Journal of Information Technology, Computer Engineering and Artificial Intelligence (2024)
Received June 2025 / Revised July 2025 / Accepted July 2025
Journal of Information Technology, Computer Engineering and Artificial Intelligence (ITCEA)
https://journal.redtechidn.org/index.php/itcea

Design and Development of a Client-Server Architecture Database Information System for Inventory Management and Transaction Efficiency

Afifah Trista Ayunda[1], Master Edison Siregar[2], Ephraim Eleazer Reva Manopo[3], Rizky Anggara Wicaksono[4], Jeremy Nugrahanto Kotten[5]

[1,3,4] *Information System, Pradita University, 15810, Indonesia*
[2] *Informatic Engineering, Pradita University, 15810, Indonesia*

Abstract. Transaction management and inventory control in automotive Micro, Small, and Medium Enterprises (MSMEs) often rely on manual record-keeping systems, leading to stock data discrepancies, accounting vulnerabilities, and operational inefficiencies. This study aims to design and implement a Client-Server database information system named Bengkelqu-AI (Accounting & Inventory) at Bengkel Zulva Motor. Data collection was conducted through structured interviews with the business owner to identify business entity requirements and daily transaction workflows. The system was developed using the PHP programming language, MySQL DBMS, Apache web server via the XAMPP package, and Visual Studio Code. The relational database design for *dbpenjualan* yielded five integrated tables (*tvendor*, *tcustomer*, *tbarang*, *tpembelian*, and *tbpenjualan*). The implementation results demonstrate that the system successfully facilitates data processing through Create, Read, Update, and Delete (CRUD) operations, automated report generation, and database backup features in .sql format. The deployment of the Bengkelqu-AI system is proven to enhance transaction recording accuracy, minimize potential losses due to human error, and simplify real-time inventory monitoring.

Keywords: Relational Database, Client-Server, Bengkelqu-AI, Inventory, Transaction, MySQL, PHP.

1. Introduction

Transaction management and spare part inventory control represent fundamental aspects in the operational execution of automotive service businesses. Inventory data accuracy and the reliability of financial recording significantly dictate operational efficiency and customer satisfaction levels (Sari and Nugroho 2022). Nevertheless, the majority of micro- and small-scale MSMEs still rely on conventional, paper-based, or memory-reliant bookkeeping mechanisms (Wulandari, Sagita, and Dwitiyanti 2021). This practice carries a high risk of triggering various operational challenges, such as discrepancies between physical stock counts and records, loss of sales summaries, and delays by business owners in evaluating monthly financial performance (Putri and Raharjo 2021).

Bengkel Zulva Motor, located on Jl. Puspiptek Raya No. 50, Serpong, South Tangerang, is a micro-enterprise operating in vehicle repair services and spare part sales with two employees. In executing its operations, Bengkel Zulva Motor lacks a digitally integrated transaction, inventory, and financial management system. The primary constraints encountered include fragmented transaction logging, the

[1]Corresponding author's email: afifahayunda22@gmail.com

inability to monitor inventory levels in real time, and the absence of structured financial reporting (Utomo, Triayudi, and Sholihati 2021).

To address these issues, a web-oriented database information system named Bengkelqu-AI (Accounting & Inventory) was developed. This system is designed to facilitate the integration of inventory management, sales and purchase transaction processing, operational report printing, and data security features via periodic backups (Fathansyah 2021). This paper aims to present the architectural analysis process, the relational database schema design, and the implementation results of the system at Bengkel Zulva Motor.

2. Methods

The methodology for the research and development of this system was executed through several structured stages:

Data Collection Method: Primary data collection was conducted via structured direct interviews with the primary stakeholder, namely the owner of Bengkel Zulva Motor (Mr. Sihna). The interviews focused on eliciting information regarding daily transaction workflows, spare part inventory management, business entity identification, and frequently encountered accounting challenges.

Database Architecture Analysis and Selection: A comparative analysis was performed across several database architecture types:

- *Centralized Architecture:* Considered unsuitable due to its reliance on a single point of failure, which risks paralyzing operations if the central server encounters disruptions (Fathansyah 2021).
- *Parallel and Distributed Architectures:* Excluded due to high capital expenditure requirements and excessive complexity for a micro-scale business.
- *Client-Server Architecture:* Selected as the optimal solution because it offers management simplicity, centralized security on the database server, and concurrent access flexibility for managers and employees via client computers (Kadir 2021).

Design and Development Environment:

- *Text Editor:* Visual Studio Code was utilized for writing PHP and HTML program code (Microsoft 2021).
- *Web Server & DBMS:* XAMPP (Apache and MySQL) was used to manage the local server environment and relational database system (Apache Friends 2021; Raharjo 2022).
- *Programming Language:* PHP was employed to handle server-side scripting and SQL query execution (Putri and Raharjo 2021).

Implementation and Functional Testing: Implementation encompassed building the *dbpenjualan* database schema, creating CRUD interfaces, developing report printing scripts (*Cetakpb.php*, *Cetakstok.php*, *Cetak.php*), and testing the database backup utility.

3. Results and Discussion

3.1 Relational Database Schema Design

In establishing an appropriate architectural foundation, a comparative analysis of database architectures was conducted (Fathansyah 2021; Kadir 2021). A centralized architecture was deemed inadequate because complete dependence on a single primary server creates a single point of failure vulnerability, risking total operational shutdown in the event of system failure. Conversely, parallel and distributed architectures were considered overly complex, requiring high infrastructure investment costs that exceed the operational needs of a small repair shop. Based on considerations of cost efficiency, ease of management, and security levels, the Client-Server architecture was chosen as the most rational

option (Kadir 2021). This model enables centralized data processing on the database server while allowing multiple client computers used by the owner and employees to access the application concurrently via a network.

The operational mechanism of the Client-Server architecture in Bengkelqu-AI is structured into three sequential phases. The first phase begins with the **Client Request** flow, where users execute operational actions—such as recording sales transactions, entering spare part purchases, or checking stock availability—through the web interface. The client application packages this request into structured instructions and transmits them to the server over the network. The second phase is **Server Processing**, where the server receives and verifies authorization and security levels, then executes SQL query instructions against the MySQL database (Raharjo 2022). Data within the database is updated or retrieved according to the received instructions. The third phase is the **Server Response**, where the server converts the processing results into a format intelligible to the client application and sends the response back across the network to be rendered on the user's display.

The technological stack utilized in developing this system comprises stable and popular open-source software. Visual Studio Code served as the Integrated Development Environment (IDE), supported by various PHP extensions such as IntelliSense and Debugger (Microsoft 2021). The local server environment was managed via XAMPP, integrating the Apache web server and MySQL DBMS (Apache Friends 2021). The PHP programming language was applied for server-side scripting, business logic execution, and SQL query processing to the MySQL relational database (Putri and Raharjo 2021).

Table 1. Technology Components Used

Category / Component	Specific Tool	Primary Function in the System
Text Editor / IDE	Visual Studio Code	Program code development (HTML, CSS, PHP) with support for IntelliSense, Debugger, and Formatter extensions (Microsoft 2021).
Web Server Environment	XAMPP (Apache)	Local server software for running the Apache environment and executing PHP queries locally (Apache Friends 2021).
Programming Language	PHP (Hypertext Preprocessor)	Server-side scripting language for business logic, form handling, and data communication (Putri and Raharjo 2021).
Database Management System	MySQL	Relational Database Management System (RDBMS) for storing, managing, and manipulating operational data (Raharjo 2022).

The storage structure in Bengkelqu-AI was designed as a relational database named *dbpenjualan*, comprising five primary tables (Putri et al. 2025). These tables include master entities (*tvendor*, *tcustomer*, and *tbarang*) and transaction entities (*tpembelian* and *tbpenjualan*). The *tvendor* entity stores supplier profiles with the primary key id_vendor, while *tcustomer* stores customer records with the primary key id_customer. Spare part categories and physical quantities are managed within the *tbarang* table, which stores item codes, item names, purchase prices, selling prices, and available stock quantities (Fikri et al. 2023). Meanwhile, complete historical procurement records are recorded in the *tpembelian* table, and sales and service histories are logged in the *tbpenjualan* table.

From the perspective of database normalization theory, the structure of the transaction tables (*tpembelian* and *tbpenjualan*) exhibits denormalization practices (Fathansyah 2021). Attributes such as nama_vendor, nama_barang, and harga_beli are stored redundantly within *tpembelian*, despite foreign key relationships with id_vendor and id_barang. In accounting and inventory information systems, this denormalization serves a crucial role in historical snapshotting. This guarantees that future

modifications to master data—such as selling price adjustments or supplier cost increases—do not retroactively alter completed historical transaction totals. However, this structure introduces potential update anomalies if an item or vendor name is updated in the master tables without application-level synchronization mechanisms.

Table 2. Database Schema Structure (*dbpenjualan*)

Table Name	Primary Function	Primary Key (PK)	Foreign Key (FK)	Attributes & Data Types
tvendor	Master supplier data	id_vendor int(5)	None	nama_vendor varchar(50), alamat varchar(50), no_telp int(12)
tcustomer	Master customer data	id_customer int(5)	None	nama_customer varchar(50), no_telp int(15)
tbarang	Spare part inventory master	id_barang int(5)	None	kode_barang varchar(5), nama_barang varchar(100), asal_barang varchar(100), harga_beli decimal(12,0), harga_jual decimal(12,0), jumlah_barang int(4)
tpembelian	Purchase transaction logs	id_pembelian int(11)	id_vendor int(5), id_barang int(5)	nama_vendor varchar(100), tanggal date, nama_barang varchar(100), harga_beli decimal(12,0), jumlah int(4), total int(12)
tbpenjualan	Sales transaction logs	id_penjualan int(11)	id_customer int(5), id_barang int(5)	nama_pelanggan varchar(20), tanggal date, nama_barang varchar(20), harga_jual decimal(12,0), jumlah int(20), total int(20)

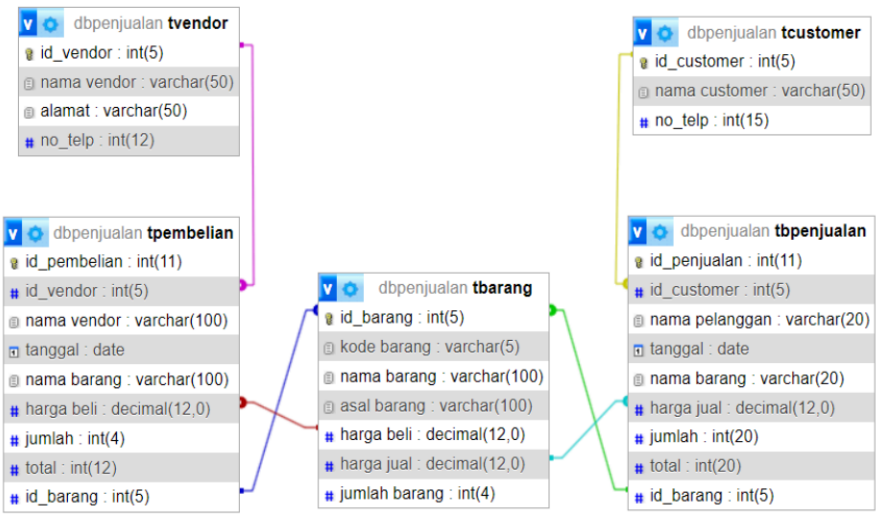

Figure 1. Database

The Bengkelqu-AI application supports daily operational management through a structured web interface (Dewi et al. 2021). Data manipulation is achieved via CRUD operations across three main modules. In the procurement module, the manager can log new inventory purchases, update cost prices, search transactions via keywords, and delete transaction logs via confirmation pop-ups. In the inventory module, the system provides input forms for new items, physical stock adjustments, and real-time margin tracking between purchase and selling prices. The sales module facilitates cashier transaction logging by recording customer names, purchased items, quantities, and automated total cost calculations.

To support financial transparency and data security, Bengkelqu-AI includes automated report printing and database backup mechanisms. Reports are accessible directly from module interfaces via dedicated action buttons executing specific formatting scripts: *Cetakpb.php* for purchase reports, *Cetakstok.php* for inventory reports, and *Cetak.php* for sales reports. Generated reports present concise data alongside official header details ready for printing or digital archiving. Meanwhile, database maintenance is managed via a dedicated module titled *Backup My Database (bmd)*. Through this interface, the system administrator enters local server credentials and the database name (*dbpenjualan*), then executes the backup action to export the database structure and records into a secure .sql file stored locally (Fathansyah 2021).

Table 3. Main System Modules and Operational Functions

Main Module	Interface / Script Files	Primary Operational Function
Purchase Logging	Pembelian.php Cetakpb.php	Processing supplier procurement data, searching transaction history, calculating subtotals, and printing purchase summaries.
Inventory Management	Stok.php Cetakstok.php	Inputting new items, monitoring physical stock counts, setting profit margins, and printing inventory reports.
Sales Logging	Penjualan.php Cetak.php	Recording sales transactions, logging customer details, calculating revenue, and printing sales receipts/reports.
Backup Utility	bmd/index.php	Backing up the complete structure and data of *dbpenjualan* into a compressed .sql file.

3.2 Client-Server Architectural Workflow

The Client-Server operational workflow of Bengkelqu-AI executes through three primary stages:

1. **Client Request:** The user (owner or employee) opens the web interface via a web browser on a client machine and inputs transaction data (e.g., recording a sale or adding stock). The client application sends the data request over the network to the server.
2. **Server Processing:** The Apache web server receives the request, validates authorization, and processes the SQL query directed to the MySQL DBMS (Apache Friends 2021; Raharjo 2022). The database executes the update operation.
3. **Server Response:** The server returns the processing state or query results to the client in an easily readable format (such as save confirmation or on-screen table updates).

3.3 Implementation of Operational Functionalities

The primary capabilities implemented within the Bengkelqu-AI system include:

- Transaction & Inventory Data Processing (CRUD):
    - *Procurement Data:* Facilitates logging purchase orders from vendors, keyword-based search queries, data updates, and record deletion with security pop-up confirmations.

  - *Inventory Data:* Manages spare part entities, purchase/selling price adjustments, and available stock level monitoring (Fikri et al. 2023).
  - *Sales Data:* Handles daily cashier entries with automated total price calculations based on item quantity (Dewi et al. 2021).
- Automated Report Generation: Users can print transaction summaries directly from each module using dedicated report action buttons. Specialized formatting scripts generate print-ready documents for Sales (*Cetak.php*), Procurement (*Cetakpb.php*), and Inventory (*Cetakstok.php*).
- Database Backup Utility: To ensure data integrity and system resilience against failures, Bengkelqu-AI integrates the *bmd* (Backup My Database) utility. Administrators can download .sql backup files directly through a simplified web interface (Fathansyah 2021).

4. Conclusions

The implementation of the Bengkelqu-AI database information system at Bengkel Zulva Motor demonstrates significant improvements in operational efficiency (Sari and Nugroho 2022). The choice of a Client-Server architecture provides flexible, secure, and centralized data access (Kadir 2021). The integrated *dbpenjualan* relational schema minimizes manual recording errors, prevents losses caused by inventory discrepancies, and accelerates precise report generation. Built-in backup utilities further bolster long-term data sustainability (Fathansyah 2021). For future enhancements, implementing database triggers is recommended to automatically deduct inventory stock upon logging sales transactions.

References

Apache Friends, 2021. *XAMPP Apache + MariaDB + PHP + Perl*. Available at: https://www.apachefriends.org/.

Dewi, I.A., Miftahuddin, Y., Fattah, M.A., Palenda, C.B. and Erawan, S.F., 2021. Point of sales system in InHome Café website using agile methodology. *Journal of Innovation and Community Engagement*, 1(1), pp.1–19.

Fathansyah, 2021. *Basis Data*. Edisi Revisi. Bandung: Informatika Bandung.

Fikri, M., Husain, B.M., Ndruru, I.P., Ndruru, F. dan Laiya, F., 2023. Rancang bangun sistem informasi persediaan barang berbasis website. *Jurnal Teknologi Komputer dan Informasi (JURTIKOM)*, 3(1), pp.25–34.

Kadir, A., 2021. *Konsep dan Tuntunan Praktis Basis Data*. Yogyakarta: Andi Publisher.

Microsoft, 2021. *Visual Studio Code: Code Editing. Redefined*. Available at: https://code.visualstudio.com/.

Panchadria, P.A., Ferawati, F. dan Santoso, S., 2022. Toko sembako terintegrasi payment gateway Midtrans berbasis Android. *Jurnal Tren Bisnis Global*, 2(2), pp.45–53.

Putri, M.P., Nadeak, E., Malahayati, Rahmi, N., Rini, A., Sari, D.N., Kurniati, Kusmiati, H. dan Pratama, R.A.A., 2025. *Sistem Manajemen Basis Data Menggunakan MySQL*. Bandung: Widina Media Utama.

Putri, R.A. dan Raharjo, B., 2021. Sistem informasi penjualan berbasis web pada UMKM menggunakan PHP dan MySQL. *Jurnal Sistem Informasi Indonesia*, 6(2), pp.101–110.

Raharjo, B., 2022. *Belajar Otodidak MySQL*. Edisi Kedua. Bandung: Informatika.

Ricardo, H. dan Fajrin, A.A., 2022. Aplikasi sistem pakar mendeteksi kerusakan pada mesin Toyota 4A-FE berbasis web. *Jurnal Comasie*, 6(2), pp.11–19.

Sari, I.P., Syahputra, A., Zaky, N., Sibuea, R.U. dan Zakhir, Z., 2022. Perancangan sistem aplikasi penjualan dan layanan jasa laundry sepatu berbasis website. *Blend Sains Jurnal Teknik*, 1(1), pp.20–28.

Sari, N.D. dan Nugroho, H., 2022. Pengaruh sistem informasi penjualan online terhadap efisiensi operasional UMKM. *Jurnal Ilmiah Informatika*, 7(3), pp.90–99.

Utomo, A., Triayudi, A. dan Sholihati, I.D., 2021. Point of sales menggunakan metode agile development pada Bengkel Mandala Motor. *Jurnal Teknologi Informasi*, 5(1), pp.15–23.

Wulandari, A., Sagita, S.M. dan Dwitiyanti, N., 2021. Perancangan sistem informasi pelayanan jasa pada Bengkel Las Listrik Mitra Baja Abadi. *Jurnal Riset dan Aplikasi Mahasiswa Informatika (JRAMI)*, 2(3), pp.412–419