By:

Vijaykumar R Pai – PES1201702013

Ayush Pratyay – PES1201702164

Subham Singh – PES1201801830

Optimizing courier delivery system

**OPTIMIZING COURIER DELIVERY SYSTEM**

1. **Introduction**

Purpose - A Software as a service (SaaS) web application where in the system automates the delivery services that can make decisions as to which delivery point should be reached first for feasible ways by taking list of address from all the nearest delivery points.

Document conventions -

DB - Database

DDB - Distributed Database

ER - Entity Relationship

Intended Audience and Reading Suggestions - This project is a prototype for optimizing courier delivery system. This project is useful for the courier companies and as well as to the delivery executives.

Project Scope - The purpose of this courier delivery system is to ease the courier delivery management and to create a convenient and ease to use application for delivery executives trying to deliver package. The system is based on a non-relational database with its courier delivery management and tracking functionalities.

References -

* GitHub
* Stack overflow
* JavaScript tutorials

**2) Overall Description**

Product Perspective - Optimizing courier delivery system stores the following information

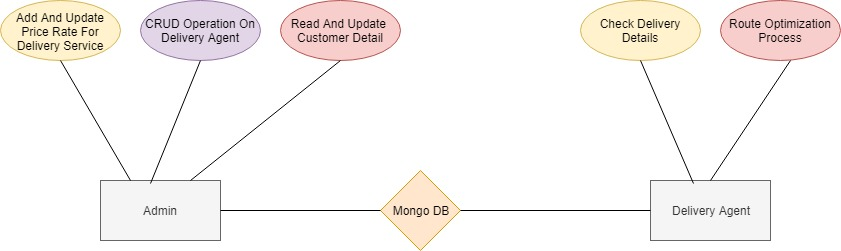
**Package details**: It gives a detailed description of the package along with the delivery person’s name and the package number.

**Tracking the package**: A feature wherein it allows to live track the exact location of the package.

**Optimizing the delivery route**: Allows the delivery executive to deliver the package without any hassle by optimizing the delivery route according to the traffic.

Product features –

The major features of the courier delivery system is shown in the below E-R Diagram



User classes and characteristics - Users of the system should be able to track their package by simply entering the unique package id in the search box, which was given by the respective courier company.

The customer will be able to track his/her package by entering the package number in the search box.

The Admin should have the following management functionalities:

* Should be able to add new package and its details
* Must be able to delete the existing package
* Will have access to all package details
* Will have list and details of all delivery executives
* Will be able to track agent
* Will be able to change application settings

Operating Environment

Operating environment for the delivery system is as listed below:

* Distributed database
* Client / Server system
* Operating system : Windows/Linux/MacOS
* Database : NoSQL
* Platform : Node.js

Design and Implementation Constraints

* The global schema, fragmentation schema and allocation schema.
* NoSQL commands for above queries/applications
* Implement the database at least using a centralized management system.

User Documentation:

As the product is CMS, user documentation becomes a critical component of the system as it shall provide specific guidelines to a user for using the system.

Assumptions and Dependencies

Let us assume that this is a deployed courier delivery system and it is used in the following application:

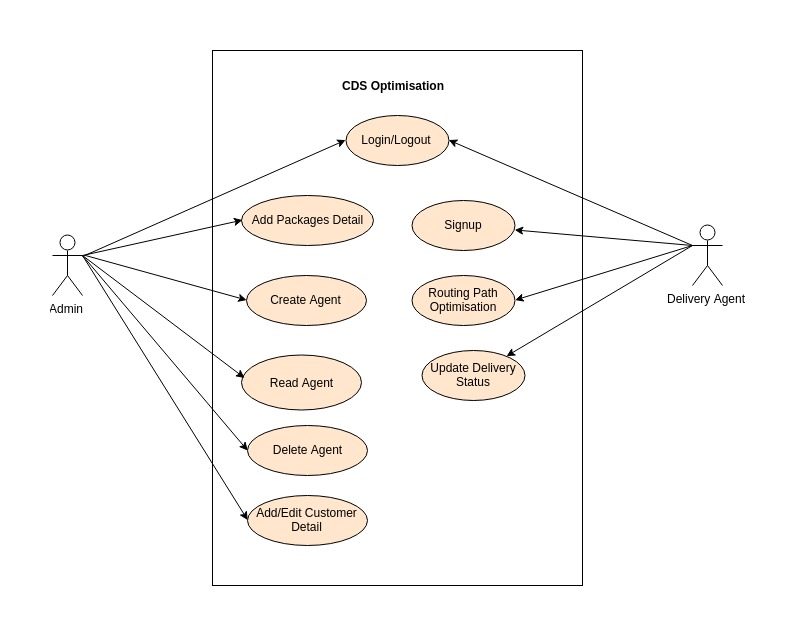
* Get real time traffic updates that will allow the courier delivery executive to deliver the package within a shortest span of time.
* Reroute the delivery route of the package to be delivered according to the traffic on route.

**3) System Features**

The courier delivery system maintains information about number of delivery executives and their details, packages to be delivered and their details and tracking the movement of package.

**Functional requirements**

Other system features include:



Client / Server System:

The term client/server refers primarily to an architecture or logical division of responsibilities, the client is the application - frontend (Here the client is the delivery executives) and the server is the DBMS (backend).

**4) External Interface Requirements**

User interfaces:

Front-end software: HTML, CSS and React js

Back-end software: MongoDB (NoSQL)

Hardware interfaces:

* Windows / Ubuntu 18.06
* Web or mobile browser which supports CGI, HTML and JavaScript

Software interfaces:

|  |  |
| --- | --- |
| **Software used** | **Description** |
| Operating System | We have chosen Windows Operating System for its best support and user-friendliness |
| Database | To save the delivery details and delivery agent details, we have chosen NOSQL(MongoDB) |
| JavaScript | To implement the project, we have chosen JavaScript language for its more interactive support |
| Heroku Cloud Deployment | The platform for deploying our app, we choose Heroku NodeJS Platform. |
| MLab MongoDB cloud Deployment | MLab MongoDB cloud service to store data in NoSQL database. |

Communication interfaces:

The project supports on Chrome, Firefox, Safari, and Opera.

**5) Other Nonfunctional Requirements**

Performance Requirements:

The steps involved to perform the implementation of courier delivery system are listed below

**E-R Diagram:**

The E-R Diagram constitutes a technique for representing the logical structure of a database in a pictorial manner. This analysis is then used to organize data as a relation, normalizing relation and finally obtaining a relation database.

ENTITIES: Which specify distinct real-world items in an application.

PROPERTIES / ATTRIBUTES: Which specify properties of an entity and relationships.

RELATIONSHIPS: Which connect entities and represent meaningful dependencies between them.

Safety Requirements:

If there is extensive damage to a wide portion of the database due to catastrophic failure, such as a disk crash, the recovery method restores a past copy of the database that was backed up to archival storage (typically tape) and reconstructs a more current state by reapplying or redoing the operations of committed transactions from the backed up log, up to the time of failure.

Security Requirements:

Security systems need database storage just like many other applications. However, the special requirements of the security market mean that vendors must choose their database partner carefully.

Software Quality Attributes:

* AVAILABILITY: The courier should be delivered within the stipulated date and time.
* CORRECTNESS: The delivery executive should start from the correct place at the right time and reach the designated destination.
* MAINTAINABILITY: The admin must be able to maintain the correct schedules of the packages to be delivered.
* USABILITY: The delivery schedules should satisfy a maximum number of customers as the courier must be delivered according to the convenience of the person receiving it.