# 

HP Data Quality ETL App Design Specification

|  |  |  |
| --- | --- | --- |
| **Author** | **Revision Number** | **Date** |
| Architect | 1.0 |  |
|  |  |  |
|  |  |  |

[HP Data Quality ETL App Design Specification 1](#_Toc476869617)

[Application Design Specification 3](#_Toc476869618)

[1. Design 3](#_Toc476869619)

[1.1 Overview 3](#_Toc476869620)

[1.2 Component Requirements 3](#_Toc476869621)

[1.2.1 Assemblies 3](#_Toc476869622)

[1.2.2 TopCoder Generic Components 3](#_Toc476869623)

[1.2.3 Third Party Components 3](#_Toc476869624)

[1.3 Application Management 4](#_Toc476869625)

[1.3.1 Transaction 4](#_Toc476869626)

[1.3.2 Configuration 4](#_Toc476869627)

[1.3.3 Persistence 4](#_Toc476869628)

[1.3.4 Thread-Safety and Concurrency 4](#_Toc476869629)

[1.3.5 Logging 4](#_Toc476869630)

[1.3.6 Auditing 4](#_Toc476869631)

[1.3.7 Exception Handling 4](#_Toc476869632)

[1.3.8 Internationalization 5](#_Toc476869633)

[1.3.9 Security 5](#_Toc476869634)

[1.3.10 Performance 5](#_Toc476869635)

[1.3.11 Scalability 5](#_Toc476869636)

[1.4 Deployment Constraints 5](#_Toc476869637)

[1.4.1 Technology overview 5](#_Toc476869638)

[1.5 Development Standards: 5](#_Toc476869639)

[1.5.1 Interfaces Classes Overview 5](#_Toc476869640)

[1.5.2 Changes to Existing System 5](#_Toc476869641)

[2. User Interface 5](#_Toc476869642)

[3. Included Documentation 5](#_Toc476869643)

[3.1 Architecture Documentation 5](#_Toc476869644)

[4. Future Enhancements 6](#_Toc476869645)

# Application Design Specification

# Design

Client wants to build an Extract Transform and Load (ETL) application that will allow user to move data between a source file or database and a destination directory or database. The goal of this challenge is to create the underlying architecture for this Data Quality app. The tool will allow moving file and relational data on a regular basis and will display the status of the currently scheduled and recently completed jobs.

This architecture defines backend and frontend for the app.

## Overview

The application will consist of the following:

* Spring REST controllers layer.
* Service layer (for jobs only) and DAO layer with the Spring Data JPA default implementation.

The application is built with Spring Boot (please check details [here](https://spring.io/guides/gs/spring-boot/)).

## Component Requirements

### Assemblies

* HP Data Quality ETL App Record DAO Backend Backend Assembly

This assembly is responsible for Record DAO interfaces and implementations (com.hp.etl.services.recorddao, com.hp.etl.services.recorddao.impl packages) as shown on UML Class Diagram.

* HP Data Quality ETL App Controllers and Services Backend Assembly

This assembly is responsible for the rest of implementation (all classes and interfaces except com.hp.etl.services.recorddao, com.hp.etl.services.recorddao.impl packages) as shown on UML Class Diagram.

* HP Data Quality ETL App Frontend Assembly

This assembly is responsible converting prototype into fully functional web app that accesses backend via REST API.

### TopCoder Generic Components

None

### Third Party Components

* Spring Framework 4.1.5

It is used for configuration and REST functionality.

* Apache log4j 1.2

This is used for logging.

* MySQL 5.7.x

This is the database system.

* Spring Boot 1.3.x

This is used for creating standalone Spring app.

* Apache POI 3.15

This is used for Excel reading/writing.

## Application Management

### Transaction

Spring declarative transactions will be used in the modifiable method. The method will be annotated with @Transactional.

### Configuration

Configuration will be done with Spring IoC.

The init method will be annotated with @PostConstruct and will throw ConfigurationException if any error in configuration like missing the required config parameter appears.

All configuration fields are marked as <<injected>> or @Autowired on class diagrams.

### Persistence

The default persistence will use Spring Data JPA specifications that will access the MySQL database.

### Thread-Safety and Concurrency

The backend services, Spring MVC controllers, interceptors and exception handlers should be effectively thread-safe. It can be assumed that the entities being persisted won't change during a persistence operation (for example, only one thread will work with a given entity instance at a time), but with that exception, the rest of the code will be able to be called from multiple threads. Also, the use of the IoC container to inject configurations will not be treated as a factor in thread-safety.

### Logging

The application will log activities and exceptions using Log4j logger.

Errors will be logged at ERROR level, and activity at DEBUG level, general information will be logged at INFO level.

Sensitive information like access token, user credentials and database connection credentials should not be logged.

Specifically, logging will be performed as follows, if logging is turned on.

- All exceptions will be logged at ERROR level, and automatically log inner exceptions as well.

o Format: Simply log the text of exception: [Error in method {className.methodName}: Details {error details}]

- All informational messages will be logged at INFO level.

- All debug messages will be logged at DEBUG level.

The logging will also provide a timestamp, but this is expected to be done automatically by the used logging mechanism, so explicit need to pass this information is not needed.

Diagnostic logging:

* Use some informational logging before and after major method invocations. Use judiciously, do not log too much or too little
* In the event of an error or exception, be liberal with diagnostic logging. When an exception is encountered, support staff need as much contextual information available as possible for troubleshooting.
* Consider writing performance log messages for long running processes
* Always use a common ID such as a “request id” in log messages to allow for collation of log messages written across multiple servers / log files for the same request.

### Auditing

Main entities (Job, Project) have audit related fields.

### Exception Handling

The architecture defines the following:

ConfigurationException – thrown if any error in configuration occurs.

HPDataQualityException – base exception, thrown if any error occurs.

EntityNotFoundException - indicates that the entity wasn’t found.

### Internationalization

No.

### Security

For authorization purpose all controller methods will be annotated with @PreAuthorize() that calls AuthorizationService.hasPermission. A single user may have one role:

-- Admin – creates and modifies users, has visibility to all Projects and Jobs in the System.

-- User - only has access to own projects and jobs.

For authorization the application extends WebSecurityConfigurerAdapter and configures it with the custom security components. The basic authentication will be used.

### Performance

The search API endpoints use pagination to improve the performance. The amount of joins DB in minimized to improve performance. Please see TCUML for details.

### Scalability

There is no particular scalability requirement, and the proposed architecture does not prevent the application from being scalable.

## Deployment Constraints

This application will be deployed to Tomcat 8 serving as an application container and Apache HTTP Server.

### Technology overview

* Java 8
* REST API
* MySQL 5.7.x
* Spring Framework 4.x <http://projects.spring.io/spring-framework/>
* Spring Boot 1.3.x <http://projects.spring.io/spring-boot/>
* Log4j 1.2 <http://logging.apache.org/log4j/1.2/>
* Apache POI 3.15 <https://poi.apache.org/download.html>

## Development Standards:

The assembly development must adhere to the guidelines as outlined in the [TopCoder Assembly Competition Tutorial](http://apps.topcoder.com/wiki/display/tc/Assembly+Competition+Tutorials).

## Interfaces Classes Overview

See the TCUML file.

## Changes to Existing System

None

# User Interface

None.

# Included Documentation

## Architecture Documentation

* Class Diagrams
* Sequence Diagrams
* Application Design Specification
* ERD
* Swagger API definition

# Future Enhancements

None.