|  |  |
| --- | --- |
|  | **Cognizant Academy**  **truYum**  **Servlet & JSP Specification Document**  **Version 1.0** |
| |  |  |  |  | | --- | --- | --- | --- | |  | **Prepared By / Last Updated By** | **Reviewed By** | **Approved By** | | **Name** | Anshul Bias, Ankitha, Raaghul V, Shreya Sree, Yasaschandra |  |  | | **Role** | Programmer Analyst Trainee |  |  | | **Signature** |  |  |  | | **Date** | 18 Jan 2022 |  |  | |
|  |

Table of Contents

[1.0 Introduction 4](#_Toc12281353)

[1.1 Purpose of this document 4](#_Toc12281354)

[1.2 Definitions & Acronyms 4](#_Toc12281355)

[1.3 Project Overview 4](#_Toc12281356)

[1.4 Scope 4](#_Toc12281357)

[1.5 Intended Audience 4](#_Toc12281358)

[1.6 Hardware and Software Requirment 4](#_Toc12281359)

[1.7 Eclipse Configuration for Tomcat Server 5](#_Toc12281360)

[1.7.1 Add Tomcat Server to Eclipse 5](#_Toc12281361)

[1.7.2 Add Tomcat library to eKart project 5](#_Toc12281362)

[2.0 Class Diagram 7](#_Toc12281363)

[2.1 Servlets 7](#_Toc12281364)

[3.0 Design for View Menu Item List Admin (TYUC001) 8](#_Toc12281365)

[3.1 Sequence Diagram 8](#_Toc12281366)

[3.2 Servlet 8](#_Toc12281367)

[3.3 JSP 9](#_Toc12281368)

[4.0 Design for View Menu Item List Customer (TYUC002) 10](#_Toc12281369)

[4.1 Sequence Diagram 10](#_Toc12281370)

[4.2 Servlet 10](#_Toc12281371)

[4.3 JSP 11](#_Toc12281372)

[5.0 Design for Edit Menu Item (TYUC003) 11](#_Toc12281373)

[5.1 Sequence Diagram (Show Edit Menu Item Form) 11](#_Toc12281374)

[5.2 Sequence Diagram (Edit Menu Item) 12](#_Toc12281375)

[5.3 Servlet 13](#_Toc12281376)

[5.4 JSP (edit-menu-item.jsp) 13](#_Toc12281377)

[5.1 JSP (edit-menu-item-status.jsp) 14](#_Toc12281378)

[6.0 Design for Add Cart (TYUC004) 15](#_Toc12281379)

[6.1 Sequence Diagram 15](#_Toc12281380)

[6.1 Servlet 16](#_Toc12281381)

[6.1 JSP (menu-item-list-customer.jsp) 16](#_Toc12281382)

[7.0 Design for View Cart (TYUC005) 17](#_Toc12281383)

[7.1 Sequence Diagram 17](#_Toc12281384)

[7.2 Servlet 18](#_Toc12281385)

[7.3 JSP (cart.jsp) 18](#_Toc12281386)

[7.1 JSP (cart-empty.jsp) 18](#_Toc12281387)

[8.0 Design for Remove Item from Cart (TYUC006) 20](#_Toc12281388)

[8.1 Sequence Diagram 20](#_Toc12281389)

[8.2 Servlet 21](#_Toc12281390)

[8.3 JSP (cart.jsp) 21](#_Toc12281391)

[9.0 Standards and Guidelines 21](#_Toc12281392)

[9.1 JSP 21](#_Toc12281393)

[9.2 Servlets 22](#_Toc12281394)

[10.0 Submission 22](#_Toc12281395)

[10.1 Code submission 22](#_Toc12281396)

[11.0 Change Log 23](#_Toc12281397)

# Introduction

## Purpose of this document

The purpose of this document is to define the server side implementation of the truYum application.

## Definitions & Acronyms

|  |  |
| --- | --- |
| Definition / Acronym | Description |
| JSP | Java Server Pages |
| JSTL | Java Standard Tag Library |

## Project Overview

Refer truYum-use-case-specification.docx for understanding the functionality and features.

## Scope

1. Creation of JSP and Servlets for Courier Tracking System project

## Intended Audience

* Scrum Master
* Project Manager
* Test Manager
* Development Team
* Testing Team

## Hardware and Software Requirment

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. Apache Tomcat 9

## Eclipse Configuration for Tomcat Server

### Add Tomcat Server to Eclipse

In this module the eKart application will be deployed and tested in Tomcat Server. Tomcat is integrated with Eclipse and the following steps will guide for configuring Tomcat Server in Eclipse.

1. Extract Tomcat 9 zip file to D:
2. Open Eclipse
3. Select Window > Show View > Servers
4. Click on the link “No servers are available. Click this link to create a new server..”
5. Expand “Apache” and select “Tomcat v9.0 Server”
6. Click “Next”
7. Click “Browse” and select the folder where Tomcat 9 had been extracted
8. Click “Next” and then click “Finish”
9. Now the Tomcat Server will be visible in the “Servers” view.

### Add Tomcat library to Courier Tracking System project

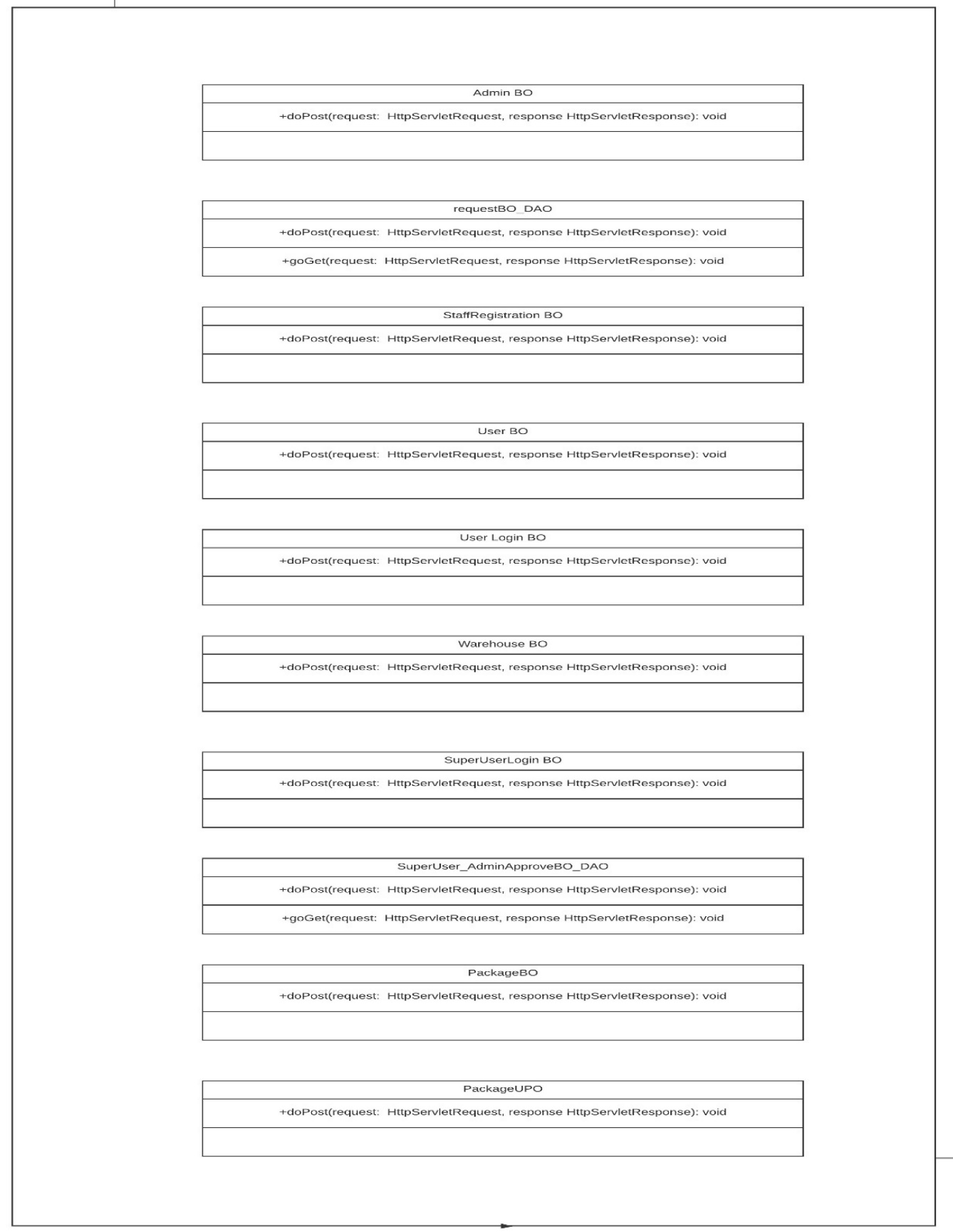
Once the Tomcat Server is configure in Eclipse, we need to ensure that the Tomcat library is available to truYum project. Follow the steps below to get this done.

1. Right click on truYum project and select the options below
2. Properties > Build Path > Libraries > Add Library .. > Server Runtime > Next > Apache Tomcat v9.0
3. Click “Apply & Close”

# Class Diagram

## Servlets

Refer the diagram below and create classes accordingly. In addition to the classes below, all the classes defined in Java Specification Document as well is applicable.

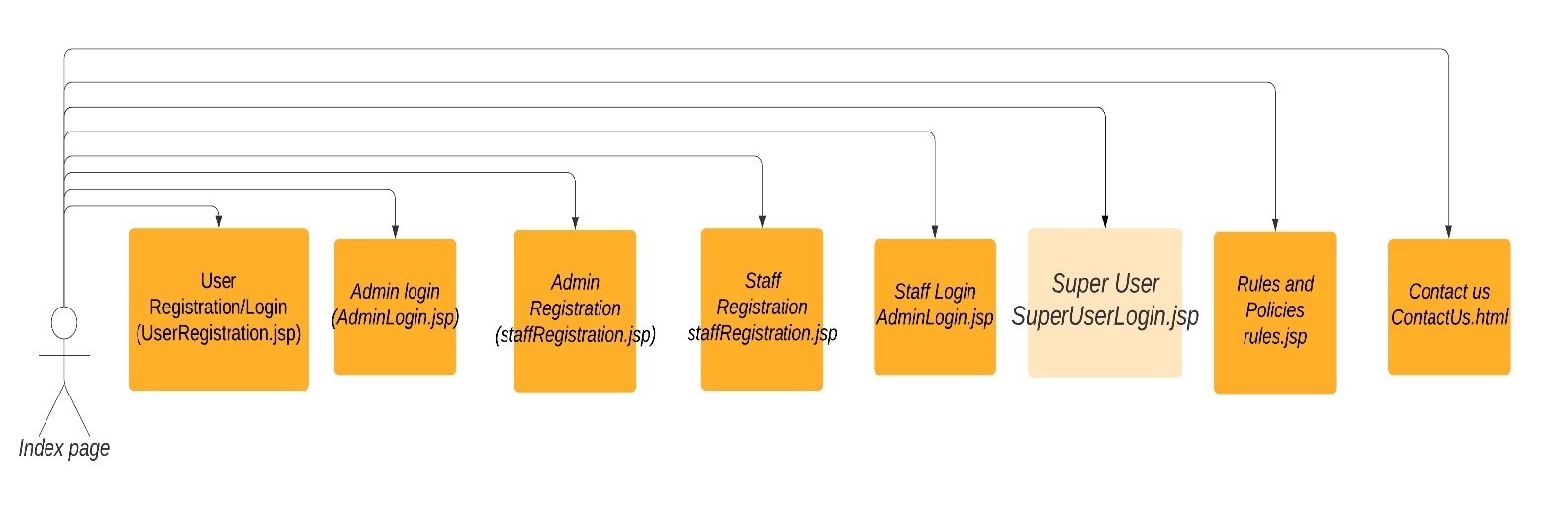


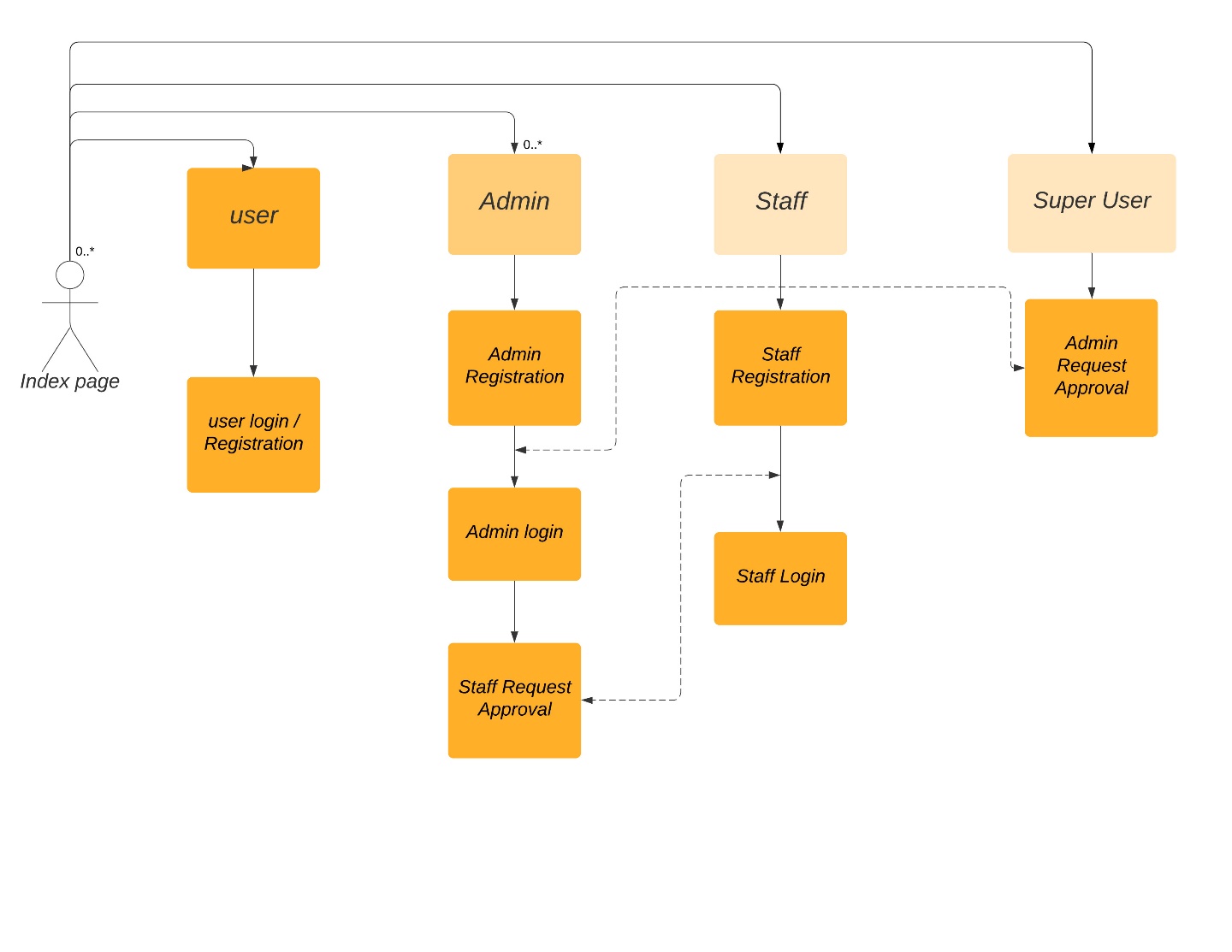
com.cts.bo

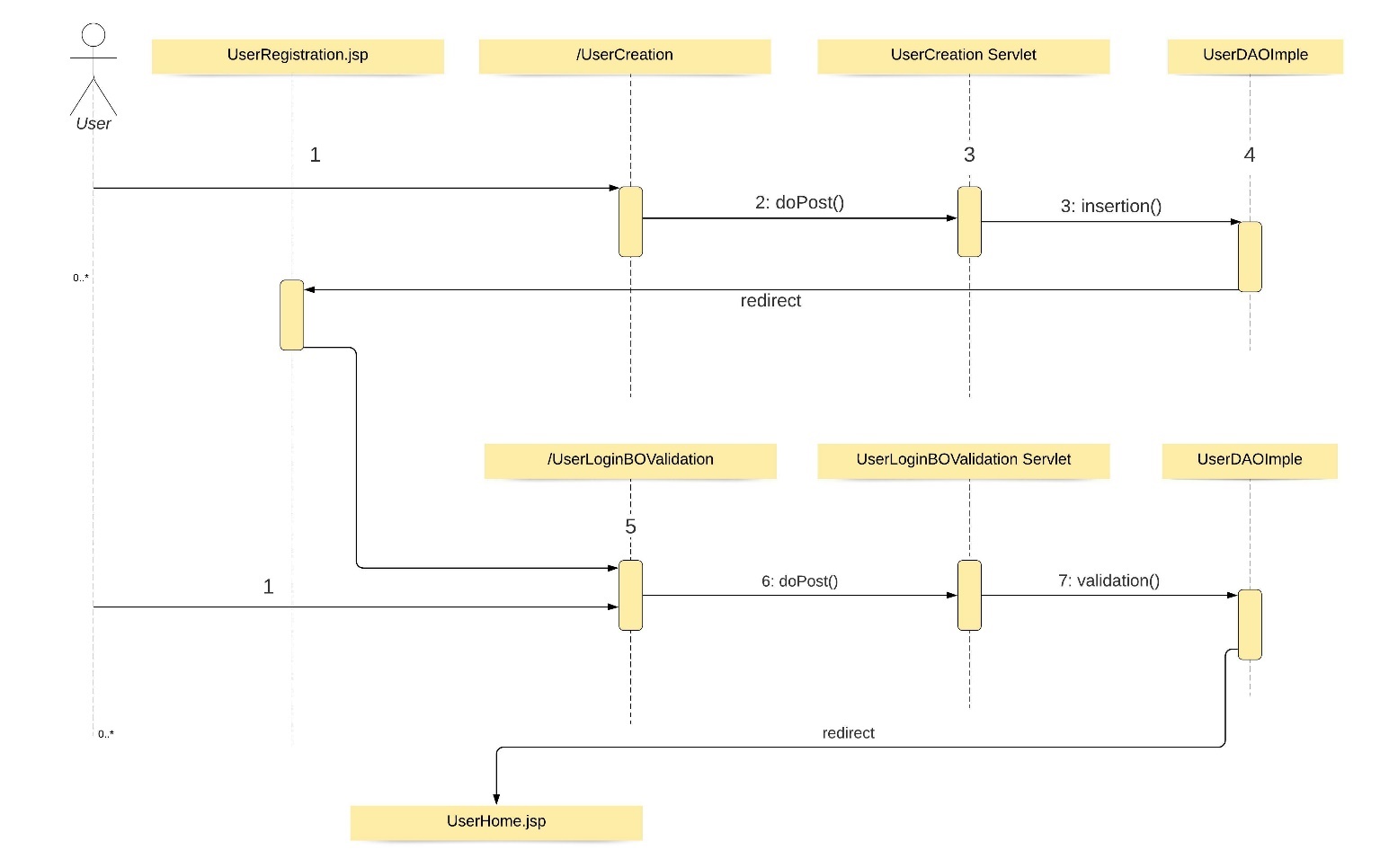
# Design for users in UserRegistration page

## Sequence Diagram

The following diagram explains the flow when one clicks on the specific URL. Refer steps included after the sequence diagram.







**Steps Explanation**

1. User access the /UserCreation URL through UserRegistration.jsp
2. If new user fills the details and click registers, /UserCreation URL is executed.
3. The servlet receives the request in doPost() method.
4. Instantiate User by getting parameters through getParameters() and setting it.
5. Later the refernce variable is used in the instance of UserDaoImpl and insertion() method is used.
6. Data gets stored in database by the UserDao and redirects to UserRegistration.jsp for loggining in.
7. The user provides his credentials, and when the login button is clicked, / UserLoginBOValidation is executed.
8. The servlet receives the request in doPost() method.
9. Instantiate User by getting parameters through getParameters() and setting it.
10. An instance of UserDaoImpl is created, and the user variable is used with validation() method.
11. This checks the database whether the user inputted credentials are correct or not.
12. If the credentials match with in db, it gets redirect to UserHome.jsp

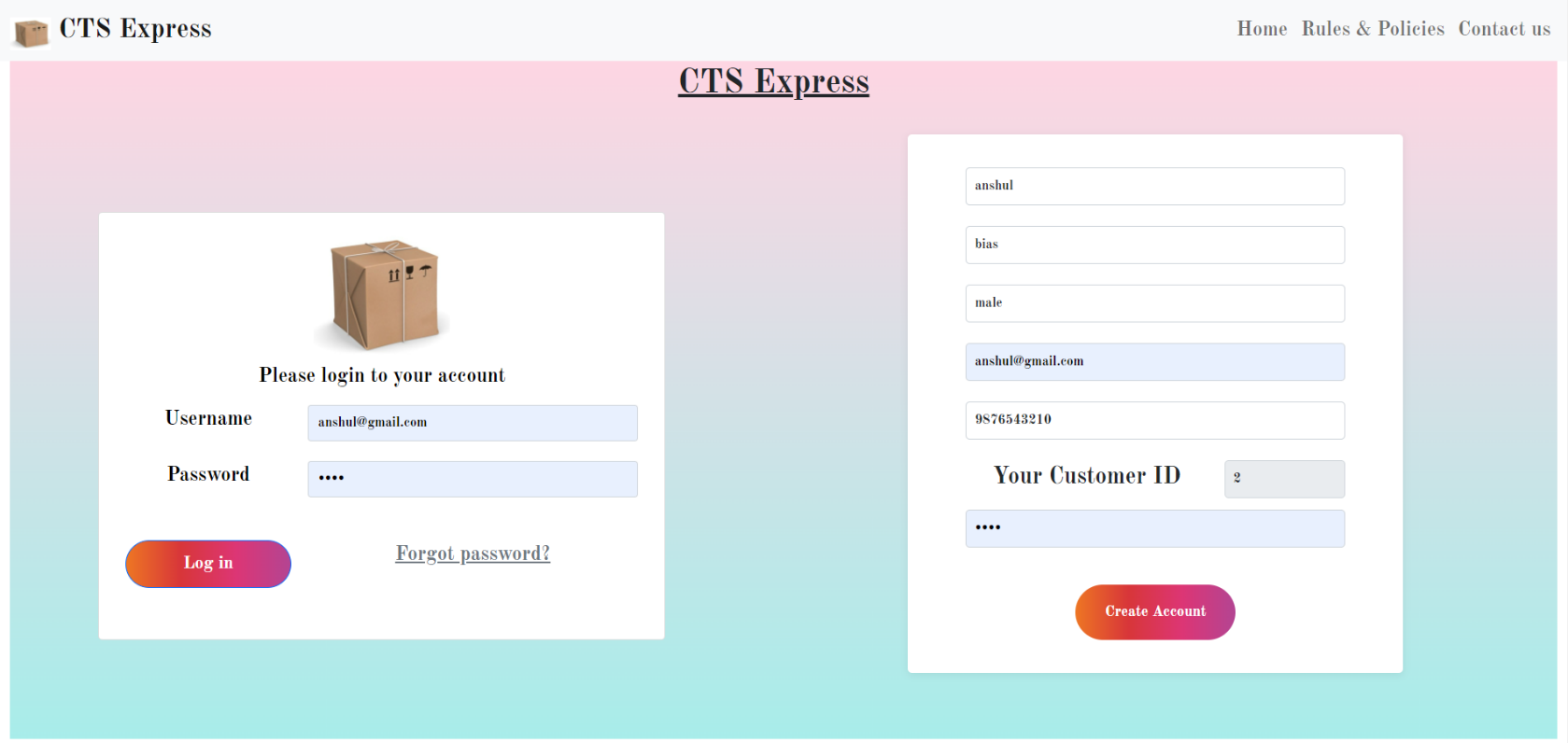
## Servlet

| **Servlet Class Name** | **URL** | **Description** |
| --- | --- | --- |
| UserBO | UserRegistration.jsp | This URL is the registration and log-in page for User(s).  This servlet sends the inputted form data of users from the jsp and invokes UserDaoImpl method for storing in database. |
| UserLoginBO | UserRegistration.jsp | This URL is the registration and log-in page for User(s).  The servlet sends the form details of User from the .jsp and checks in databse for validation of credentials.  If the credentials are matching, it redirects to UserHome.jsp |

## JSP

List the menu items for admin.

|  |  |  |
| --- | --- | --- |
| **HTML** | **Renamed JSP** | **Description** |
| src/main/webapp/ UserRegistration.html | src/main/webapp/ UserRegistration.jsp | Sends form data for insertion in db for user creation and validation for user login. |

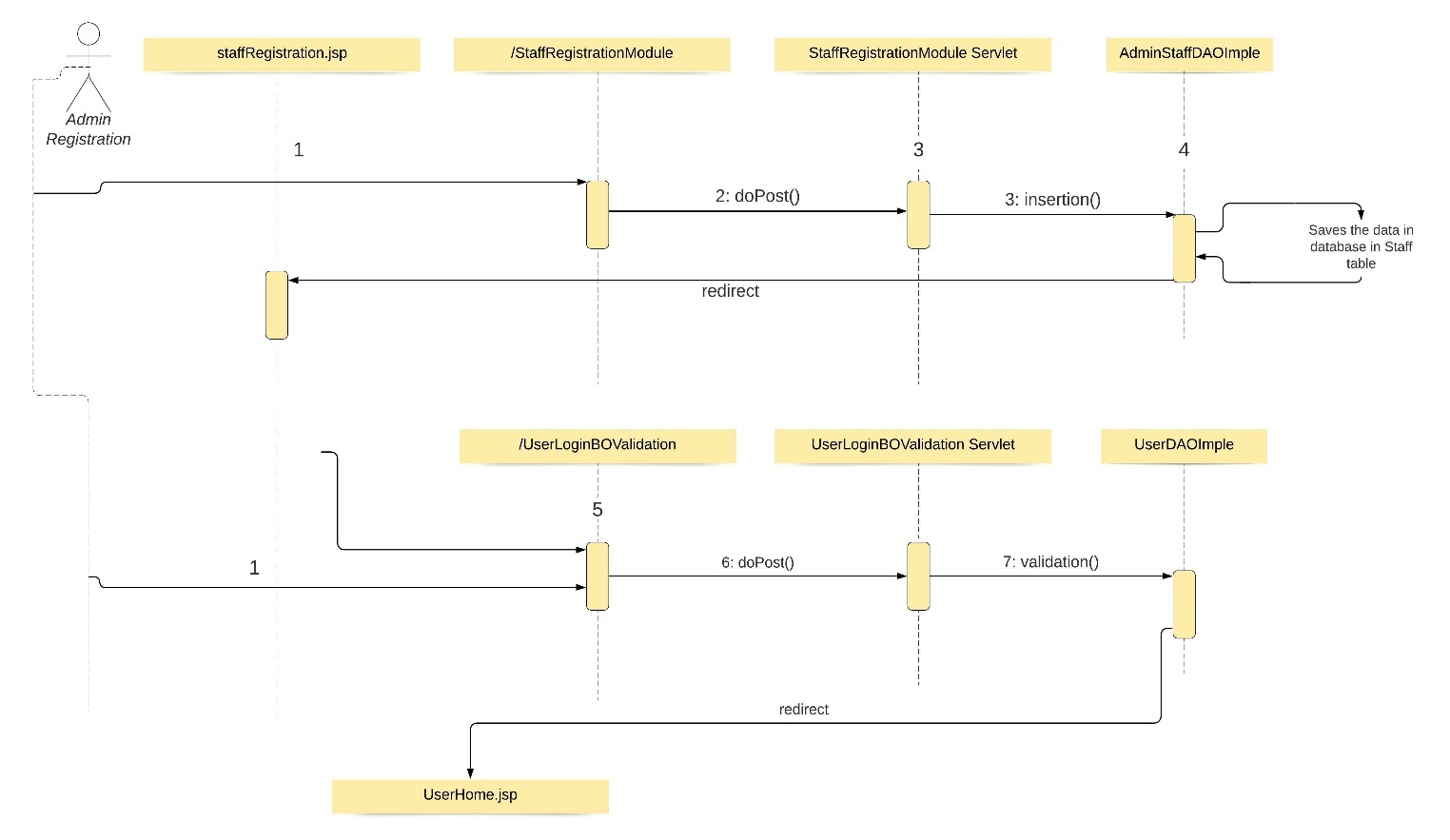


/ UserCreation

/ UserLoginBOValidation

# Design for New Staff or New Admin Registration

## Sequence Diagram



**Steps Explanation**

1. New Admin or new staff need to register himself clicks on the Admin Registration or Staff Registration link in Home page and gets redirect to staffRegistration.jsp.
2. After filling the form and submitting, the / StaffRegistrationModule is executed.
3. StaffRegistrationModule servlet receives the request in doPost() method
4. Instantiate AdminStaffDAOImple and assign it to AdminStaffDAOImple reference variable menuItemDao.
5. The AdminStaffDAOImple executes the insertion() to save data in db.
6. If data is successfully inserted in db, it redirects to staffRegistration.jsp with message “Successfully Inserted”
7. If data is not successfully inserted in db, it redirects to staffRegistration.jsp with message “Not Successfully Inserted”

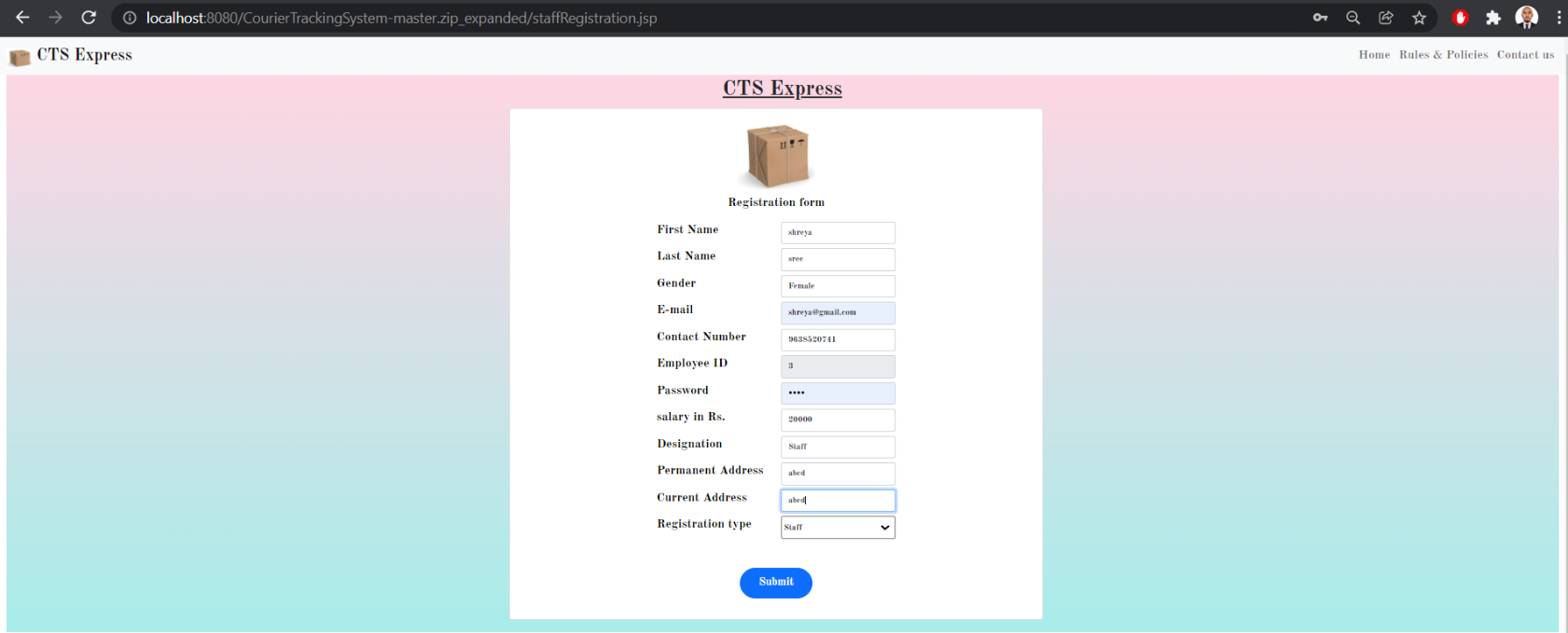
## Servlet

| **Servlet Class Name** | **URL** | **Description** |
| --- | --- | --- |
| StaffRegistrationModule Servlet | staffRegistration.jsp | This URL is the registration page for new admin and new staff.  This servlet sends the form data to save in db and returns to same JSP with message. |

## JSP

List the menu items for customer.

|  |  |  |
| --- | --- | --- |
| **HTML** | **Renamed JSP** | **Description** |
| src/main/webapp/ StaffRegistration.html | src/main/webapp/ StaffRegistration.jsp | Registration page for Staff and Admin |



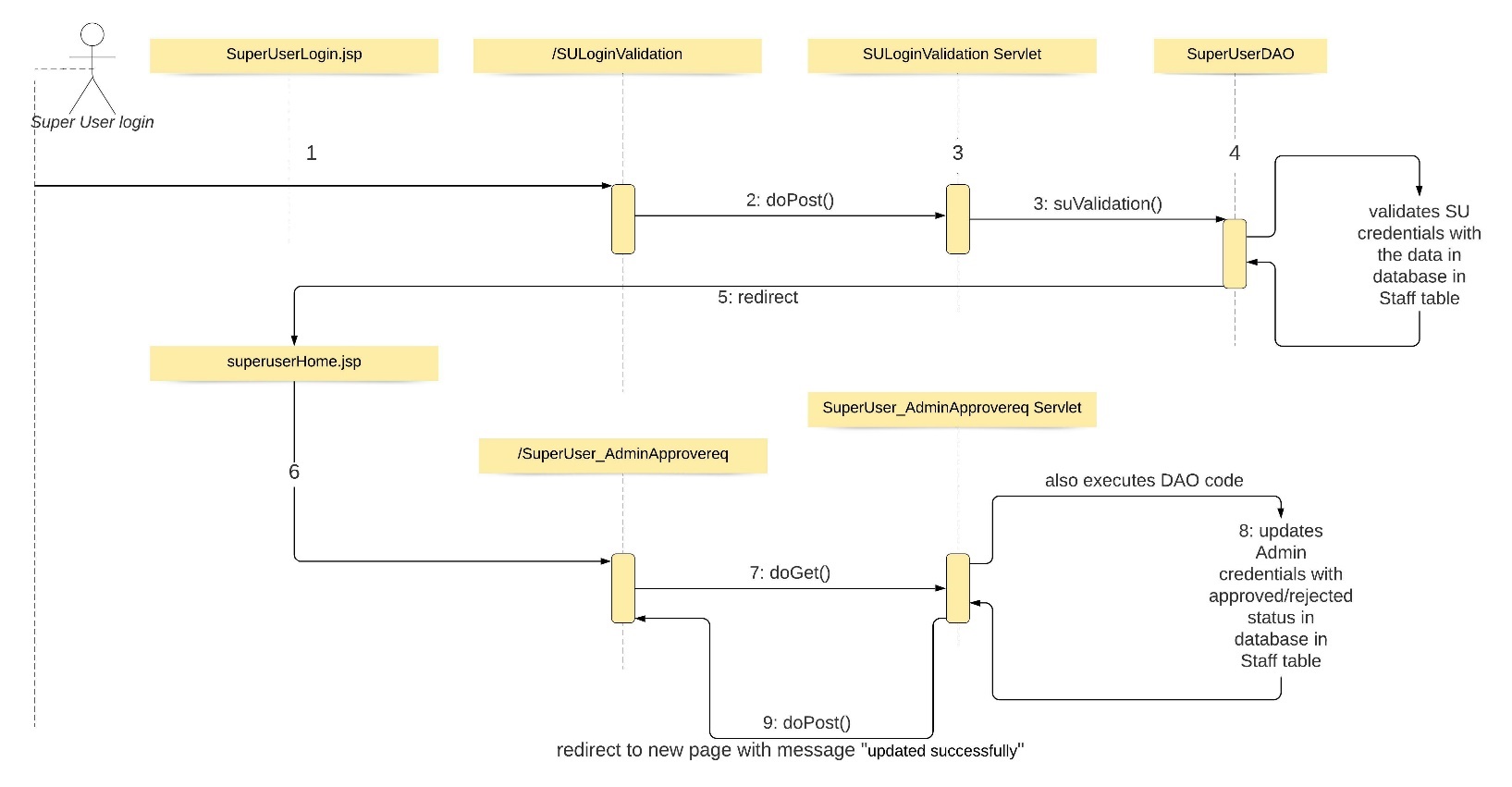
/StaffRegistrationModule

# Design for Super User Login and Admin approval

**Steps Explanation**

1. Super User enters his/her login credentials which is pre-loaded in bd.
2. On submitting the form in SU login page, the /SULoginValidation is executed.
3. The servlet receives the request in doPost() method
4. Instantiate SuperUserDAO and validate the login credentials using suValidation() method.
5. If the credentials matches with the data in db, then the servlet redirects to superuserHome.jsp. else, it redirects to index.html home page.
6. After successful logging in, Super User can approve/reject pending admin approval request by clicking on “Approve/Reject Admin Registration Requests”.
7. On clicking it, the servlet /SuperUser\_AdminApprovereq is executed.
8. This servlet displays pending admin approval request details by getting data from db by displaying it using doGet() method.
9. Upon successfully updating the permissions for each admin requests, the clicking update permission button the doPost() method runs, the servlets displays updated successfully message.

## Sequence Diagram

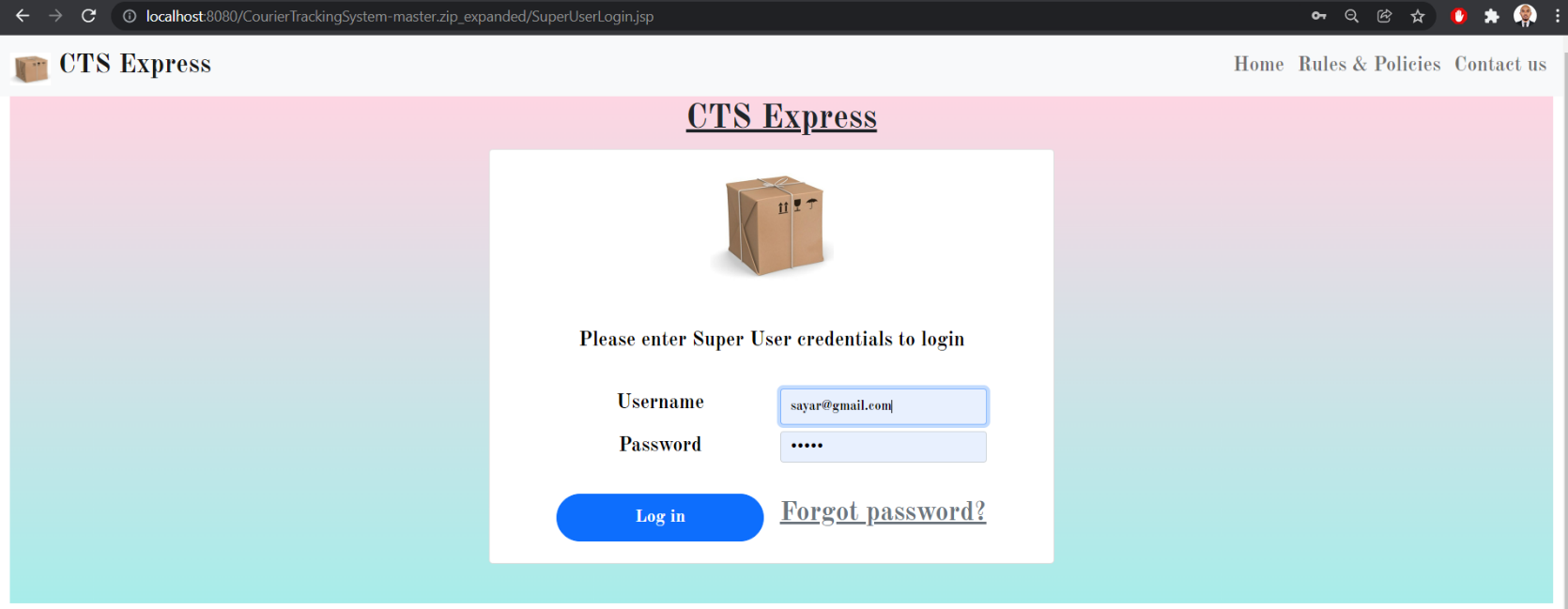


## Servlet

| **Servlet Class Name** | **URL** | **Description** |
| --- | --- | --- |
| SuperUserLoginBO | SuperUserLogin.jsp | This servlet is invoked when superuser enters his credentials in form for logging in.  It sends the form data to validate the login credentials and redirects to superuserHome.jsp if matches, else redirects to Index.jsp |
| SuperUser\_AdminApproveBO\_DAO | superuserHome | This servlet is invoked when Super User clicks “Approve/Reject Admin Registration Requests” and collects all pending approval of admin.  SuperUser can update the approval to either approve or reject for each admin request. |

## JSP (SuperUserLogin.jsp)

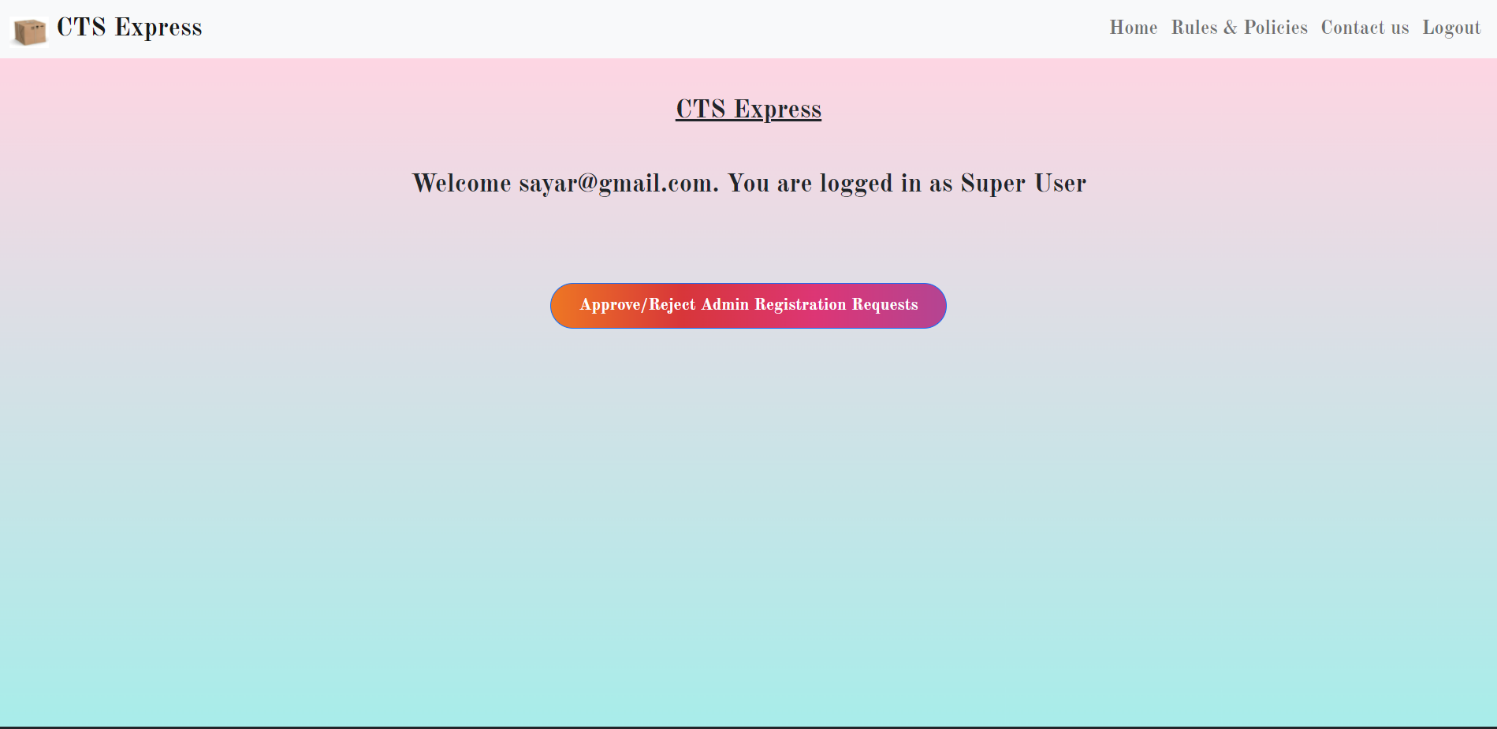
|  |  |  |
| --- | --- | --- |
| **HTML** | **Renamed JSP** | **Description** |
| src/main/webapp/ SuperUserLogin.html | src/main/webapp/ SuperUserLogin.jsp | Login page for super user |



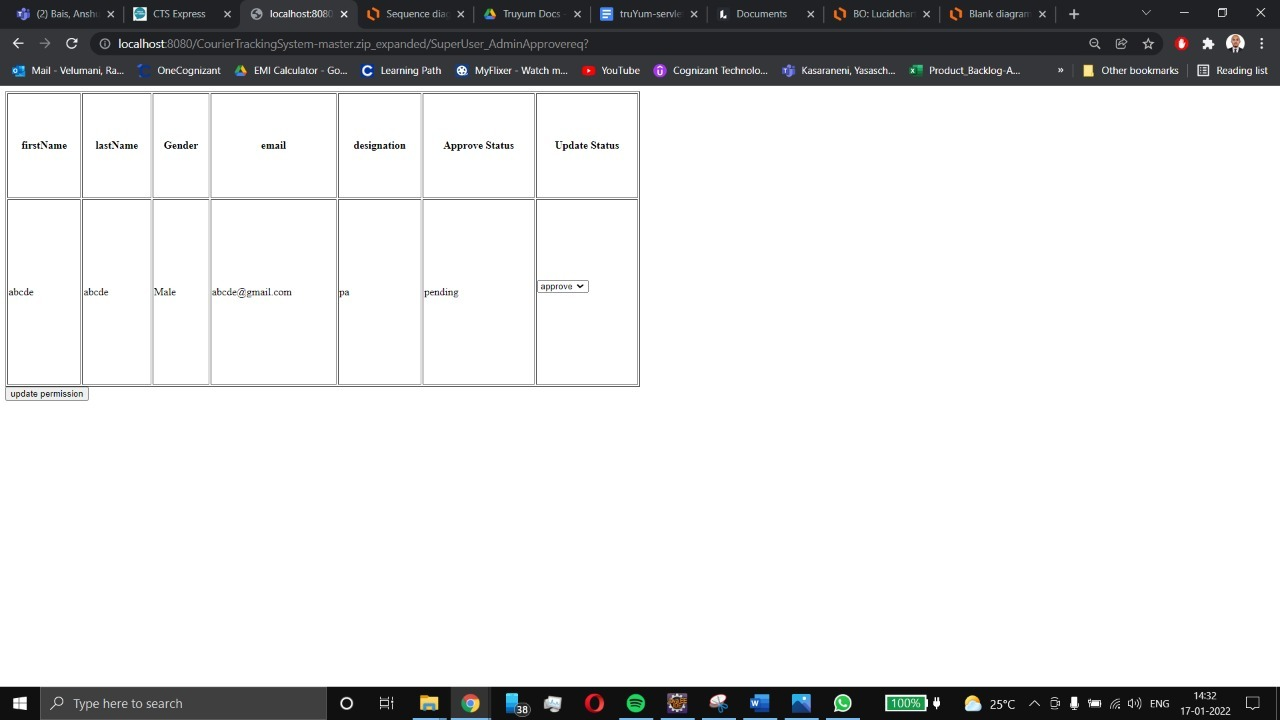
/SULoginValidation

## JSP (superuserHome.jsp)

|  |  |  |
| --- | --- | --- |
| **HTML** | **Renamed JSP** | **Description** |
| src/main/webapp/ superuserHome.html | src/main/webapp/ superuserHome.jsp | Home page for super user and admin request approval page |



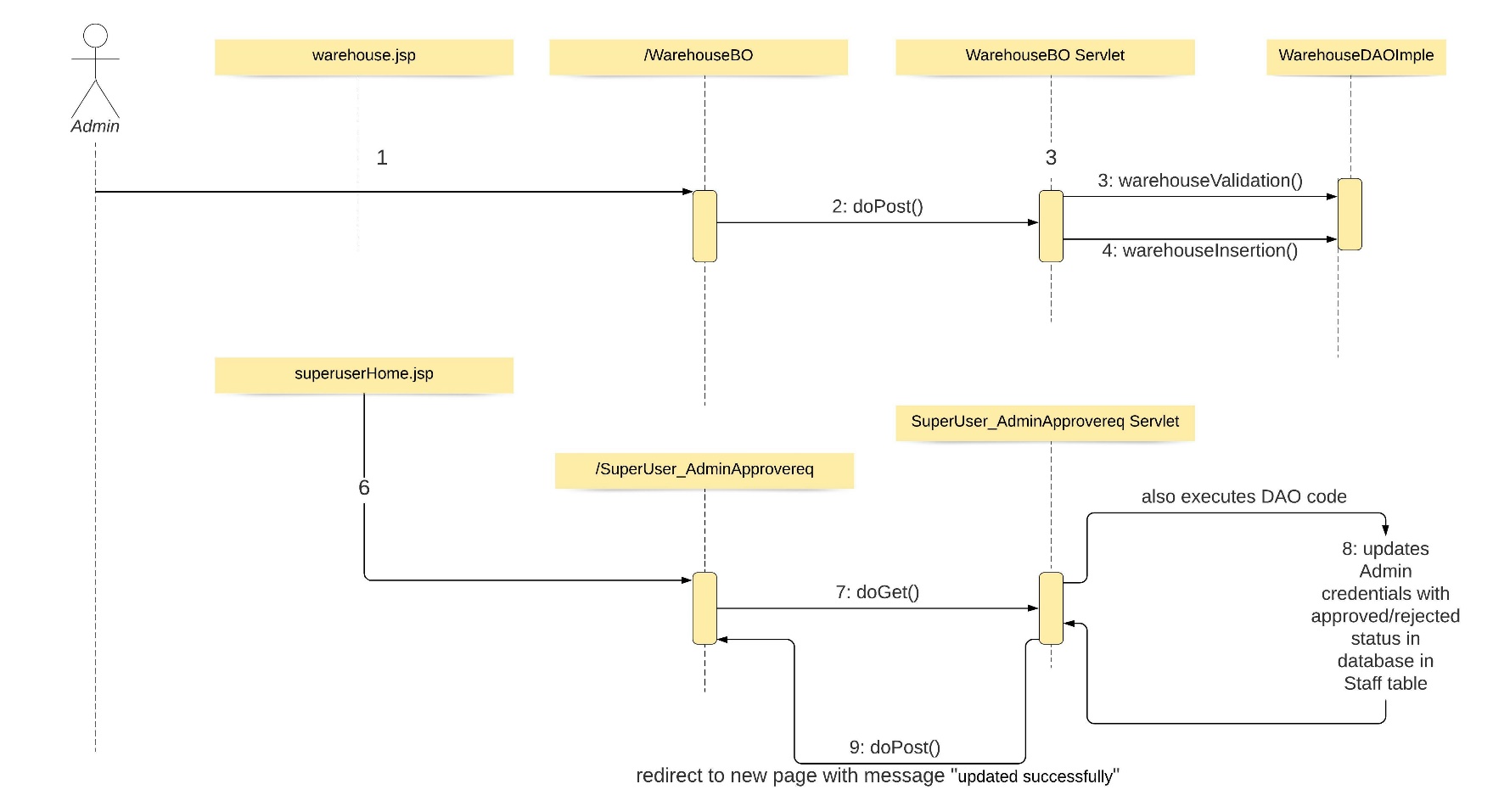
/SuperUser\_AdminApprovereq : doGet()



/SuperUser\_AdminApprovereq : doPost()

# Design for Warehouse Update

## Sequence Diagram



**Steps Explanation**

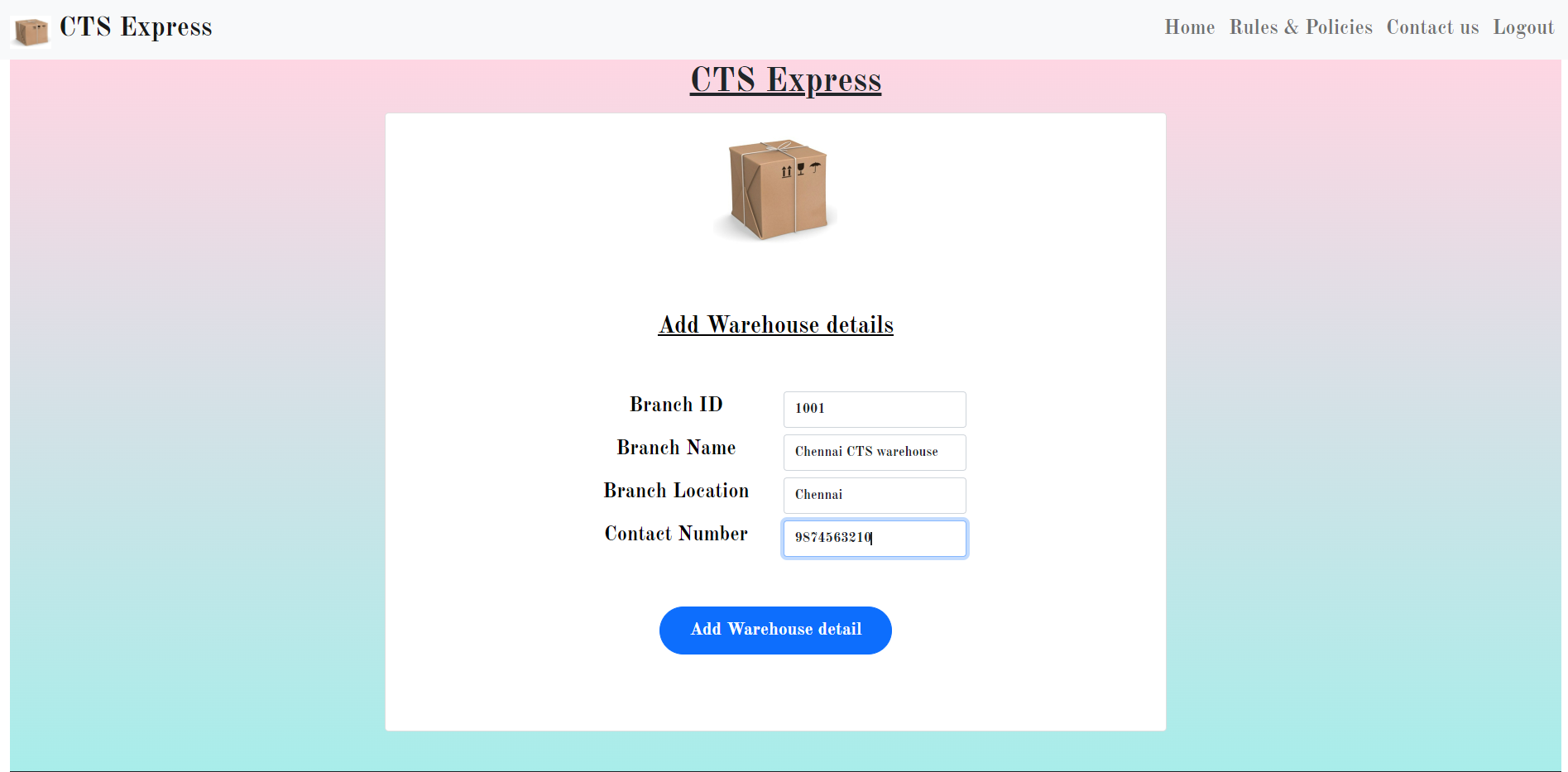
1. Admin after successfully loggedin clicks on warehouse update and gets redirected to “warehouse.jsp”.
2. The warehouse details are to be filled in the form, and when submitted the /WarehouseBO servlet is executed.
3. The servlet sends the form data through doPost() method
4. Instantiate WarehouseDAOImple and the values are passed as warehouse object in warehouseValidation() method.
5. If the credentials matches with the data in db, then the servlet displays the message “Details already added”.
6. If the credentials do not match with values in db, the warehouseInsertion() methods inserts the form data in db and prints message “Warehouse Details Successfully Inserted”.

## Servlet

| **Servlet Class Name** | **URL** | **Description** |
| --- | --- | --- |
| WarehouseBO | warehouse.jsp | This servlets adds the warehouse details from form to database if the inputted values are not already present in databse. |

## JSP (warehouse.jsp)

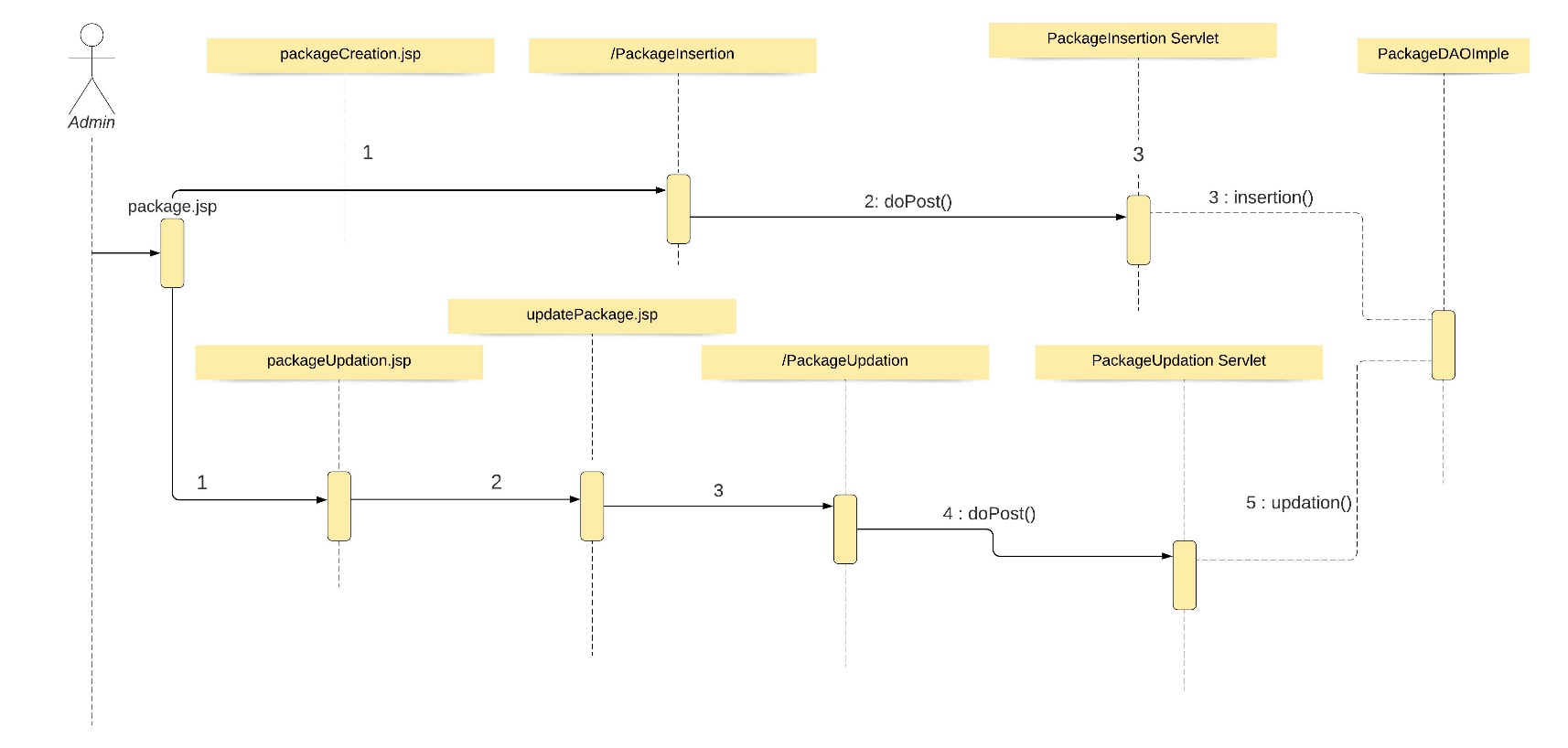
This JSP is the same the file defined in View Menu Item List Customer section.



/WarehouseBO : doPost()

# Design for Package Creation & Package Updation

## Sequence Diagram



**Steps Explanation**

1. Admins after successful login clicks on package link to create or update package in package.jsp.
2. On clicking Package Creation link and gets redirect to packageCreation.jsp, where he fill the package details in a form.
3. Submitting the form in packageCreation.jsp, /PackageInsertion servlet is executed.
4. The form data is sent using doPost() method.
5. An instance of PackageDAOImple is created to save the data in data base using insertion() method.
6. After successful inserting the data in database, the page will display message “Successfully Insereted”.
7. On clicking Package Updation gets redirect to packageUpdation.jsp, where the details of package are displayed.
8. One can update the current location, date and type of package on clicking Update link.
9. On clicking Update link in each row get redirects to updatepackage.jsp? with the details in packageUpdation.jsp.
10. One can make required updates in location, date, type of parcel in the dispayed form and submit for update.
11. On submitting the form in updatepackage.jsp?, /PackageUpdationservlet is executed and the page gets redirect to package.jsp.
12. The form data is sent using doPost() method.
13. An instance of PackageDAOImple is created to modify the updated data in data base using updation() method.

## Servlet

| **Servlet Class Name** | **URL** | **Description** |
| --- | --- | --- |
| PackageBO | packageCreation.jsp | This servlets creates new package and saves in database |
| PackageUPO | updatepackage.jsp? | This servlet updates existing package details and saves in database |

## JSP (packageCreation.jsp)

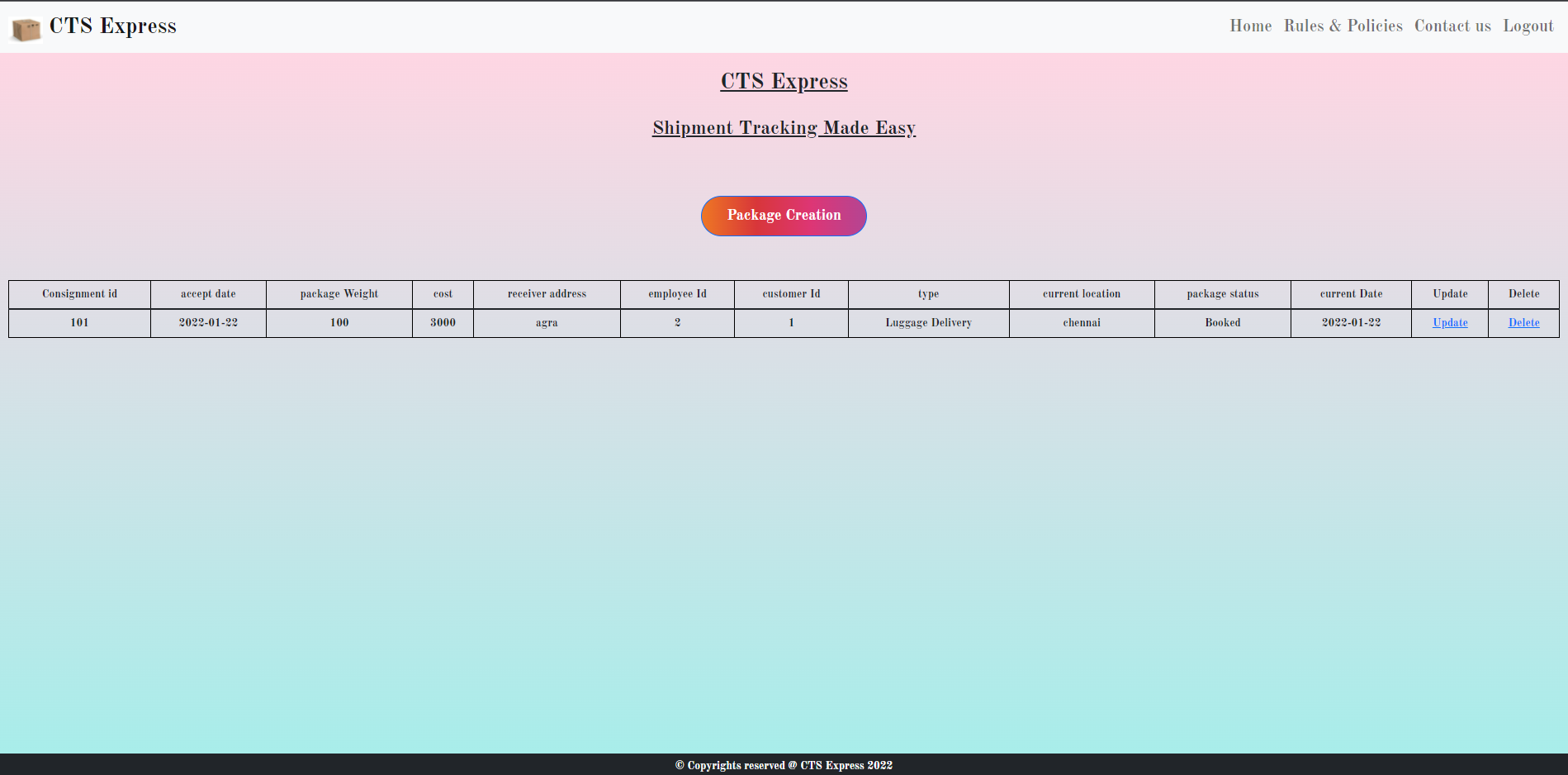
|  |  |  |
| --- | --- | --- |
| **HTML** | **Renamed JSP** | **Description** |
| src/main/webapp/ packageCreation.html | src/main/webapp/ packageCreation.jsp | Displays the page for admin to create new package |



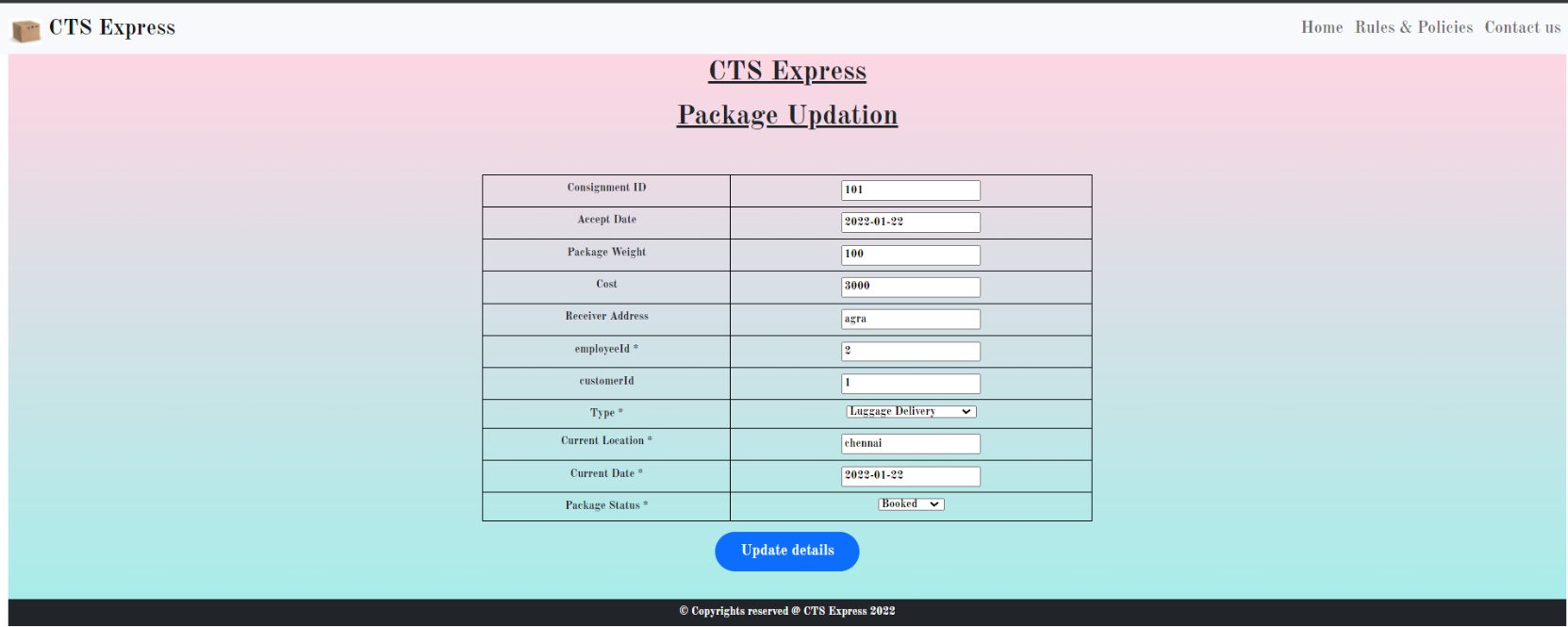
/PackageInsertion : doPost()

## JSP (updatepackage.jsp)

|  |  |  |
| --- | --- | --- |
| **HTML** | **Renamed JSP** | **Description** |
| src/main/webapp/ packageUpdation.html | src/main/webapp/ packageUpdation.jsp | Displays the page to update or delete each package |
| src/main/webapp/ updatepackage.html | src/main/webapp/ updatepackage.jsp | Displays the form for updating the package details |



Redirect to updatepackage.jsp



/PackageUpdation : doPost()

# Standards and Guidelines

## JSP

1. Dynamic content should be always generated using JSTL
2. Usage of Scriptlets should be completely avoided
3. Indentation should be inline with the standards defined for HTML
4. Use formatting JSTL tags for displaying currency and dates

## Servlets

1. All coding standards applicable for Java are applicable here
2. Servlet should not contain any business logic

# Submission

## Code submission

Once your code is evaluated by the trainer and all the issues reported by the trainer are corrected, the code needs to be submitted.

# Change Log

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Changes Made | | | |
| V1.0.0 | Initial baseline created on <18-01-2022> by <Raaghul Velumani> | | | |
| 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** |
|  |  |  |  |