Architectural Design Record (ADR)

Title: Legacy AS/400 Reporting System Modernization
Date: 2025-06-27
Status: In Progress

Context

This decision is about how we implement global reporting structure in the new application while integrating with legacy system. We are modernizing a legacy AS/400 (IBM i) system that currently uses SpoolFlex, an add-on application providing a 5250 terminal interface for spool file management. The existing system allows users to view/work with AS/400 spool files, convert to PDF/Excel formats, email reports, save to shared drives, and automate report generation through pre-defined jobs.

The modernization effort aims to replace the outdated 5250 terminal interface completely with a modern web-based user interface while maintaining equivalent functionality.

Decision

We will implement a hybrid reporting architecture that bridges the legacy AS/400 system with modern web technologies. We will keep all existing functionality in-place and simply wrap the current system. This structure will also support our second phase of re-writing the existing RPG reports in a modern framework within our Node.js REST API backend.

Core Components:

1. **Tabular Report Log SQL Table** - Central metadata repository:
    - Fields: Report Name, CreatedBy, isShared, Date, Time, LocationPath, Reference Number
2. **Shared Drive Storage** - File system location for converted report outputs
    - *Note: the client and the applications moving reports to the share will need the appropriate security access.
3. **Legacy System Wrapper Program** - Bridge component that accepts parameters (SpoolFileLocation, User, OutputPath, ConversionType), retrieves spool files, converts to desired format, generates timestamped filenames, saves to output path, and records metadata in SQL table
4. **Modern Backend API** - RESTful service with GET endpoints for report log retrieval, filtering by report type/user permissions, and shared/private access control
5. **Web Client Interface** - Browser-based UI component that displays available reports and opens them directly from shared drive locations. This should allow user to open the file folder location, open the report in a separate window, or popup download the report from the browser.

Reasoning

Report Log Benefits

The centralized report log provides a scalable foundation for future modernization. As we gradually migrate reports from the legacy system to native backend implementations, we can maintain the same process: write to the report log and save files to shared drive locations. This consistent approach ensures:

- **Unified reporting interface** regardless of report source (legacy or modern)
- **Error tracking capabilities** - failed report generations can be logged with error details
- **Audit trail** for all report activities across the entire system
- **Seamless migration path** without disrupting user workflows

Alternatives Considered

Alternative 1: Direct File System Access

- **Pros**: Users work directly with output files in storage locations
- **Cons**: Requires page-specific mappings to share drive locations (not scalable), vulnerable to storage location changes, no global repository of report activities
- **Rejected**: Duplicates Windows File Explorer functionality without added value

---

Consequences

Positive

- Gradual migration path without operational disruption
- Preserved functionality with modern user experience
- Scalable architecture supporting future report implementations
- Centralized error handling and audit capabilities
- Flexible output formats (PDF, HTML, CSV)

Negative

- Maintains dependency on legacy AS/400 system
- Additional system complexity and potential failure points
- Ongoing shared drive storage management requirements

Implementation Notes

1. Implement Report Log SQL table and wrapper program
2. Build RESTful backend service with filtering capabilities
3. Develop web client interface
4. Migrate automated jobs and optimize performance

Review Date: Now