# XML Processor Middleware Application - Functional Specification

## 1. Overview

The XML Processor Middleware Application is a web-based system designed to process and store XML documents containing Advanced Shipping Notice (ASN) data. The application provides a user-friendly interface for uploading XML files, validates them against predefined schemas, and stores the processed data in a relational database.

## 2. User Interface Requirements

### 2.1 Frontend Interface

- Modern, responsive web interface built with React

- Navigation bar with Upload, History, Transform, and UX sections

- File upload component for XML documents

- Status display for upload and processing operations

- Error messaging system for invalid uploads or processing issues

- Data visualization of processed ASN records

### 2.2 History View

- Table display of all processed files

- File name, processing date, and status information

- Status indicators (Success, Error, Warning)

- Number of records processed per file

- Error messages for failed processing attempts

- Auto-refresh capability (every 30 seconds)

### 2.3 Transform Page

- Split-panel interface showing XML elements and database fields

- Dynamic XSD structure loading and refresh capability

- Mapping creation between XML elements and database fields

- Configuration saving functionality

- Visual feedback for mapping operations

- Real-time validation of mapping rules

## 3. Functional Requirements

### 3.1 XML File Processing

- Support for various XML file formats

- Validation against XSD schemas

- Handling of different date formats

- Processing of both header and line item data

- Support for different XML structures and naming conventions

- Prevention of duplicate entries

### 3.2 Data Management

- Storage of ASN header information

- Storage of ASN line items

- Prevention of duplicate entries

- Data validation and error handling

- Support for various field types (dates, numbers, text)

- Transaction management

### 3.3 Mapping Configuration

- Dynamic XSD structure loading

- Real-time XSD refresh capability

- XML to database field mapping

- Mapping rule management (create, delete)

- Configuration persistence

- Validation of mapping rules

## 4. Business Rules

### 4.1 Data Validation Rules

- XML files must conform to specified XSD schemas

- Required fields must be present and valid

- Dates must be in acceptable formats

- Line items must be associated with valid headers

- Mapping rules must be valid and complete

### 4.2 Processing Rules

- Duplicate ASN entries should be prevented

- All line items must be associated with a header

- Data type conversions must maintain accuracy

- Error handling must be comprehensive

- Mapping rules must be applied consistently

## 5. Error Handling

### 5.1 User Feedback

- Clear error messages for invalid file uploads

- Processing status updates

- Validation failure notifications

- Database operation status feedback

- Mapping configuration error notifications

### 5.2 System Errors

- XML parsing errors

- Schema validation failures

- Database operation failures

- Network connectivity issues

- Mapping rule validation errors

## 6. Performance Requirements

- File upload size limits: up to 10MB

- Processing time: < 30 seconds per file

- Concurrent user support: up to 50 users

- Response time: < 2 seconds for UI operations

- History view refresh: every 30 seconds

## 7. Security Requirements

- Secure file upload handling

- Input validation and sanitization

- Protection against XML external entity (XXE) attacks

- Database security measures

- User authentication and authorization

## 8. Compliance Requirements

- Data format compliance with industry standards

- Proper error logging and tracking

- Audit trail for data modifications

- Data retention policies

- Mapping configuration versioning

## 9. Integration Points

- Frontend to Backend API communication

- Database connectivity

- File system interactions

- XSD schema management

- Mapping configuration management

## 10. Success Criteria

- Successful processing of various XML formats

- Accurate data storage and retrieval

- Proper handling of edge cases

- User-friendly interface and operation

- Reliable error handling and reporting

- Effective mapping configuration management

# XML Processor Middleware Application - Technical Specification

## 1. System Architecture

### 1.1 Frontend Architecture

- Framework: React with TypeScript

- State Management: React Context API

- UI Components: Material-UI (MUI)

- HTTP Client: Axios

- Development Server: Node.js

- Build Tool: Create Reacat App

### 1.2 Backend Architecture

- Framework: Spring Boot

- Language: Java 17

- Database: H2 (embedded)

- XML Processing: JAXP (Java API for XML Processing)

- Build Tool: Maven

- Server: Embedded Tomcat

### 1.3 System Components

- Frontend Application (Port 3000)

- Backend API Server (Port 8080)

- H2 Database Console (Port 8080/h2-console)

- File System Storage for XML/XSD

## 2. Frontend Implementation

### 2.1 Core Components

- App.tsx: Main application component and routing

- AuthContext.tsx: Authentication context provider

- Login.tsx: User authentication interface

- TransformPage.tsx: XML mapping configuration

- ProcessedFiles.tsx: History view component

- UploadPage.tsx: File upload interface

### 2.2 Features

- Protected routes with authentication

- Real-time file processing status

- Dynamic XML element mapping

- Auto-refreshing history view

- Responsive layout design

- Error handling and user feedback

### 2.3 API Integration

- Authentication endpoints

- File upload API

- XML processing status API

- Mapping configuration API

- History retrieval API

## 3. Backend Implementation

### 3.1 Controllers

- AuthController: Authentication management

- FileController: File upload handling

- MappingController: Mapping configuration

- ProcessingController: XML processing

- StatusController: Processing status

### 3.2 Services

- XmlProcessorService: XML file processing

- XsdService: XSD structure parsing

- MappingService: Mapping rule management

- AuthService: User authentication

- FileService: File operations

### 3.3 Models

- AsnHeader: ASN header information

- AsnLine: ASN line items

- ProcessedFile: File processing records

- MappingRule: XML-to-DB mapping rules

- User: User authentication data

## 4. Database Schema

### 4.1 Tables

- ASN\_HEADER

- id (PK)

- asn\_number

- shipment\_date

- supplier\_id

- supplier\_name

- ASN\_LINE

- id (PK)

- header\_id (FK)

- line\_number

- item\_number

- quantity

- uom

- description

- PROCESSED\_FILE

- id (PK)

- filename

- processed\_date

- status

- error\_message

- MAPPING\_RULE

- id (PK)

- xml\_path

- database\_field

- table\_name

- data\_type

- is\_attribute

- description

- USER

- id (PK)

- username

- password

- role

## 5. API Endpoints

### 5.1 Authentication

- POST /api/auth/login

- POST /api/auth/logout

- GET /api/auth/status

### 5.2 File Processing

- POST /api/upload

- GET /api/files/status/{id}

- GET /api/files/history

### 5.3 Mapping Configuration

- GET /api/mapping/xsd-structure

- GET /api/mapping/rules

- POST /api/mapping/rules

- DELETE /api/mapping/rules/{id}

- POST /api/mapping/save-configuration

## 6. Security Implementation

### 6.1 Authentication

- Session-based authentication

- Password encryption

- Role-based access control

- CSRF protection

- Secure session management

### 6.2 Data Security

- Input validation

- XML external entity prevention

- SQL injection protection

- File upload validation

- Secure error handling

## 7. Error Handling

### 7.1 Frontend Error Handling

- API error interceptors

- User feedback components

- Form validation

- File type validation

- Network error handling

### 7.2 Backend Error Handling

- Global exception handler

- Custom error responses

- Validation error handling

- File processing errors

- Database operation errors

## 8. Testing Strategy

### 8.1 Frontend Testing

- Component unit tests

- Integration tests

- End-to-end tests

- User interface testing

- Cross-browser testing

### 8.2 Backend Testing

- Unit tests for services

- Integration tests

- API endpoint testing

- XML processing tests

- Database operation tests

## 9. Deployment

### 9.1 Development Environment

- Local development setup

- H2 database configuration

- Development server settings

- Hot reload configuration

- Debug logging

### 9.2 Production Environment

- Production build process

- Database configuration

- Server deployment

- Environment variables

- Logging configuration

## 10. Monitoring and Maintenance

### 10.1 Logging

- Application logs

- Error tracking

- Performance monitoring

- User activity logging

- System health checks

### 10.2 Maintenance

- Database backups

- Log rotation

- Performance optimization

- Security updates

- Bug fixes and patches