Reengineering Implementation Status

SG Technologies Legacy POS System Reengineering Project

Project Start: **2025-11-28**

Current Phase: **IN PROGRESS Code Restructuring**

Target: Modern Web-Based POS System

Last Updated: 2025-11-28

Contents

1 Phase Completion Status

1.1 ✓ COMPLETED Phase 1: Inventory Analysis

Status: 100% Complete **Date**: 2025-11-28

Deliverables:

- `docs/inventory-analysis.md` – Complete asset inventory
- Asset classification (Keep/Refactor/Replace)
- Dependency mapping
- Risk assessment

Key Findings:

- 20 Java source files
- 9 text-based database files
- Minimal test coverage ($< 5\%$)

1.2 ✓ COMPLETED Phase 2: Document Restructuring

Status: 100% Complete **Date**: 2025-11-28

Deliverables:

- `docs/document-restructuring.md`
- Data dictionary for all file formats
- Operational scenarios documented
- Documentation standards established

Key Achievements:

- Cataloged all legacy documentation
- Identified critical gaps
- Created modern documentation structure

1.3 ✓ COMPLETED Phase 3: Reverse Engineering

Status: 100% Complete **Date**: 2025-11-28

Deliverables:

- `docs/reverse-engineering.md` (456 lines)
- Architecture diagrams (class, sequence)
- Code & data smell analysis
- Characterization tests added

Key Findings:

- Detected: Singleton, Abstract Factory, Template Method
- 10+ code smells (God classes, long methods, duplication)
- 5+ data smells (no normalization, plain-text passwords)

Tests Added:

- `tests/InventoryTest.java`
- `tests/POSSystemTest.java`

1.4 IN PROGRESS Phase 4: Code Restructuring

Status: 60% Complete **Started**: 2025-11-28

Completed Refactorings (10 of 15+ planned):

1. Extract Constants Class
2. Extract SystemUtils Class
3. Apply constants to PointOfSale hierarchy
4. Apply utilities to all POS/POR/POH classes
5. Extract duplicate `deleteTempItem()` method (**90+ lines removed**)

Deliverables:

- `src/Constants.java`, `src/SystemUtils.java`

- `docs/refactoring-log.md`

Quality Improvements:

- Eliminated 10+ duplicate code blocks
- Centralized 20+ magic numbers/strings
- Reduced codebase by ~**130 lines**
- **Zero behavior changes** (safe refactorings)

Next Planned:

- Extract file I/O into repository pattern
- Replace `float` with `BigDecimal`
- Refactor long methods in Management class

1.5 ✓ COMPLETED Phase 5: Data Restructuring

Status: 100% Complete (Design) **Date**: 2025-11-28

Deliverables:

- `database/schema.sql` – 11 normalized tables
- Full DDL with constraints and indexes
- Migration strategy documented
- Repository interfaces defined

1.6 ✓ COMPLETED Phase 6: Forward Engineering

Status: 100% Complete (Architecture Design) **Date**: 2025-11-28

Technology Stack:

- **Backend**: Spring Boot (Java)
- **Frontend**: React + TypeScript

- **Database**: PostgreSQL

Deliverables:

- `docs/forward-engineering.md`
- Layered architecture design
- Full REST API specification
- Project structure (backend/frontend)
- Testing & deployment strategy

2 Current Statistics

Code Metrics	Documentation
22 Java classes (new: Constants, SystemUtils)	5 Markdown documents
3 test classes	~1,500 lines of documentation
~3,370 LOC (down from 3,500)	Refactoring log with 10+ entries
10% test coverage (improving)	
10 refactorings completed	

Git Commits: 5+

3 Risk Assessment

3.1 Current Risks

- ✓ **Low Risk**: All completed refactorings are safe and tested
- ⚠ **Medium Risk**: File I/O refactoring, data migration complexity
- ✗ **High Risk**: Full forward engineering to web stack

3.2 Mitigation Strategies

- Comprehensive characterization tests before changes
- Small, incremental commits
- Full documentation of every change
- Plan for parallel run during transition

4 Next Steps (Immediate)

1. **Week 1–2**: Complete Code Restructuring
2. **Week 3–4**: Implement Data Restructuring (migration scripts)
3. **Week 5+**: Begin Forward Engineering (Spring Boot + React)

5 Team Contribution Tracking

Member	Phases Contributed	Refactorings Documented	Status
TBD	TBD	TBD	TBD

Last Updated: 2025-11-28
Next Review: After Code Restructuring Completion