Courier Management System

19-12-2021

# Overview

## Project Background and Description

Courier companies have existed for a long time now, starting with posts (letters) to all kinds of goods. Courier has become an essential part of peoples’ lives, as it helps them to send things to whomever they want, hence, connecting loved ones, helping in growing businesses, invitations, etc.

Handling too many couriers is a tedious task, and it becomes even more difficult when the couriers are not just in a city or a district or a state, but across a country and even across the globe. And for the sender, tracking the courier becomes almost impossible.

Courier Management System is a web-based application meant to digitalize the courier management and delivery for a courier company. This system will help any courier company to manage their couriers digitally, and will also help the customers to track the couriers.

The system aims to manage all courier operations, from courier pickup to delivery. The system’s major functionalities are:

* Courier creation in company database
* Courier payment management
* Courier tracking (For customer)
* Creating delivery jobs for delivery persons (by manager)
* Updating courier status timely by processor and delivery persons

## Project Scope

* The project aims to provide a courier management system.
* The project will run as a web application that can be accessed using any modern supported web browser (Google Chrome, Safari, Mozilla Firefox, etc.).
* The application must be responsive, to enable use on mobile devices.
* The application should be modular, to facilitate easy development of future features and/or extensions.

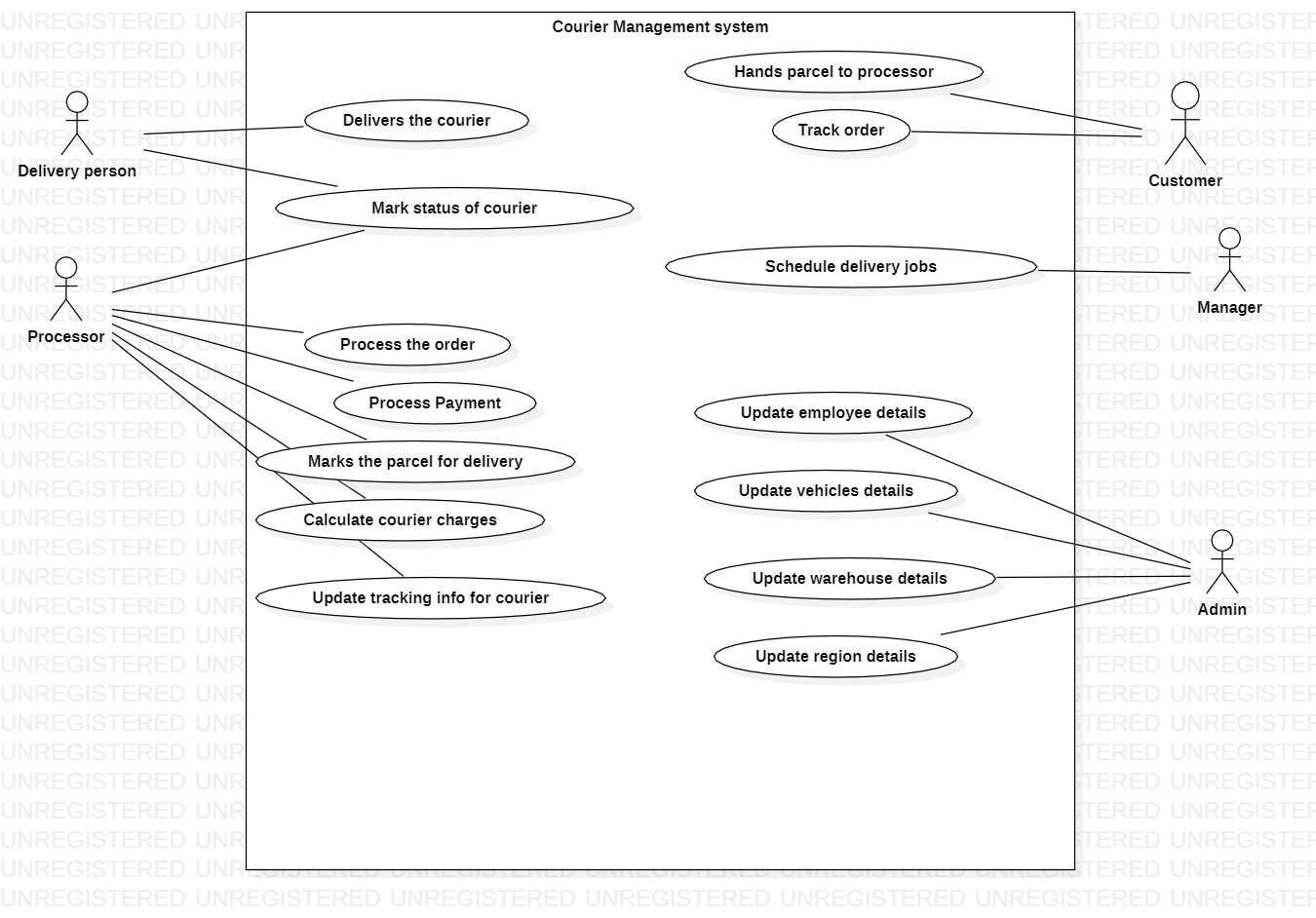
# Use case diagram

### Primary Actors:

* Processor
* Delivery Person

### Secondary Actors:

* Customer
* Manager
* Admin



# Use cases (In fully dressed format)

## USE CASE 1: Process Parcel(courier)

**Primary actor:** Processor

**Stakeholders and Interests:**

Customer**:** Wants to courier his parcel, receipt of courier, and tracking info.

Processor: Wants to create a new courier, take payment, update courier details, and return courier receipt and tracking info to customer

**Pre-conditions:** Processor is identified and authenticated

**Post-conditions:**

* Courier is created
* Payment is updated in system
* Receipt is generated
* Tracking is created. Tracking info shared with customer

**Basic Flow:**

1. Customer arrives at courier office, with parcel
2. Customer requests a courier, for parcel, to the processor
3. Processor logs into the system to create new courier
4. Processor creates a new courier in the system
5. Processor inputs courier details, like weight, dimensions, type, destination, in the system
6. System displays total cost and expected delivery date, and generate payment invoice
7. Processor provides payment invoice to customer
8. Customer pays processor
9. Processor updates payment details in the system
10. System prints courier receipt and tracking details
11. Processor hands over the courier receipt and tracking details to customer

**Extensions:**

3a.Processor is not identified or authentication fails upon login

1. System signals error, and records it
2. Processor tries again

5a. Processor inputs a destination where service is not available

1. System shows an info, saying the area code is not serviceable
2. Processor cancels the courier
3. Customer takes the parcel and leaves

6a. System detects an invalid input from processor

1. System displays error to processor, describing about the error
2. Processor tries again with correct input
3. System displays total cost and expected delivery date, and generate payment invoice

6b. System ran into an error while calculating the cost/ expected delivery date/ generating payment invoice

1. System record and displays the error
2. Processor tries to create a new courier

8a. Customer pays in Cash

1. Processor keeps the money in cash register, and/or returns the change to the customer
2. Processor updates cash payment in the system

8b. Customer pays via online payment methods (Paypal/UPI(India only)/others)

1. Processor validates that payment is successful
2. Processor updates online payment in the system

8c. Customer pays via online payment methods (Paypal/UPI (India only)/others)

1. Processor validates that payment is failed
2. Processor asks customer to make the payment again
3. Once payment is successful, Processor updates online payment in the system

8d. Customer pays with card (Credit/Debit)

1. Processor validates that payment is approved
2. Processor updates card payment in the system

8e. Customer pays with card (Credit/Debit)

1. Processor validates that payment is failed
2. Processor asks customer to make the payment again
3. Processor updates card payment in the system

## USE CASE 2: Courier Delivery

**Primary actor:** Delivery person

**Stakeholders and Interests:**

Customer**:** Wants to courier his parcel, receipt of courier, and tracking info.

Processor: Wants to create a new courier, take payment, update courier details, and return courier receipt and tracking info to customer

Manager: Wants to assign delivery jobs to delivery persons

Delivery Person: Wants to deliver the order

**Pre-conditions:** Delivery person is identified and authenticated

**Post-conditions:**

* Courier is delivered
* Courier status is updated in system
* Tracking is updated

**Basic Flow:**

1. Manager logs into the system
2. Manager creates a new delivery job
3. System fetches the pending parcels
4. System fetches the available delivery persons
5. System shows the pending parcels and available delivery persons to the manager
6. Manager assigns parcel(s) to a delivery person
7. System updates status of the parcels
8. System adds parcel to delivery job
9. System assigns job to delivery person
10. Delivery person takes the order for delivery
11. Delivery person logs into the system
12. Delivery person delivers the parcel and updates the courier status in system

**Extensions:**

1a. Manager login fails

1. Manager tries again

3a. System run into an error while fetching pending parcel

1. System records and logs the error. Displays error to manager
2. Manager tries after some time

4a. System run into an error while fetching available delivery persons

1. System records and logs the error. Displays error to manager
2. Manager tries after some time

10a. Delivery persons takes only a few of the parcels for delivery

1. Delivery person logs into the system
2. Delivery person delivers the parcel and updates the courier status in system
3. Delivery person comes back and takes other parcels for delivery. Repeats steps 1 and 2

10b. Delivery person gets sick and takes abrupt leave

1. Delivery person cancels the delivery job assigned
2. System updates parcel status (to pending for delivery)

10c. Delivery person could not carry a parcel with him

1. Delivery person logs into the system
2. Delivery person updates status of the parcel (to pending for delivery)

11a. Delivery person login fails

1. Delivery person tries again

12a. Delivery person is unable to reach the delivery location

1. Delivery person checks contact details for parcel delivery, contacts the receiver and takes help in delivering the parcel
2. Delivery person updates courier status

12b. Receiver is unavailable to take the parcel

1. Delivery person checks contact details for parcel delivery, contacts the receiver and takes help in delivering the parcel
2. Delivery person updates courier status

1a. Receiver can take the parcel on the next day

1. Delivery person updates status of the parcel (to pending for delivery)

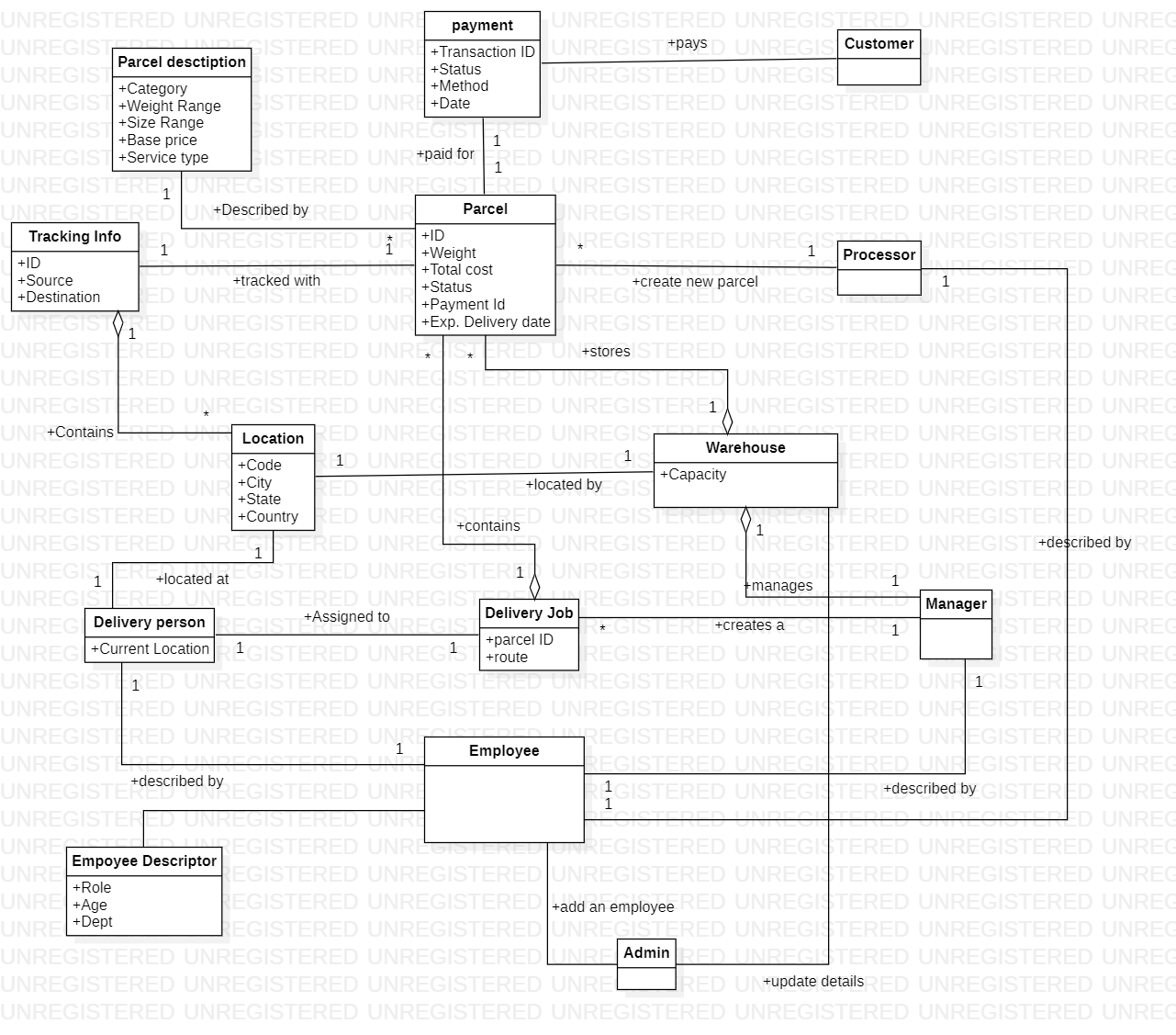
1b. Receiver cannot receive the parcel

1. Delivery person marks the courier for returning to the shipper with comments

1c. Receiver denies to accept the parcel

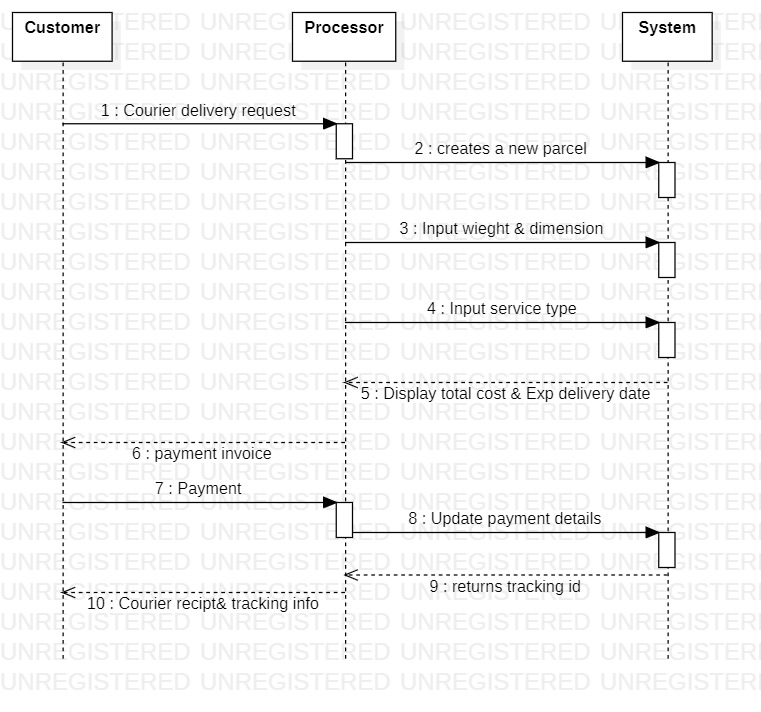
1. Delivery person marks the courier for returning to the shipper with comments

# Domain Model

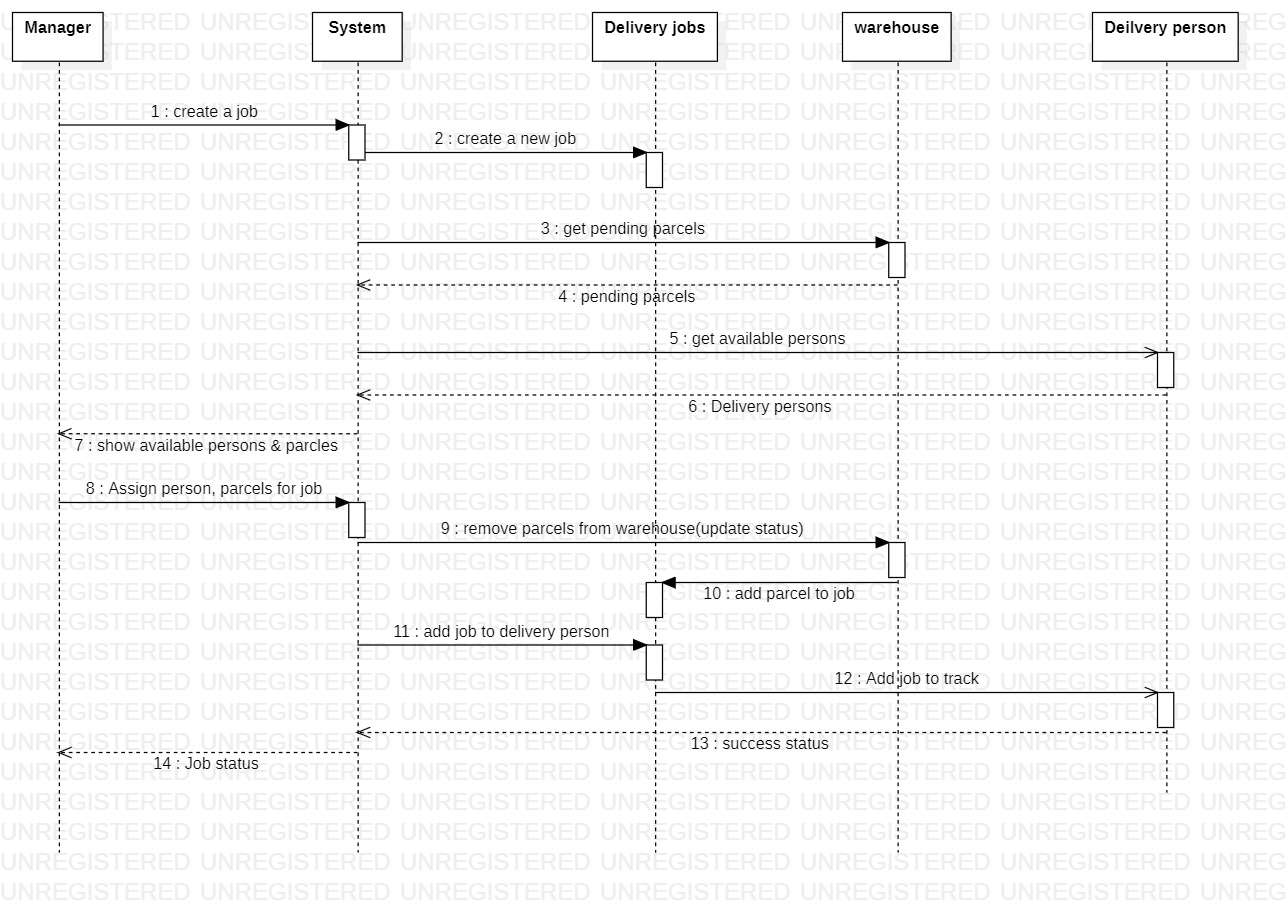


# Sequence Diagram

## USE CASE 1: Courier Request



## USE CASE 2: Courier Delivery



# Application Design patterns

### GOF Design Patterns:

From the Gang of Four Design Patterns, the patterns that can be applied to improve the Object Oriented Design of the system are:

* Chain of Responsibility
* Adapter

### GRASP Design Pattern

From the GRASP Design Patterns, the patterns that can be applied to improve the Object Oriented Design of the system are:

* The Controller
* The Creator