|  | **Cognizant Academy**  **Courier Tracking System**  **JDBC Specification Document**  **Version 1.0** |
| --- | --- |
| |  | **Prepared By / Last Updated By** | **Reviewed By** | **Approved By** | | --- | --- | --- | --- | | **Name** |  |  |  | | **Role** |  |  |  | | **Signature** |  |  |  | | **Date** |  |  |  | |
|  |

**Table of Contents**

[**1.0**](#_heading=h.1fob9te) **Introduction 3**

[**1.1**](#_heading=h.3znysh7) **Purpose of this document 3**

[**1.2**](#_heading=h.2et92p0) **Definitions & Acronyms 3**

[**1.3**](#_heading=h.tyjcwt) **Project Overview 3**

[**1.4**](#_heading=h.1t3h5sf) **Scope 3**

[**1.5**](#_heading=h.4d34og8) **Intended Audience 3**

[**1.6**](#_heading=h.17dp8vu) **Hardware and Software Requirement 3**

[**2.0**](#_heading=h.3rdcrjn) **Class Diagram 5**

[**2.1**](#_heading=h.26in1rg) **Data Access Layer 5**

[**2.2**](#_heading=h.lnxbz9) **Db.java 6**

[**3.0**](#_heading=h.35nkun2) **DAO for Registration and Validation of Admin and Staff. 7**

[**3.1**](#_heading=h.1ksv4uv) **AdminStaffDAOImple.java 7**

[**4.0**](#_heading=h.44sinio) **DAO for Registration, Validation and Retrieving package details of User 8**

[**4.1**](#_heading=h.2jxsxqh) **UserDAOImple.java 8**

[**5.0**](#_heading=h.z337ya) **DAO for Insertion and Updation of Package details 9**

[**5.1**](#_heading=h.1y810tw) **PackageDAOImple.java 9**

[**6.0**](#_heading=h.4i7ojhp) **DAO for Insertion and Validation of Warehouse details 9**

[**6.1**](#_heading=h.2xcytpi) **WarehouseDAOImple.java 9**

[**7.0**](#_heading=h.3whwml4) **Standards and Guidelines 10**

[**7.1**](#_heading=h.2bn6wsx) **DAO 10**

[**8.0**](#_heading=h.qsh70q) **Submission 10**

[**8.1**](#_heading=h.3as4poj) **Code submission instructions 10**

[**9.0**](#_heading=h.1pxezwc) **Change Log 11**

# Introduction

## Purpose of this document

The purpose of this document is to define the JDBC module implementation for Courier Tracking System project.

## Definitions & Acronyms

| **Definition / Acronym** | **Description** |
| --- | --- |
| DAO | Data Access Object |
| JDBC | Java Database Connectivity |

## Project Overview

Refer Courier Tracking System-use-case-specification.docx for understanding the functionality and features.

## Scope

1. Creation of DAO classes and methods for reading and persisting data of Courier Tracking System application.

## Intended Audience

* Product Owner
* Scrum Master
* Application Architect
* Project Manager
* Test Manager
* Development Team
* Testing Team

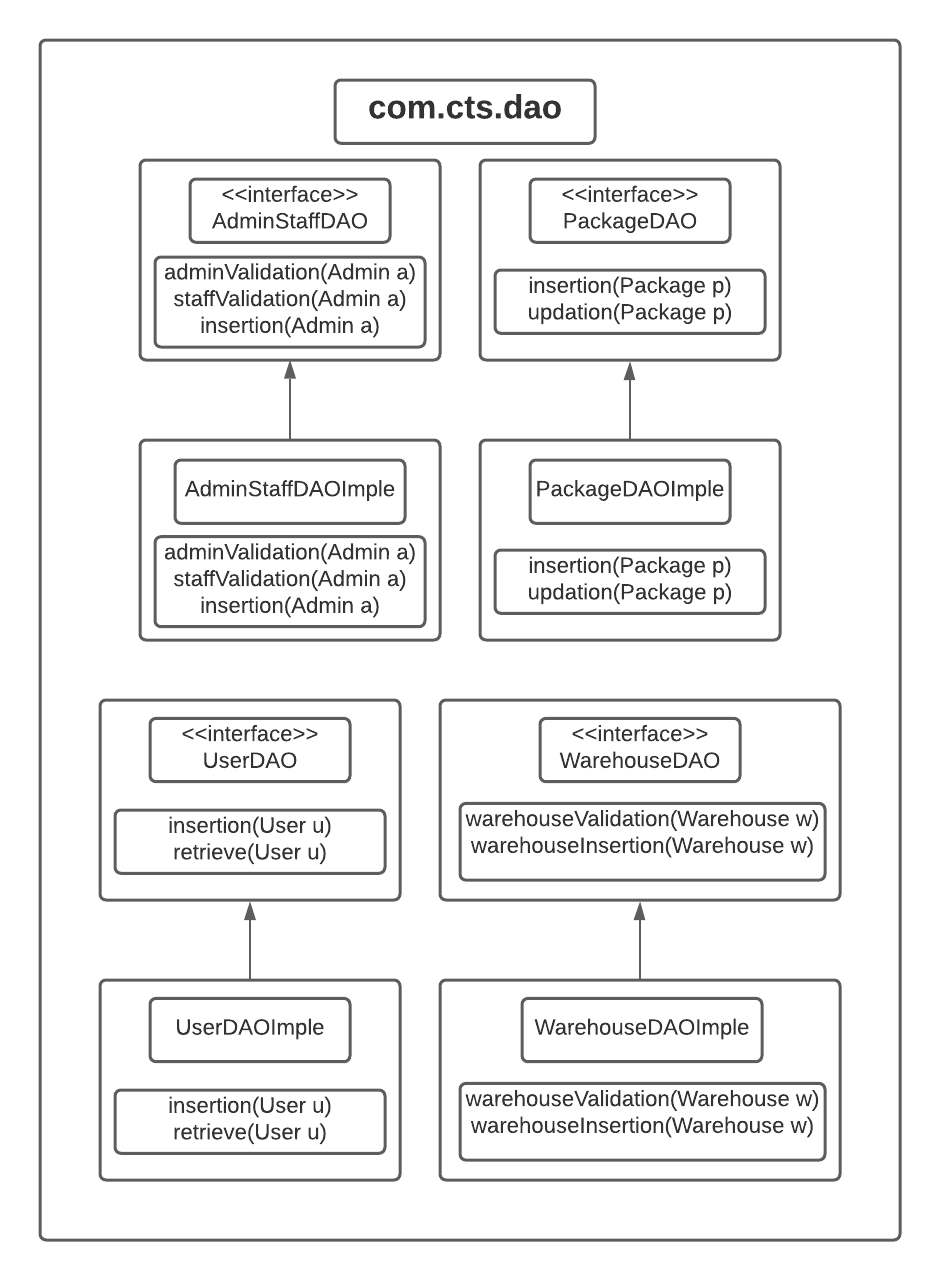
## Hardware and Software Requirement

1. Hardware Requirement:
   1. Developer PC with 4GB Ram
2. Software Requirement
   1. Git
   2. JDK 1.8
   3. Eclipse IDE for Enterprise Java Developers 2019-03 R
   4. MySQL Community Server 8.0

# Class Diagram

## Data Access Layer

Refer to the diagram below and create classes accordingly.

 Arrows represent implementation of an interface.

Make note that getConnection is a static method.

Highlighted classes are the ones that need to be implemented in this specification.

## Db.java

This class will be used by each Dao implementation class for getting the database connection.

The connection details have to be stored in a Db.java class file. Refer details below:

driver= com.mysql.jdbc.Driver

connection-url= jdbc:mysql://localhost:3306/cts

user=root

password=root

**static getDb(): Connection**

1. Using Class.forName() method set mysql as JDBC Driver.
2. Using getConnection method of DriverManager class set connection with database in mysql using “root” as username and “root” as password.

# DAO for Registration and Validation of Admin and Staff.

## AdminStaffDAOImple.java

**public int adminValidation(Admin a)**

1. Get connection using Db class.
2. Using PreparedStatement execute the select query that retrieves all the records from staff table where loginType=’A’
3. Iterate through the ResultSet
4. For each item in the ResultSet check the retrieved email and password with entered email and password and return a corresponding int value.

**public int staffValidation(Admin a)**

1. Get connection using Db class.
2. Using PreparedStatement execute the select query that retrieves all the records from staff table where loginType=’S’
3. Iterate through the ResultSet
4. For each item in the ResultSet check the retrieved email and password with entered email and password and return a corresponding int value.

**public int insertion(Admin a)**

1. Get connection using Db class.
2. Using PreparedStatement execute the select query that inserts all the records into staff table
3. Using executeUpdate get an int value and return it.

# DAO for Registration, Validation and Retrieving package details of User

## UserDAOImple.java

**public int insertion(Admin a)**

1. Get connection using Db class.
2. Using PreparedStatement execute the select query that inserts all the records into user table
3. Using executeUpdate get an int value and return it.

**public int validation(User u)**

1. Get connection using Db class.
2. Using PreparedStatement execute the select query that retrieves all the records from user table
3. Iterate through the ResultSet
4. For each item in the ResultSet check the retrieved email and password with entered email and password and return a corresponding int value.

**public ArrayList<Package> retrieve(User u)**

1. Initialize an ArrayList of Package.
2. Get connection using Db class.
3. Using PreparedStatement execute the select query that retrieves all the records from package table where customerId is equals to user entered customerId.
4. Iterate through the ResultSet
5. Instantiate a Package object and add retrieved data into it using constructor injection
6. Add the package object into ArrayList and return the List.

# DAO for Insertion and Updation of Package details

## PackageDAOImple.java

**public int insertion(Package p)**

1. Get connection using Db class.
2. Using PreparedStatement execute the select query that inserts all the records into package table
3. Using executeUpdate get an int value and return it.

**public int updation(Package p)**

1. Get connection using Db class.
2. Using PreparedStatement execute the select query that updates all the records into package table
3. Using executeUpdate get an int value and return it.

# DAO for Insertion and Validation of Warehouse details

## WarehouseDAOImple.java

**public int warehouseInsertion(Warehouse w)**

1. Get connection using Db class.
2. Using PreparedStatement execute the select query that inserts all the records into warehouse table
3. Using executeUpdate get an int value and return it.

**public int warehouseValidation(Warehouse w)**

1. Get connection using Db class.
2. Using PreparedStatement execute the select query that retrieves all the records from warehouse table
3. Iterate through the ResultSet and For each item check the retrieved branchId with entered branchId and return a corresponding int value.

# Standards and Guidelines

## DAO

1. All Java coding standards are applicable
2. Read database connection details from Db.java file

# Submission

## Code submission instructions

Once your code is evaluated by the trainer and all the issues reported by the trainer are corrected, the code needs to be submitted to the remote repository. Follow the steps below to submit the code to remote repository.

1. In Windows Explorer go to the Courier Tracking System folder
2. Right click on the empty space in the right hand side of Windows Explorer and select “Git Bash here”
3. Execute the following commands

To display the added or modified files

git status

To stage the added or modified files

git add .

To display the staged files

git status

To save the code to local repository

git commit -m "jdbc"

To transfer the changes from local machine to server

git push origin master

1. Successful execution of the above commands will upload the files to the server repository.
2. Login into <https://code.cognizant.com>
3. Click on the project Courier Tracking System
4. Check if the files that are uploaded correctly with appropriate folder structure.

# Change Log

|  | **Changes Made** | | | |
| --- | --- | --- | --- | --- |
| V1.0.0 | Initial baseline created on <dd-Mon-yy> by <Name of Author> | | | |
| Vx.y.z | <Please refer the configuration control tool / change item status form if the details of changes are maintained separately. If not, the template given below needs to be followed> | | | |
| **Section No.** | **Changed By** | **Effective Date** | **Changes Effected** |
|  |  |  |  |