# Brutor Functional Requirements

Motor Parts E-Commerce System — Functional Requirements

**Name:**

Barliso, Vincent Andri V.

Brutas Ericksson B.

**Section:**

BSIT-S-2A

**Submitted to:**

Mr. Dalisay

**Date:**

October 28, 2025

# 1. Introduction

1.1 Purpose

This document defines the functional requirements for Brutor, an SQL-backed e-commerce website for selling motorcycle parts. It focuses on achievable functionality for a university project using PHP and SQL as core technologies.

1.2 Scope

Brutor will support product cataloging, user and seller management, shopping cart and checkout, order history, reporting (sales, revenue, stock), and basic admin features. The system will be implemented as a PHP web application with a MySQL database.

# 2. Functional Modules & Requirements

Each module below lists concise functional requirements. Each requirement has an ID for quick reference. Tables below are organized by module to match the requested format.

## 2.1 Inventory Management

| ID | Requirement | Notes / Acceptance Criteria |
|---|---|---|
| REQ-INV-01 | CRUD for Products | Admin can create, read, update, delete product records: name, category, brand, model, price, condition, image, stock. |
| REQ-INV-02 | Product Categories | Support categories (e.g., Engine, Electrical, Bodywork, Consumables). Products must be filterable by category. |
| REQ-INV-03 | Stock for Materials | Only 'Materials' category tracks stock quantity. Stock is decremented when sold; restocks create expense entries. |
| REQ-INV-04 | Product Images | Each product stores an image so the UI displays assigned images instead of hard-coded resources. |
| REQ-INV-05 | Product Search & List | Customers can search/filter products; admin product list includes stock and edit buttons. |

### 2.2 Users & Authentication

| ID | Requirement | Notes / Acceptance Criteria |
|---|---|---|
| REQ-USER-01 | User CRUD (Customers) | Customers can register (sign up), login, update profile; admin can manage customer records. |
| REQ-USER-02 | Authentication & Roles | Two roles: 'Admin' and 'User'. Role enforcement for admin features (product/supplier/expense management). |
| REQ-USER-03 | Password Storage (Project Simplification) | For the class project: passwords stored as plain text if required by instructor; note security trade-offs. |
| REQ-USER-04 | Profile & Purchase History | Customer can view their past orders, displayed by transaction grouped by date/time. |

### 2.3 Suppliers & Sellers

| ID | Requirement | Notes / Acceptance Criteria |
|---|---|---|
| REQ-SUP-01 | Supplier CRUD | Admin can add/update suppliers (name, contact, lead time). |
| REQ-SUP-02 | Supplier Link to Products | Products may store an optional supplier reference for procurement tracking. |
| REQ-SUP-03 | Seed Data | Provide sample supplier entries for demo and grading. |

### 2.4 Orders, Cart & Checkout

| ID | Requirement | Notes / Acceptance Criteria |
|---|---|---|
| REQ-ORD-01 | Shopping Cart | Customers add products to a cart; for materials user sets quantity, for tools user sets rental dates. |
| REQ-ORD-02 | Merge Cart Lines | If the same material (same product) is added twice for the same dates, update quantity instead of creating duplicate rows. |

| | | |
|---|---|---|
| REQ-ORD-03 | Checkout & Transaction | At checkout create a Transaction record that references many line items (rental rows and/or materialsSold rows). |
| REQ-ORD-04 | Payments & Receipts | Record payment amount, compute change, create a downloadable receipt (TXT) and save a copy in the app 'receipts' folder. |
| REQ-ORD-05 | Stock Decrement | When a material is sold, decrement stock; for non-material rented items optionally decrement stock by 1 at checkout if desired. |

### 2.5 Rentals & MaterialsSold

| **ID** | **Requirement** | **Notes / Acceptance Criteria** |
|---|---|---|
| REQ-REN-01 | Rental Records | Store per-item rental entries: transactionID, customerID, productID, startDate, dueDate, rateCharged. |
| REQ-MAT-01 | Materials Sold Records | Store per-material sales: transactionID, productID, quantity, rateCharged. |
| REQ-REL-01 | Transaction Link | Both rentals and materials Sold link to the same Transaction table to allow one-to-many relationships per checkout. |

### 2.6 Payments & Expenses

| **ID** | **Requirement** | **Notes / Acceptance Criteria** |
|---|---|---|
| REQ-PAY-01 | Payment Recording | Record payments tied to a Transaction (amountPaid, paidOn). |
| REQ-EXP-01 | Expense CRUD | Admin can add expenses; expenses are deducted from revenue when computing net income. |
| REQ-EXP-02 | Expense vs. Restock | Restocking materials can optionally auto-create an Expense entry (purchase cost) to reflect cash outflow. |

### 2.7 Reporting & Analytics

| ID | Requirement | Notes / Acceptance Criteria |
|---|---|---|
| REQ-RPT-01 | Date-range Reports | Reports for rentals, materials sold, and overall transactions with filters for date range. |
| REQ-RPT-02 | Totals & Net Income | Compute total income (rentals, materials), total expenses, and net income for the selected range. |
| REQ-RPT-03 | Switchable Views | Reports UI offers toggle/buttons: show Rentals only, Materials only, or All Transactions (combined). |
| REQ-RPT-04 | Export | Allow export of report result as CSV or allow professor to view the grid during demo. |

### 2.8 Admin Utilities

| ID | Requirement | Notes / Acceptance Criteria |
|---|---|---|
| REQ-ADM-01 | Product Import/Seed | Provide SQL seed and a simple importer for sample products to ease demo setup. |
| REQ-ADM-02 | Restock Flow | Admin restock updates Product.Stock and (optionally) creates an Expense record for the restock cost. |
| REQ-ADM-03 | DB Backup Guidance | Provide SQL dump to restore schema and initial data when moving the project to another PC. |

## 3. Non-Functional Requirements (Brief)

• Implementation: PHP (backend), MySQL (database), HTML/CSS/JS for frontend; VB.NET/WinForms alternative for simplified local demo is acceptable per instructor.
• Security: For the class project, minimal authentication is acceptable; note that plain-text passwords are insecure.
• Performance: Designed for a single-branch demo; no heavy optimization required.

- Deliverables: SQL dump, project source (.sln or PHP files), README with setup steps, and sample seed data.

## 4. Acceptance Checklist (for Professor Demo)

| Feature | Demo Criteria |
|---|---|
| User sign-up / login | Create account, login as customer; admin login works. |
| Product CRUD | Admin can add/edit/delete product with image and stock (materials). |
| Add to cart & Checkout | Customer can add items, checkout, and receive saved receipt. |
| Reports | Run date-range reports showing totals and net income. |
| Data portability | Provide SQL dump and README; app loads sample data. |

## 5. Notes & Assumptions

- Keep the schema simple and normalized (3NF). Avoid premature optimization.
- Images stored as BLOB are portable across server restarts; ensure DB backups include BLOB data.
- If the instructor requires hashed passwords, implement hashing; this document documents both options.
- This FR is intentionally concise for an undergraduate course demo.