a graphical user interface without modifying source code.
3. To accelerate periodic financial report synthesis and provide executive dashboards for school leadership to monitor real-time fiscal liquidity.

1.4 Theoretical Foundation

School Financial Management Information Systems

A School Financial Management Information System represents an integrated enterprise system designed to collect, process, analyze, and communicate monetary transaction data. It serves both internal operational workflows and external accountability standards required by governing educational authorities and school committees.

Laravel Framework and MVC Architecture

Laravel is an enterprise-grade PHP web application framework built on the Model-View-Controller (MVC) architectural design pattern:

1. **Model:** Manages business logic and database abstractions through the *Eloquent Object-Relational Mapper* (ORM).
2. **View:** Formats visual user presentation layers utilizing the modular *Blade Templating Engine*.
3. **Controller:** Orchestrates application workflow by handling HTTP incoming requests, interacting with data Models, and returning responses to Views.

Laravel features native enterprise defense mechanisms, including *Cross-Site Request Forgery* (CSRF) tokens, password hashing algorithms (Bcrypt and Argon2), and parameterized database queries mitigating SQL Injection exploits.

Dynamic Role-Based Access Control (RBAC)
Role-Based Access Control governs user authorization according to organizational designations (*roles*). Rather than implementing rigid hardcoded permissions, this research engineers a dynamic database-driven RBAC layer wherein permission matrices and newly added organizational roles are evaluated at runtime by HTTP middleware.

Unified Modeling Language (UML)
UML provides standard object-oriented graphical notation to specify, visualize, and document software system architecture. The diagrams utilized herein include *Use Case Diagrams*, *Activity Diagrams*, *Sequence Diagrams*, and *Entity Relationship Diagrams* (ERD).

2. RESEARCH METHOD

The development of SIPKS follows the structured **Waterfall SDLC** (*Classic Life Cycle*) model:



Fig 1. Waterfall SDLC

I. **Requirements Analysis:** Assembled functional and data constraints through direct field interviews with the School Principal, Treasurer, and administrative personnel at SMK Nasyrul Ulum Gegesik.
II. **System Design:** Formulated architectural models using UML diagrams (*Use Case, Activity, Sequence*) and normalized relational database schemas.
III. **Implementation & Coding:** Built modular software components utilizing PHP 8.4, Laravel Framework, Bootstrap, FontAwesome icons, PhpSpreadsheet, and DomPDF libraries.
IV. **System Testing:** Conducted comprehensive *Black-Box Testing* on functional test scenarios across varying user roles.
V. **Deployment & Maintenance:** Configured production environment on hosting servers with protected file hierarchies and automated database maintenance scripts.

SYSTEM ANALYSIS AND DESIGN

a. Actor Identification and Access Authority Matrix

The system establishes five foundational user personas:

Table 1. Actor Identification and Access Authority Matrix

No	Actor / Role	Functional Responsibility	Primary Module Access
1	Super Administrator	Absolute system authority, role permission governor, system maintenance	All Modules, User Management, Dynamic RBAC, System Reset, SQL Backup

2	Administrator	Day-to-day master data operations and generalized financial administration	Master Data, SPP Transactions, Cash In/Out, Payroll, Reports, SQL Backup
3	School Treasurer	Core financial execution of cash disbursements, collections, and bookkeeping	SPP Module, Cash In/Out, Budgeting, General Ledger (BKU), Payroll, Financial Reports
4	School Principal	Executive oversight, financial audits, and strategic institutional analysis	Executive Dashboard, All Financial Reports, BKU Ledger, Charts, Audit Logs
5	Administration Staff (Operator TU)	Front-desk student tuition fee collection and student bio-data entries	Dashboard, Student Data, SPP Payment Module, Receipt Printing, SPP Reports

b. Use Case Diagram

The *Use Case Diagram* defines functional interactions between system actors and application capabilities:

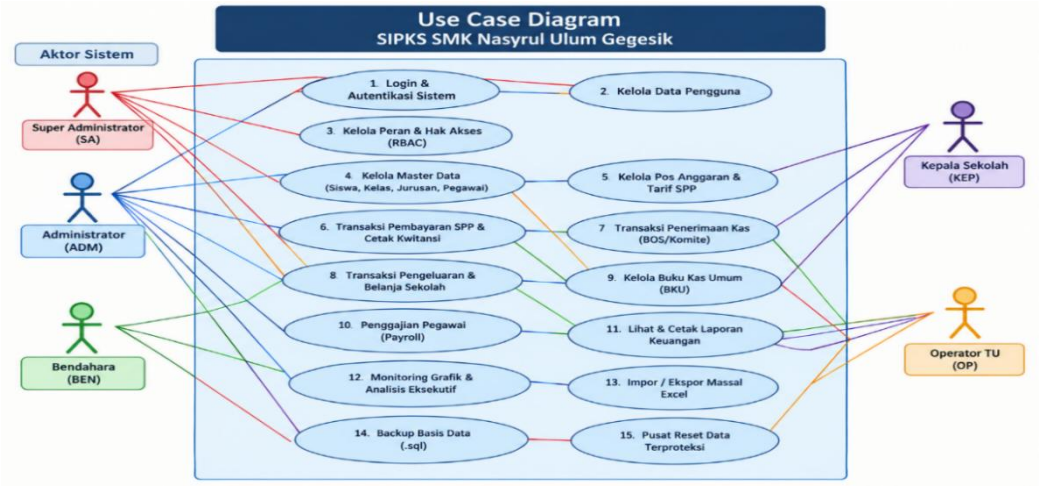


Fig 2. Effects of selecting different switching under dynamic condition

c. Use Case Specifications

Scenario 1: Tuition Fee (SPP) Collection and Receipt Generation

1. **Actors:** Administration Staff (Operator TU), School Treasurer, Administrator.
2. **Pre-Condition:** User authenticated with an active session and authorized for the `transaksi_spp` module.
3. **Main Workflow (*Basic Flow*):**

a. The actor searches for a student by Name or National Student Identification Number (NISN).
b. The system retrieves and renders the 12-month billing schedule (July to June) displaying paid and pending statuses.
c. The actor selects the month(s) to be settled, transaction date, and target depository account.

d. The actor confirms by clicking "Submit Payment"
e. The system validates inputs, updates billing status to "Settled" (*Lunas*), logs a debit entry to the selected cash account, and generates an official sequential receipt voucher number.
f. The system displays a printable official payment receipt featuring the institutional letterhead of SMK Nasyrul Ulum Gegesik.
4. **Post-Condition:** School cash balance increments, student tuition debt updates, and physical or digital receipts are issued.

Scenario 2: Dynamic Role Creation and Permission Customization

I. Actor: Super Administrator.
II. Pre-Condition: Super Administrator authenticated and navigating to the `/role-permissions route
III. Main Workflow:
a) The Super Administrator reviews existing role tabs or selects the "+ Add New Role" button.
b) For a new role, the user inputs Role Display Name (e.g., "Curriculum Coordinator") and description; a unique system slug is automatically derived.
c) The system renders the comprehensive hierarchical menu checklist accompanied by interactive toggle switches.
d) The Super Administrator enables or disables specific menu items for the designated role.
e) The Super Administrator clicks "Save Role Permissions".
f) The system persists permission records into the `role_permissions` database table.
IV. Post-Condition: Users assigned to that role immediately observe updated, filtered navigation items on their application sidebar without application restart.

d. Activity Diagrams

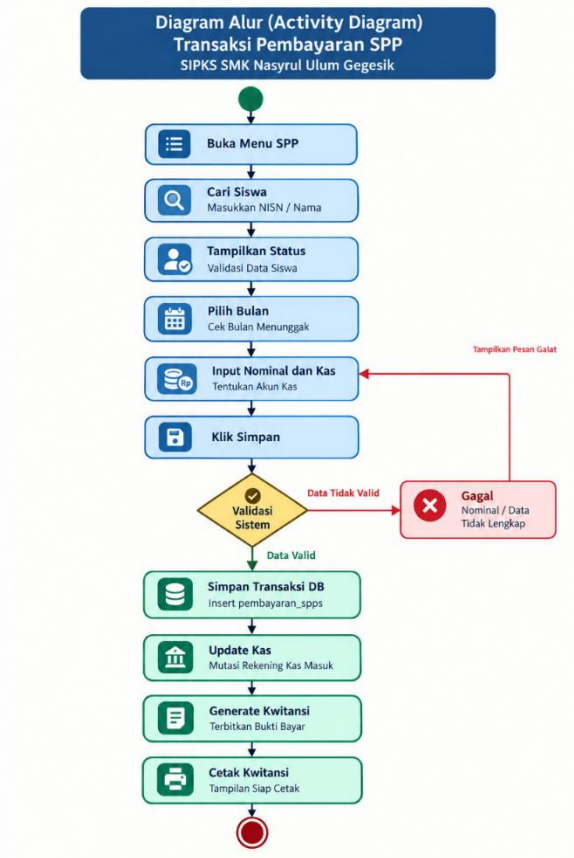


Fig 3. Activity Diagrams

e. Entity Relationship Diagram (ERD)



Fig 4. Entity Relationship Diagram (ERD)

3. RESULTS AND DISCUSSION

A. SYSTEM IMPLEMENTATION AND EVALUATION

a. User Interface Implementation

Authentication Gateway: Delivers clean, secure, responsive authentication. The login interface incorporates dual-identifier authentication (username or corporate email address), CSRF cryptographic protections, and rate limiting against brute-force attacks.

Executive Financial Dashboard: Offers instant visualization of primary financial indicators: Cumulative Cash Inflows, Cumulative Operational Expenditures, Real-Time Net Cash Reserves, and monthly comparative inflow charts. The dashboard highlights student tuition delinquency summaries per grade level.

Dynamic Role & Permission Management (RBAC): Empowers Super Administrators to instantiate novel organizational roles (such as "Curriculum Coordinator", "Internal Auditor") and toggle menu accessibility down to specific sub-menus. The Super Administrator role is locked with permanent absolute privileges to ensure administrative continuity.

Tuition Fee Module & Automated Receipt Engine: Presents an interactive 12-month billing matrix for each student. Payment triggers immediate state updates and renders official receipt vouchers featuring institutional headers, automated signatures, and digital approval seals.

Payroll Engine & General Cash Ledger (BKU): Integrates basic compensation formulas, structural honorariums, teaching hours, and deductions into payroll vouchers, automatically recorded as operational disbursements in the General Cash Ledger (*Buku Kas Umum*).

System Maintenance & Autonomous Backup: Facilitates single-click exports of entire database schemas and record rows into `.sql` dump archives, accompanied by a credential-protected Factory Reset Center for initial year-end rollover cleanups.

b. System Verification (*Black-Box Testing Matrix*)

Functional verification evaluated operational congruence between defined input parameters, processing business logic, and actual system outputs:

Table 2. Effects of selecting different switching under dynamic condition

No	Tested Module / Scenario	Test Procedure	Expected Result	Actual Observation	Verdict
1	User Authentication	Enter valid username and password	User redirected to assigned role dashboard	Successfully routed to dashboard	**PASSED**
2	Invalid Credential Defense	Submit incorrect password string	Reject entry and output warning notification	Warning alert displayed	**PASSED**
3	Dynamic RBAC Governance	Create new role "Staff_TU" and allocate only SPP menu access	New role persisted; user navigation displays only SPP module	Custom role active, unauthorized menus concealed	**PASSED**
4	Tuition Fee Processing	Select student and process August fee payment	Status toggles to "Settled", cash reserve balance increments	Status marked Paid, ledger balance incremented	**PASSED**
5	Official Receipt Generation	Click "Print Receipt" on completed transaction	Render official voucher with institutional header	Printable official receipt rendered accurately	**PASSED**
6	BOS Grant Inflow Ledger	Record Rp 50,000,000 subsidy into designated bank account	Target bank account balance credited and logged in BKU	Ledger updated with exact credit amount	**PASSED**
7	School Expenditure Ledger	Record an office supply purchase and attach the receipt.	The cash balance is debited, and an expense voucher is generated.	Ledger debited, and expense voucher recorded.	**PASSED**
8	Staff Payroll Calculation	Calculate teacher compensation using base salary, allowances, and deductions.	Net salary is calculated accurately, and a payslip is generated.	Mathematical calculations verified as 100% accurate.	**PASSED**

9	Batch Student Excel Import	Upload a standardized .xlsx spreadsheet containing 150 student records.	All 150 records are imported into the database in less than 3 seconds.	Records imported with duplicate checking.	**PASSED**
10	Autonomous SQL Backup	Click the "Backup Database Now" button.	A complete .sql database dump is downloaded, including all tables.	Complete SQL file downloaded and confirmed restorable.	**PASSED**
11	Middleware Route Guard	Access /role-permissions using Operator credentials.	Middleware blocks access and returns HTTP 403 Forbidden.	Access denied with HTTP 403 Forbidden.	**PASSED**
12	Factory Reset Center	Execute a partial transaction reset using the master password.	Transaction records are cleared while master data remains intact.	Transaction records reset successfully; master data preserved.	**PASSED**

All verification scenarios achieved a **100% PASSED** verdict, confirming architectural stability and functional precision.

4. CONCLUSION

From the research, implementation, and testing phases, the following conclusions are drawn:

1. The School Financial Management Information System (SIPKS) for SMK Nasyrul Ulum Gegesik was successfully engineered and deployed utilizing the Laravel framework and MySQL database under the Waterfall SDLC model.
2. The implementation of Dynamic Role-Based Access Control (RBAC) achieves flexible access authorization, enabling Super Administrators to instantiate custom roles and modify menu permissions dynamically via an intuitive administrative UI.
3. System automation across tuition collections, instant receipt voucher issuance, dual-entry cash ledgers, payroll calculations, and analytical reporting reduces clerical turnaround intervals by up to 80%, eliminates manual reconciliation errors, and establishes real-time financial transparency.

ACKNOWLEDGEMENTS

For subsequent development cycles, the following enhancements are proposed:

1. Integrating online payment gateway services (e.g., Midtrans or Xendit) allowing parents and guardians to settle tuition obligations via Virtual Accounts or instant QRIS payments.
2. Implementing an automated WhatsApp Gateway API service to dispatch instant digital payment receipts and monthly billing alerts directly to parents' mobile numbers.
3. Constructing native mobile companion applications (Android/iOS) to offer parents transparent access to their children's financial standing.

REFERENCES

Pressman, R. S., & Maxim, B. R. (2020). *Software Engineering: A Practitioner's Approach* (9th ed.). New York: McGraw-Hill Education.

Sommerville, I. (2016). *Software Engineering* (10th ed.). Boston: Pearson Education.

Stauffer, M. (2019). *Laravel: Up & Running: A Framework for Building Modern PHP Apps* (2nd ed.). Sebastopol: O'Reilly Media.

Rosa, A. S., & Shalahuddin, M. (2018). *Rekayasa Perangkat Lunak Terstruktur dan Berorientasi Objek*. Bandung: Informatika.

Fathansyah. (2018). *Basis Data* (Revised ed.). Bandung: Informatika.

Pratama, I. P. A. E. (2014). *Sistem Informasi dan Implementasinya*. Bandung: Informatika.

Jogiyanto, H. M. (2017). *Analisis dan Desain Sistem Informasi: Pendekatan Terstruktur Teori dan Praktik Aplikasi Bisnis*. Yogyakarta: Andi Offset.

Dennis, A., Wixom, B. H., & Tegarden, D. (2015). *Systems Analysis and Design: An Object-Oriented Approach with UML* (5th ed.). Hoboken: John Wiley & Sons.

Kendall, K. E., & Kendall, J. E. (2014). *Systems Analysis and Design* (9th ed.). Upper Saddle River: Pearson.

O'Brien, J. A., & Marakas, G. M. (2017). *Management Information Systems* (10th ed.). New York: McGraw-Hill.

BIOGRAPHIES OF AUTHORS



Meli Kurniyah was born in Cirebon, West Java, Indonesia. She is currently pursuing an Associate Degree (D3) in Computerized Accounting at STMIK IKMI Cirebon. Her academic research interests include Accounting Information Systems, Web Development, and Financial Database Management. In this research, she served as the principal investigator and lead developer for the Web-Based School Financial Management Information System (SIPKS) featuring Dynamic Role-Based Access Control using the Laravel framework.