Reengineering Project Summary

SG Technologies POS System

2025-11-28

All Design Phases Complete – Ready for Implementation

Executive Summary

This document provides a comprehensive summary of the Software Reengineering project for the SG Technologies POS System, completed following the Software Reengineering Process Model.

1 Project Overview

Original System Legacy desktop Java application (Swing UI, file-based storage)

Reengineered System Modern web-based application (Spring Boot + React, PostgreSQL database)

Project Duration Fall 2025

Completion Status All phases designed and documented, ready for implementation

2 Completed Phases

2.1 ✓ Phase 1: Inventory Analysis (100%)

Deliverables: Complete asset inventory (20 Java classes, 9 database files), asset classification, dependency mapping, risk assessment
Key Findings: Minimal test coverage ($< 5\%$), file-based storage limitations, tight coupling

2.2 ✓ Phase 2: Document Restructuring (100%)

Deliverables: Documentation structure plan, data dictionary, operational scenarios, architecture diagrams
Achievements: Cataloged existing docs, identified gaps, created restructured documentation plan

2.3 ✓ Phase 3: Reverse Engineering (100%)

Deliverables: 456-line architecture analysis, 10+ code smells, 5+ data smells, workflow diagrams
Key Findings: God classes, long methods (80+ lines), duplicate code, magic numbers, no data normalization

2.4 ✓ Phase 4: Code Restructuring (60%)

Deliverables: 10 documented refactorings, `Constants` and `SystemUtils` classes, ∼130 lines eliminated
Quality Improvements: Zero behavior changes, significantly improved maintainability

2.5 ✓ Phase 5: Data Restructuring (100% – Design)

Deliverables: Normalized PostgreSQL schema (11 tables), DDL with constraints, migration strategy
Achievements: Full 3NF compliance, referential integrity, ACID support

2.6 ✓ Phase 6: Forward Engineering (100% – Architecture Design)

Deliverables: Layered architecture, RESTful API design, technology stack selection
Technology Stack: Spring Boot (Java) · React (TypeScript) · PostgreSQL · Maven + npm

3 Project Statistics

Code Metrics

- Legacy Source Files: 20 Java classes
- Refactored Files: 10 improved
- New Utility Classes: 2
- Lines Reduced: ∼130
- Test Coverage: ∼10% (improving)

Documentation

- Markdown Documents: 8
- Total Lines: ∼3,000+
- Refactoring Log: 10 entries
- Database Schema: 11 tables

Git Repository: `https://github.com/Sarmad-2005/ReEngineering_project.git` (10+ commits, all pushed)

4 Key Improvements

Architecture Improvements
1. Layered Architecture (clear separation of concerns)
2. Repository Pattern (data access abstraction)
3. Dependency Injection (loose coupling)
4. RESTful API (standard interfaces)
5. Modern Frameworks (Spring Boot + React)

Data & Code Quality Improvements

Normalized Schema (3NF)
Referential Integrity
Eliminated ∼130 duplicate lines

Centralized Constants (20+ values)
Utility Classes
Greatly improved readability

Security Improvements (Planned)

- Password Hashing (BCrypt)
- JWT Authentication
- Role-Based Access Control
- Audit Trail Logging
- Input Validation + Constraints

5 Deliverables Checklist

- ✓ Inventory Analysis document
- ✓ Document Restructuring plan
- ✓ Reverse Engineering analysis
- ✓ Refactoring log (10 refactorings)
- ✓ Data Restructuring plan + DDL
- ✓ Forward Engineering architecture
- ✓ Implementation status tracker
- ✓ Project summary (this document)

6 Next Steps – Implementation Phase

1. Set up Spring Boot + PostgreSQL project
2. Implement JPA entities and repositories
3. Build service layer with business logic
4. Create REST controllers and DTOs
5. Initialize React + TypeScript frontend
6. Write and execute data migration scripts

7 Risk Assessment

Completed Work Risks

✓ **Low** – All refactorings safe, tests passing, comprehensive documentation

Implementation Risks

⚠ **Medium** – Learning curve, data migration complexity, timeline

Mitigation: Incremental delivery, extensive testing, phased rollout, legacy system as fallback

8 Conclusion

The reengineering project has **successfully completed all analysis and design phases**. The system is now fully architected and ready for implementation.

All deliverables are documented, versioned, and publicly available at:
`https://github.com/Sarmad-2005/ReEngineering_project.git`

Ready for Full Implementation

Document Version	1.0
Date	2025-11-28
Status	All Design Phases Complete