CHAPTER ONE

INTRODUCTION

1. Background Information

Nowadays, many people shop online. Therefore, delivery of orders is very important. In our application, we are addressing an optimization problem by using the genetic algorithm by considering the position distance difference and priorities between multiple points. The reason for this problem is that customers do not know when the order will be delivered during the day, so waiting for hours of delivery in the home or office seems to be a problem. For this reason, users always go to cargo branches. However, the products can be damaged because they are transferred many times. This is also a bad experience for users and compensation to cargo companies. In addition, because cargo companies have a lot of transportation for delivery, fuel costs are very high and this is a very important problem for them. As a result, optimization is required for these problems and our application provides a solution to this problem.

1. Problem Definition

One of the biggest problems in the cargo sector is that orders cannot be delivered on time. There are many elements that prevent the delivery of the order on time. One of them is that the distribution routes are not created properly. Customers are suffering from packages that do not arrive on time. At the same time, customers do not know the delivery time of the package for hours waiting in the home and office package. The handheld terminals used by the shipping companies to register packages create extra large costs. We will integrate the functions of these handheld terminals into our application. This way, we will be able to manage courier transport information more easily and cost-effectively.

1. Motivation / Related Works

There are many routing applications on the market. However, they normally make a route between two points. In addition, commercially available applications are usually running on the web. We design our application directly as an assistant in couriers. We describe the routes and bring them to the relevant points and we can complete the delivery operations through our application. In addition, there is no information about delivery time in other commercially available applications. We will provide users with an estimated delivery time. In this way, we are motivated to do something that has not been done before.

1. Goal/Contribution

With our application, we will reduce the time loss of companies and customers. We will reduce transportation costs for companies. With our app we will be able to provide users with the location of the package and the estimated delivery time. Customers will not spend all day in their homes or offices waiting for their packages to arrive. They will be able to spend more time on their other business by moving on to the estimated delivery time. The application will ensure timely and smooth delivery of many cargoes. It is aimed to increase the daily cargo capacity of the user firms. Also, the daily fuel consumption of the cargo companies will be reduced and the company's profit will be increased. In addition, user satisfaction will be increased by shortening the delivery times of the company and providing consistent estimated delivery times. Thanks to this, the company's market share will be enlarged.

1. Project Scope

We will analyze the packages of the cargo company's customers and create the most suitable route. The carrier company's couriers will be accessible instantly. Couriers will be delivered to the package with barcode on the application will be saved in the system. After all the packages are added, various actions will be made and the appropriate route will be created. Get directions to deliver the packages. In addition, the estimated delivery time will be calculated according to the package delivery order and the customer will be informed. Customers will be able to view the location of the package from the cargo tracking screens.

1. Methodology/Tools/Libraries

In our project, we will find the shortest route by using genetic algorithm taking into account the priorities and positions for fuel saving and increasing the daily delivery capacity. We will also use machine learning algorithms in the estimated time calculation. We will use Google Maps APIs. System package information using barcode reader libraries will reach the information with web services. We aim to use React which is a JavaScript library. In this way we will be able to implement the cross-platform application.

CHAPTER TWO

Literature Review

1. Engineering Fast Route Planning Algorithms

Finding an optimal route in a transportation network between specified source and target nodes is one of the showpieces of real-world applications of algorithmic. We frequently use this functionality when planning trips with cars or public transportation. There are also many applications like logistic planning or traffic simulation that need to solve a huge number of shortest-path queries in transportation networks. The cost function may be any mix of travel time, distance, toll, energy consumption, scenic value associated with the edges. The task is to compute the costs of optimal paths between arbitrary source-target pairs.

Dijkstra’s Algorithm (Dijkstra, E.W.,1959) - The classical algorithm for route planning- maintains an array of tentative distances D[u] ≥ d(s, u) for each node. The algorithm visits (or settles) the nodes of the road network in the order of their distance to the source node and maintains the invariant that D[u] = d(s, u) for visited nodes. We call the rank of node u in this order its Dijkstra rank rks(u). When a node u is visited, its outgoing edges (u,v) are relaxed, i.e., D[v] is set to min(D[v], d(s, u) + w(u,v)). Dijkstra’s algorithm terminates when the target node is visited. The size of the search space is O(n) and n/2 (nodes) on the average. We will assess the quality of route planning algorithms by looking at their speedup compared to Dijkstra’s algorithm, i.e., how many times faster they can compute shortest-path distances.

Priority QueuesDijkstra’s algorithm can be implemented using O(n) priority queue operations. In the comparison based model this leads to O(n log n) execution time. In other models of computation (e.g. Thorup, M.,2003) and on the average (Meyer, U.,2001), better bounds exist. However, in practice the impact of priority queues on performance for large road networks is rather limited since cache faults for accessing the graph are usually the main bottleneck. In addition, our experiments indicate that the impact of priority queue implementations diminishes with advanced speedup techniques since these techniques at the same time introduce additional overheads and dramatically reduce the queue sizes.

Bidirectional Search executes Dijkstra’s algorithm simultaneously forward from the source and backwards from the target. Once some node has been visited from both directions, the shortest path can be derived from the information already gathered (Dantzig, G.B,1962). In a road network, where search spaces will take a roughly circular shape, we can expect a speedup around two —one disk with radius d(s, t) has twice the area of two disks with half the radius. Bidirectional search is important since it can be combined with most other speedup techniques and, more importantly, because it is a necessary ingredient of the most efficient advanced techniques.

Geometric Goal Directed Search (A∗) The intuition behind goal directed search is that shortest paths ‘should’ lead in the general direction of the target. A∗ search (Hart et al.,1968) achieves this by modifying the weight of edge (u,v) to w(u,v) − π(u) + π(v) where π(v) is a lower bound on d(v,t). Note that this manipulation shortens edges that lead towards the target. Since the added and subtracted vertex potentials π(v) cancel along any path, this modification of edge weights preserves shortest paths. Moreover, as long as all edge weights remain nonnegative, Dijkstra’s algorithm can still be used. The classical way to use A∗ for route planning in road maps estimates d(v,t) based on the Euclidean distance between v and t and the average speed of the fastest road anywhere in the network. Since this is a very conservative estimation, the speedup for finding quickest routes is rather small. (Goldberg and Harrelson,2005) even report a slow-down of more than a factor of two since the search space is not significantly reduced but a considerable overhead is added.

1. Improving Operations with Route Optimization

Description of Problem:Vehicle Routing Problem (VRP) can be described as the problem of creating a set of optimal routes from one, or many, depots to multiple customers, subject to a set of constraints. The objective is to deliver goods to all customers, at the same time minimising for the cost of the routes and the number of vehicles. Currently, the state-of-art solutions are obtained using the metaheuristics: (Nagata,2007) Genetic Algorithms, (Bräysy and Gendreau,2002)Tabu Search and (Tan et al.,2006)Ant Colony Optimization. These are the methods mainly used in the field nowadays.

Algorithms and Solutions: In the first version of our software, the volume of orders submitted to Route Optimizer quickly increased from 500 items per warehouse to 1000+. Theoretically, we should be fine. Our algorithm runtimes and memory usage jumped incredibly quickly from 1 minute and 500 MB to 10 minutes and 5 GB. As we tested it for higher and higher volumes, we finally reached the maximum for 2000 waypoints the module used up 25GB of RAM memory. That is why we decided to use a modified approach  called “recursive-DBSCAN”.

Recursive-DBSCAN; We decided to try (Ester et al.,1996) DBSCAN algorithm, at the same time the algorithm allows us to dig in deeper into high waypoint-density regions, while grouping remote orders together.

For a list of orders, we aim to find the radius for which the average number of waypoints will be the biggest (but the number of clusters will be higher than min\_no\_clusters). We do so by using a simple binary search algorithm.

Once we have found the optimal solution we “enter” the clusters that are too big and apply the same logic until we reach a point when each cluster contains less than max\_len\_cluster. Then, for each cluster, we run Route Optimization algorithm we have developed using Google Optimization Tools. This method will give us a similar result more quickly, and using less RAM memory.

1. Multi-objective Genetic Algorithms for Vehicle Routing Problem with Time Windows

To make for effective decision and comparisons of methodologies of route optimization problem, multi-objective optimization and pareto ranking applications of genetic algorithms were used to compare all items of route that contains customer information, many approach has been used, but genetic algorithm solution is more effective than other solution. In genetic algorithm, that must contain fitness function effectively. Genetic algorithm process replaces the fitness function by raw pareto ranks, using pareto ranking algorithm. Run the genetic algorithm successfully, the pareto front will be the set of solution obtained.In this solution, a chromosome representing a network configuration, is given by an integer string of length N that is the number of customers in particular problem instance. A gene in a given chromosome indicates the original node number assigned to a customer. A chromosome is like 2 5 1 4 7 8 6 3 9. (Hanshar et al.,2006)

1. Route Optimization To Increase Energy Efficiency and Reduce Fuel Consumption of Communal Vehicles

Route optimization for minimal fuel consumptionIn generally, the route optimization first purpose is saving fuel consumption and time. This approach is using many criterion of distance, and expand that the existing database, and determine the optimal route for fuel-economical. This model is shown on figure 1. The figure contains street network, vehicles, routes, waste bin locations. Each street are in between two junction of street segment. Each street segment was attributed with fuel-consumption factor ( fc ). The most fuel-economical route has the lowest Total Fuel Consumption (TFC) expressed by equation: TFC = Σ(Lsegi fci), where Lsegi presents the length of street segment with matched fci.(Bošković et al.,2010)

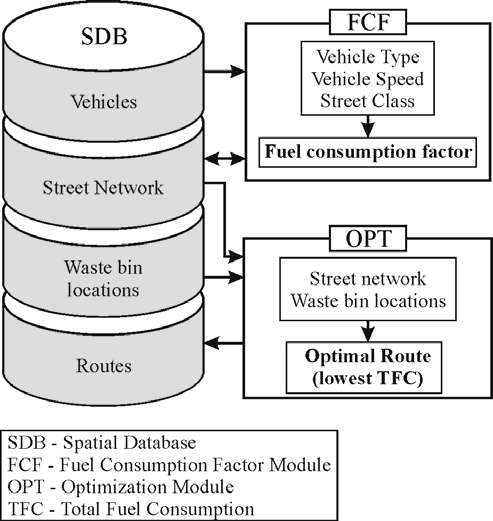
****

Figure 1.. Approach Model

1. Route optimization for solid waste collection

This approach was used in waste collection problem. But this work contains many vehicle, and routes, of course route planning like our project. A shortest path model was used in order to optimize solid waste collection/hauling processes, by aiming at minimum distance. The Route View ProTM software integrated with GIS elements such as numerical pathways, demographic distribution, container distribution and solid waste production was used as an optimization tool. (Apaydin and Gonullu,2007)

1. An Effective Genetic Algorithm for Capacitated Vehicle Routing Problem

The capacitated vehicle routing problem (CVRP), is one of the most important topic in operation cargo companies, making logistics, many little and big carriers. It is of paramount importance to thousands of companies and organizations engaged in the delivery and collection of goods or people. Metaheuristics techniques are often more suitable for CVRP to find a near optimal solution.

The CVRP is defined as an undirected graph 𝐺 = (𝑁,𝐸), where 𝑁 = {0, 1, … , 𝑛} is the set of nodes, 𝐸 = {(𝑖, 𝑗) ∶ 𝑖, 𝑗 ∈ 𝑁, 𝑖 ≠ 𝑗} is the set of edges joining the nodes. Node 0 is the depot and the other nodes represent the customers having a known demand 𝑑𝑖 for customer 𝑖. The travel distance between node 𝑖 and 𝑗 is defined by 𝑑𝑖𝑗 > 0 and each vehicle 𝑘 has a unique capacity of 𝑄k.

1. A discrete firefly algorithm to solve a rich vehicle routing problem modelling a newspaper distribution system with recycling policy

This article contains vehicle routing problem modelling using Firefly algorithm, that was first developed by Xin-She Yang in 2008 (Fister et al. 2014). Firefly algorithm is generally using for solving continuous optimization problems. Here to calculate distance between two routes, Euclidean distance was used. But Firefly algorithm cannot be applied directly to solve the problem. We must modify the algorithm to find the best solution. The modify is in order to prepare it for addressing problem. At the same time the distance calculation of our problem can use the Hamming distance between two fireflies is the number of non-corresponding elements in the sequence. (Diaz et al. 2017)

CHAPTER THREE

Requirements/Requirement Engineering

1. Functional Requirements

Login

The software has two different login screen. First login screen is for company. The company must be login to reach cargo and location information. Also, the company can manage couriers on the screen. Second screen is for couriers; every couriers has different account. When courier’s login, they can set cargo information. In addition, they can follow the route on the screen. If user information is entered incorrectly, the user is notified and the user's input is restricted after 10 incorrect entries.

Loading the Cargoes

In this function, courier scan the cargoes code via camera. The code includes cargo information such as address,size,fragile. This information transfers the software from company database. In this way, couriers can be informed about the cargo and they are more careful when transporting cargo. Thus, possible damage can be prevented.

Shortest Path and Routing

When courier scan all cargoes, routing algorithm will run. The algorithm needs current courier location. For this process, the software uses GPS data. After that, results will see on map. The next target will be shown a red dot. Also, software will be shown target distance and estimated arrival time. When the courier arrives delivery location, cargoes information will be highlighted.

Cargo Delivery

When estimated arrival time 30 minute, software sends mail to customer. In delivery time, courier take e-sign from customer. The cargo delivered option is checked and the cargo has been removed from the system. If the cargo cannot be delivered, the software automatically switches to the next delivery address. Undeliverable cargo is kept in a list. At the end of the day, it is checked and delivered to the cargo branch. The information that the cargo has reached to the branch is given to the customer by mail.

1. Non-Functional Requirements
2. Performance requirements

In the cargo package loading phase, the package information of the barcode is taken from the cargo company database. In terms of performance, only the address of the cargo receiver is requested here from server. Then, when the address is reached, the required information is requested via the server again. The purpose of this process is to reduce the wait during packet entry operations.

During the deployment of packages, route creation was supported by machine learning algorithms. This route finding process works faster than other algorithms. Here again the performance is gained.

1. Platform constraints

The target platform is mobile devices which are Android and iOS operating systems. But first we will implement the android application. If we will decide to improve this application in the future, either iOS application will be implemented.

1. Modifiability

The operating policies of each firm may not be the same. According to companies, we can easily provide this when we need to change the operation of the application. For example, route calculation can be done by calculating more than one route calculation algorithm or by own algorithms.

1. ***Reliability***

We haven't had a chance to try it yet, but we're guessing that sometimes we can't accurately identify the address on the map. This is usually due to the accuracy of the address the user has entered.

An address outside the boundaries on the map of the carrier firm can be displayed. In order to minimize such problems, various algorithms will be used to determine whether the address entered according to the range of the cargo company is deported.

1. Usability

The system is very simple to use. It also helps transport staff with voice commands during transport. When the address is reached, it will facilitate the work of the officer by bringing the details of the relevant address.

CHAPTER FOUR

DESIGN

1. Architectural View

There are many controllers to be process employee’s commands, which gives any order. Controllers have been designed to realize program functions. This project has four systems and three controllers. Each system and their controller are connected each other. The command is given any system, the command will send their controller, and in here command is treaded. In below figure 2, the architectural view of our system is given.

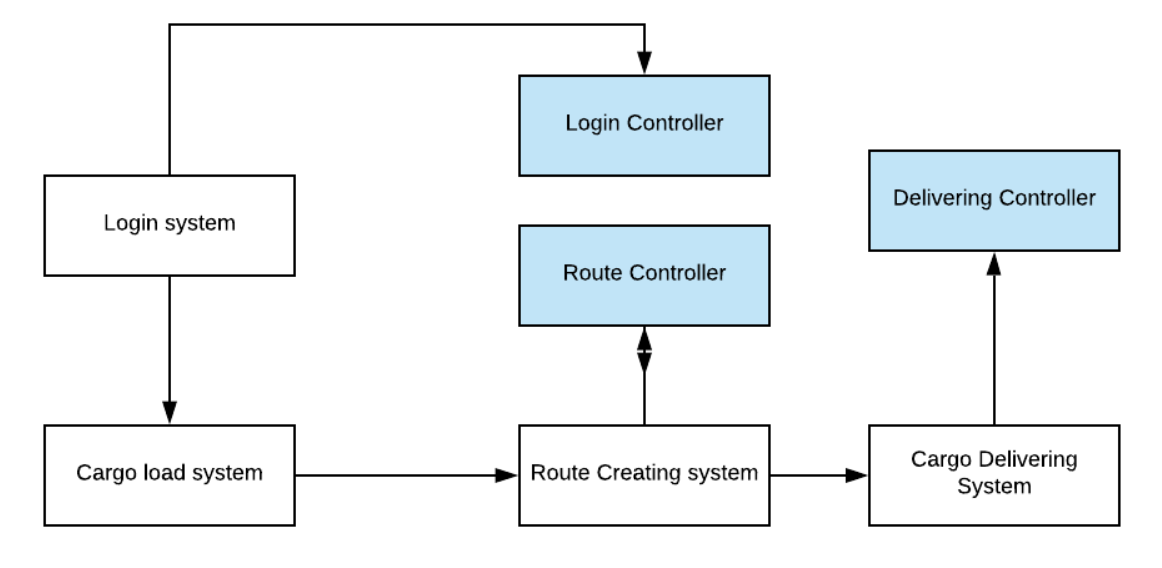


Figure 4.. Architectural View

1. Database Design/ER Diagram

The system cannot work without a database. Actually system works with two databases. One database is embedded in this program, and the other with a remote connection with a web service, we can access the firm database. There are six columns in our database, is given in figure 3. Some columns are connected between their primary and secondary key. Their relationships between columns are given in below.

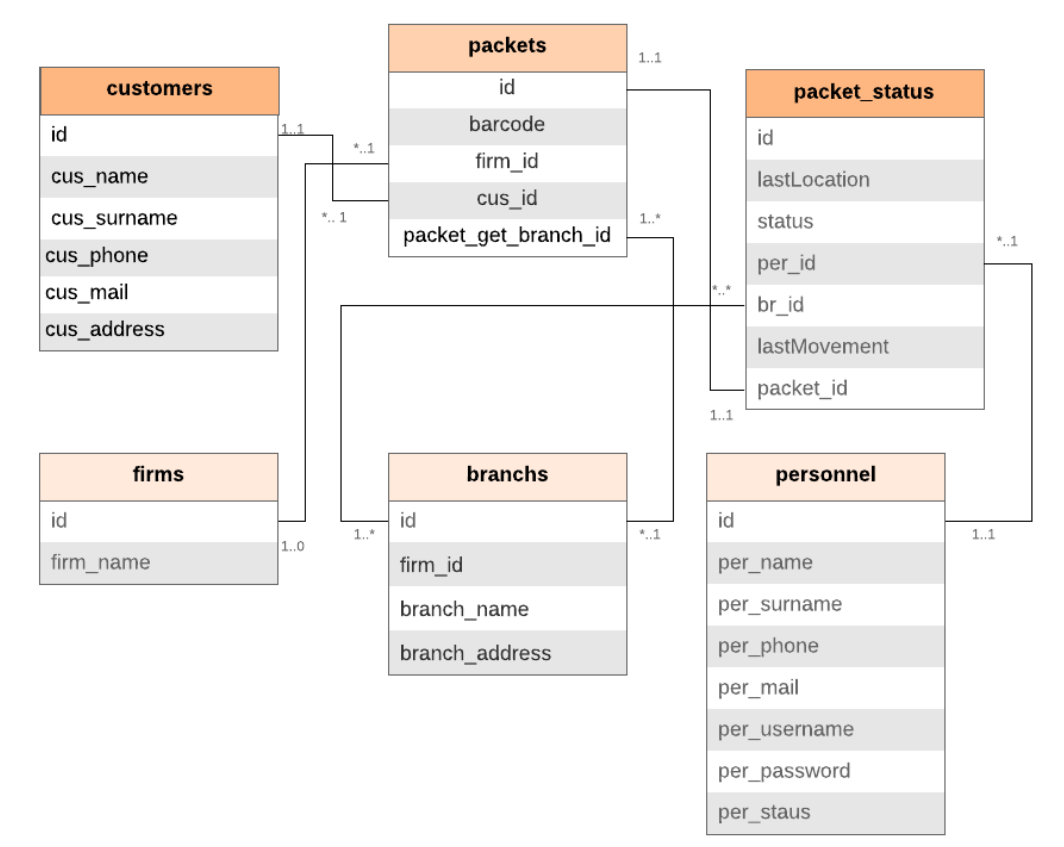


Figure . Database ER Diagram

1. UML Class Diagram

With UML Class diagram, the functions to be used are shown. Each class has one or one more than function. In additionally, each column have many attributes, which shows that attribute’s properties.

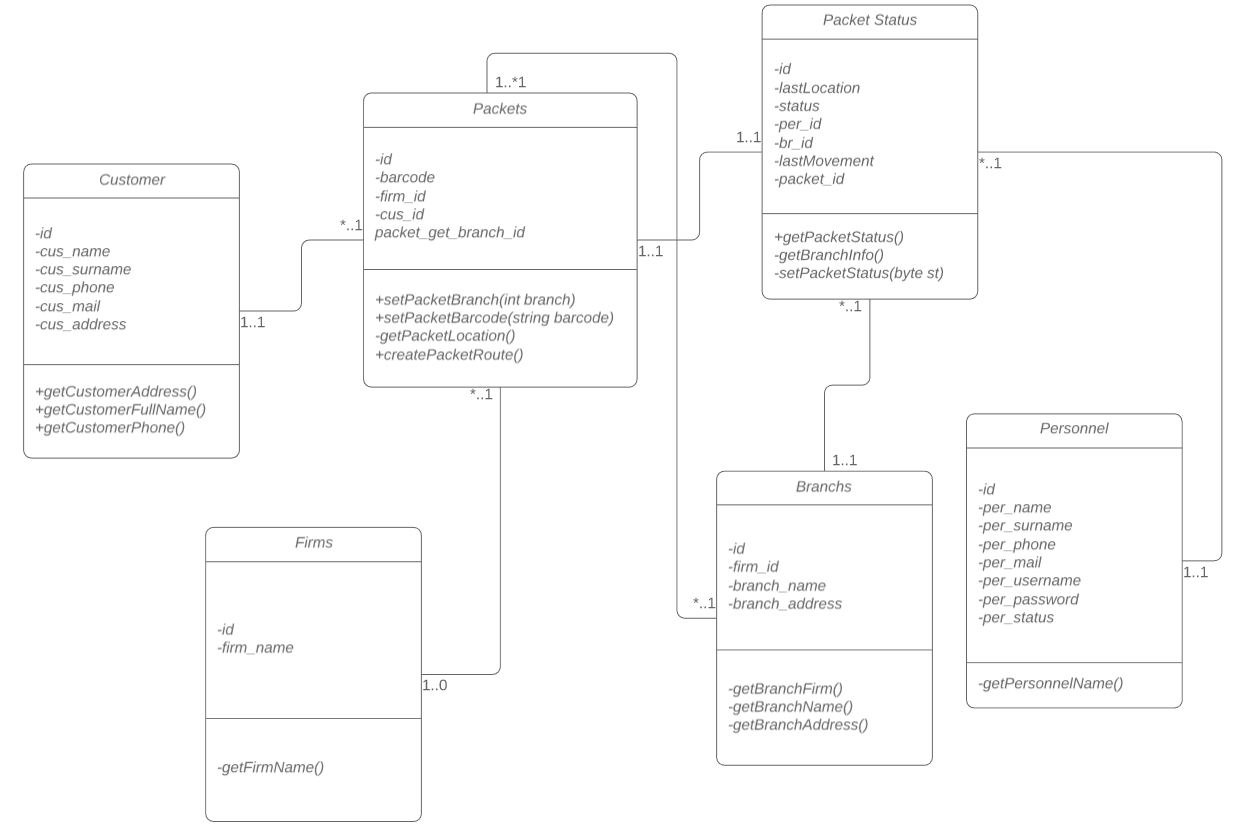


Figure 4 UML Class Diagram

In addition, at the same time there are four relation types that we using between columns’ identifier. These are 1-1, 1-0,\*-1,1-0. Each has different means.

1. UI Design

In the login screen, there are a logo, two textboxes, and login button. Employee must be fill the all spaces to login successfully.

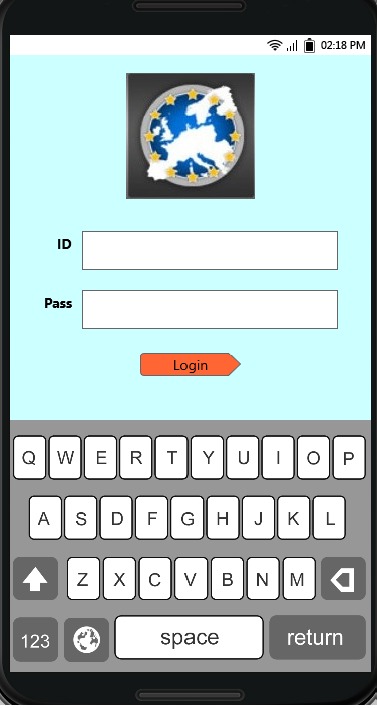


Figure 5 Mockup login screen

After the login page, main page is presented on the screen. Main screen has two buttons, which one button is load cargo button, and the other is cargo installation ok button. If the employee clicks the load cargo button, it’s camera will be open and ready to read the barcode of the packet.

In the below figure, we show that cargo transportation page, which contains many routes to access to customer, and in this page, our main genetic algorithm to be run. The best route is created, and cargo personnel will be routed.

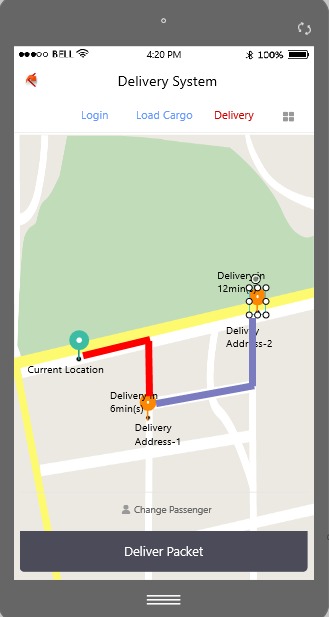


Figure 6 Mockup cargo route screen

When the cargo personnel have been arrived to customer address, and delivering process page in figure 7. Personnel waits customer sign on the sign page. And delivering will be successes.

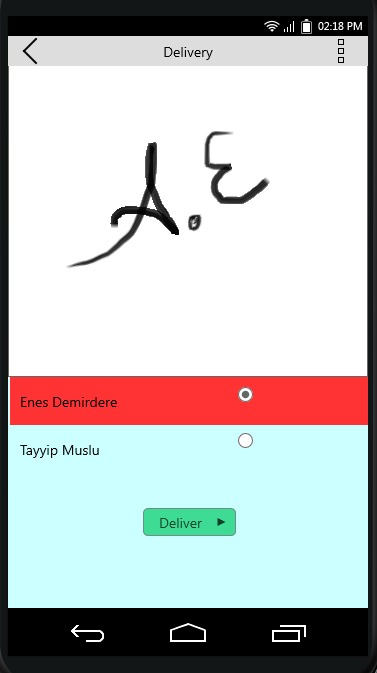


Figure 7 Mockup delivered screen

In cargo load screen page, the employee reads the packets barcode, the system gets packet’s address from firm database, and add the internal database. The adding process continue the end of packet.

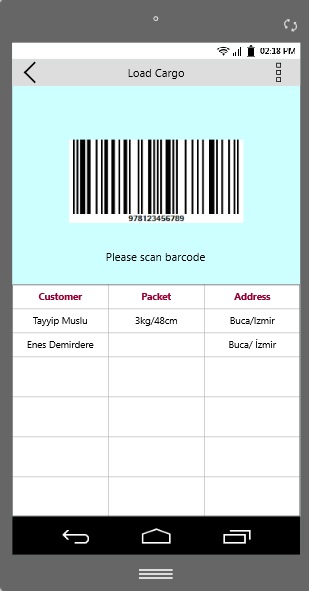


Figure 8 Mockup caro load screen

And then, the route creation process works with genetic algorithm, shows nearest location.

1. Use Cases

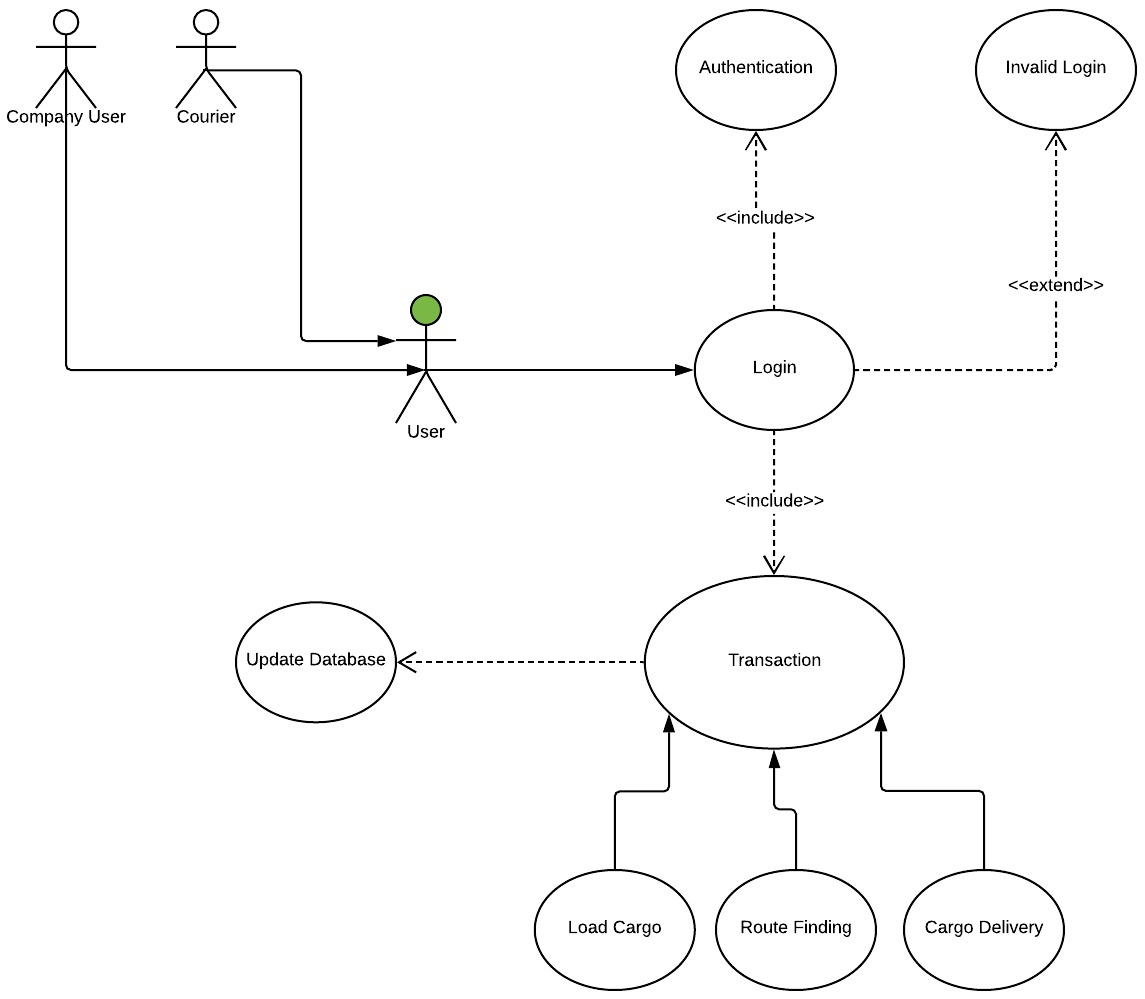


Figure 9 Login page use-case diagram

A use case diagram can summarize the details of your system's users and their interactions with the system. In figure 17 shows system’s transactions also we can see how many users on system.

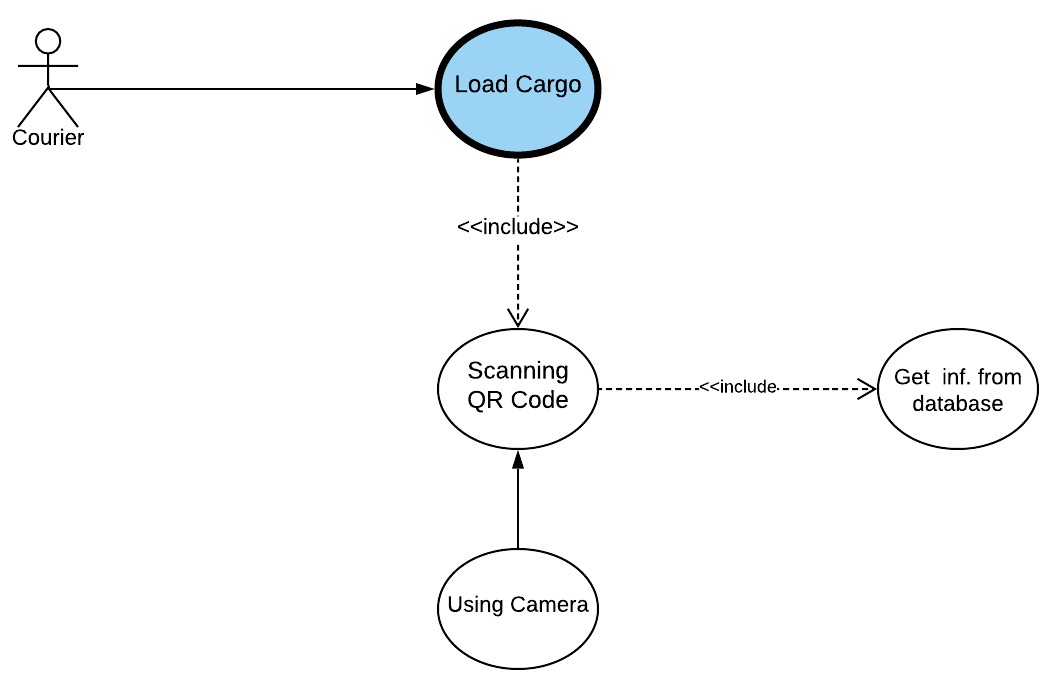


Figure 10 Cargo load page use-case diagram

Figure 10 shows us transaction of loading cargo. The transaction includes barcode scanning. Scanning barcode transaction uses camera. Barcode information taken from database.

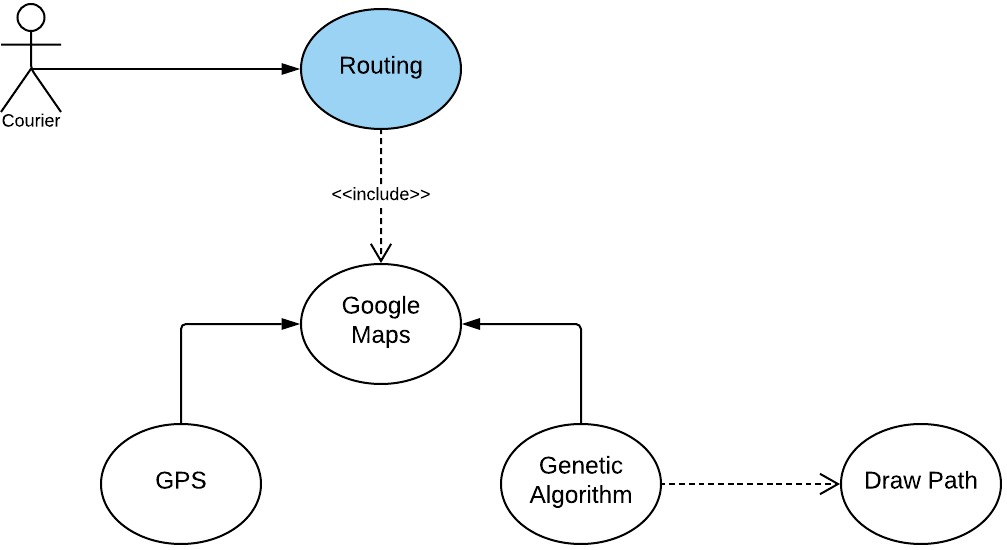


Figure 11 Route creation page use-case diagram

Figure 11 has shown routing transaction. The transaction uses google maps api. The api take location information from GPS. In addition, the transaction work with genetic algorithm, and best route draw on the map.

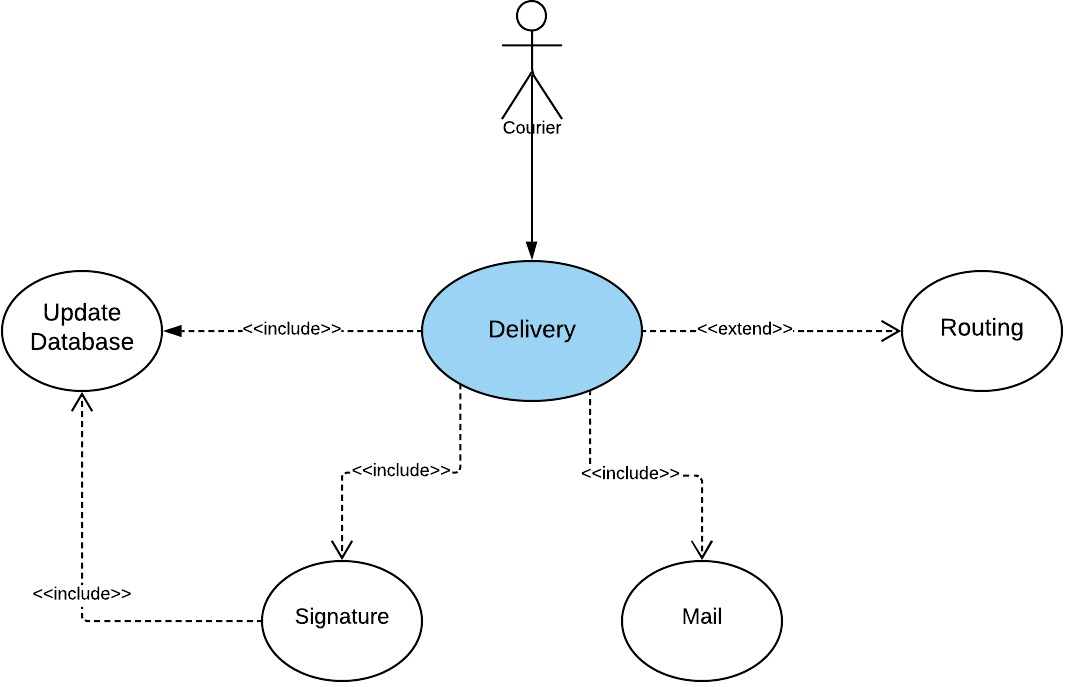


Figure 12 Delivery page use-case diagram

Figure 12 contains delivery transaction. The transaction uses for packet delivery. Customer uses action of signature. The transaction includes to draw signature on the screen. Also, mail transaction sends delivery information to customer.

1. Sequence Diagram

The first sequence diagram contains the login process. In there, the employee fills the all spaces, and click the login button. Send the login form to event request, the request goes to internal database. The login validation is made in here, the result could be two sceneries, first scenery the login failure, and the other the login success.

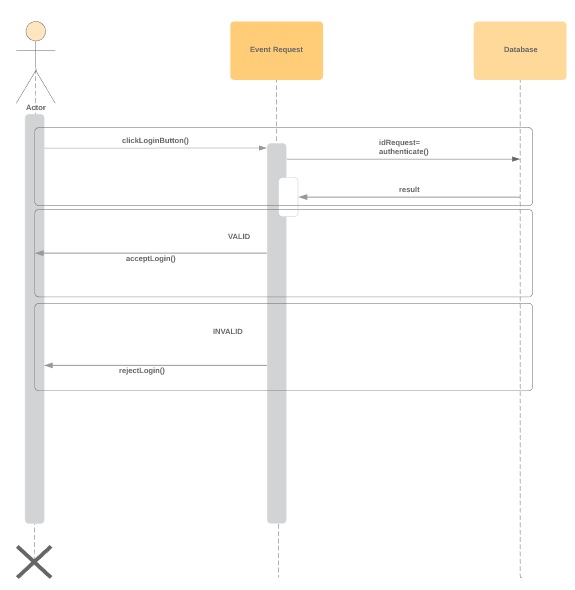


Figure 13 Login screen sequence diagram

The second scenery is cargo load page; the employee reads the barcode with camera. The program gets the barcode, and request the firm database to get packet’s address, and add to the internal database.

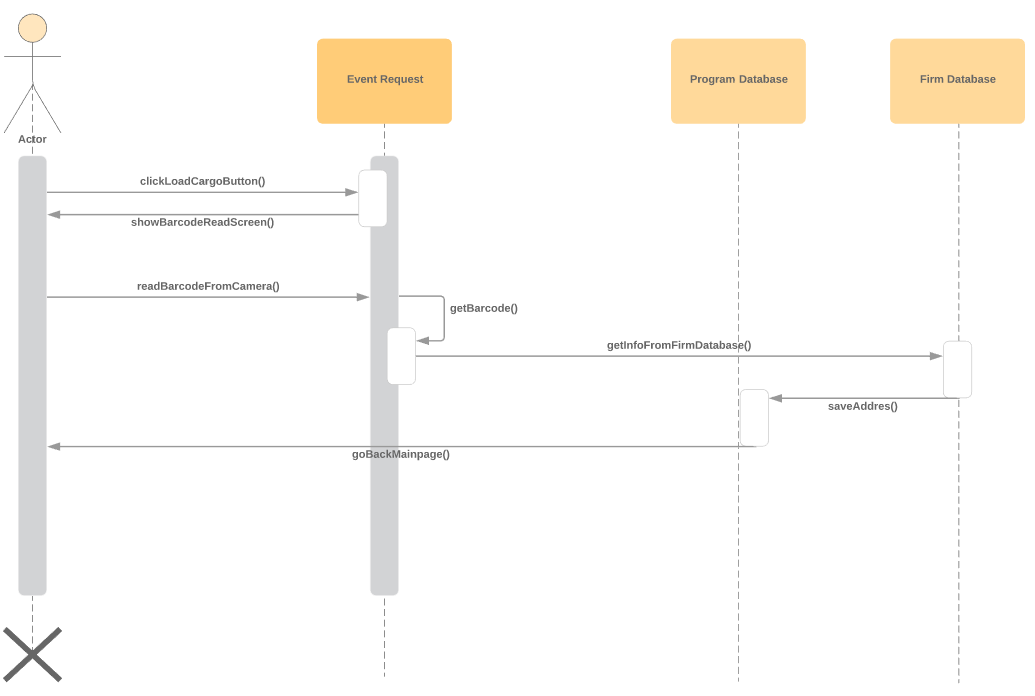


Figure 14 Load cargo page sequence diagram

The cargo delivering page sequence diagram is in here figure 15. In there, at the same time, there is the customer. The employee will get the customer.

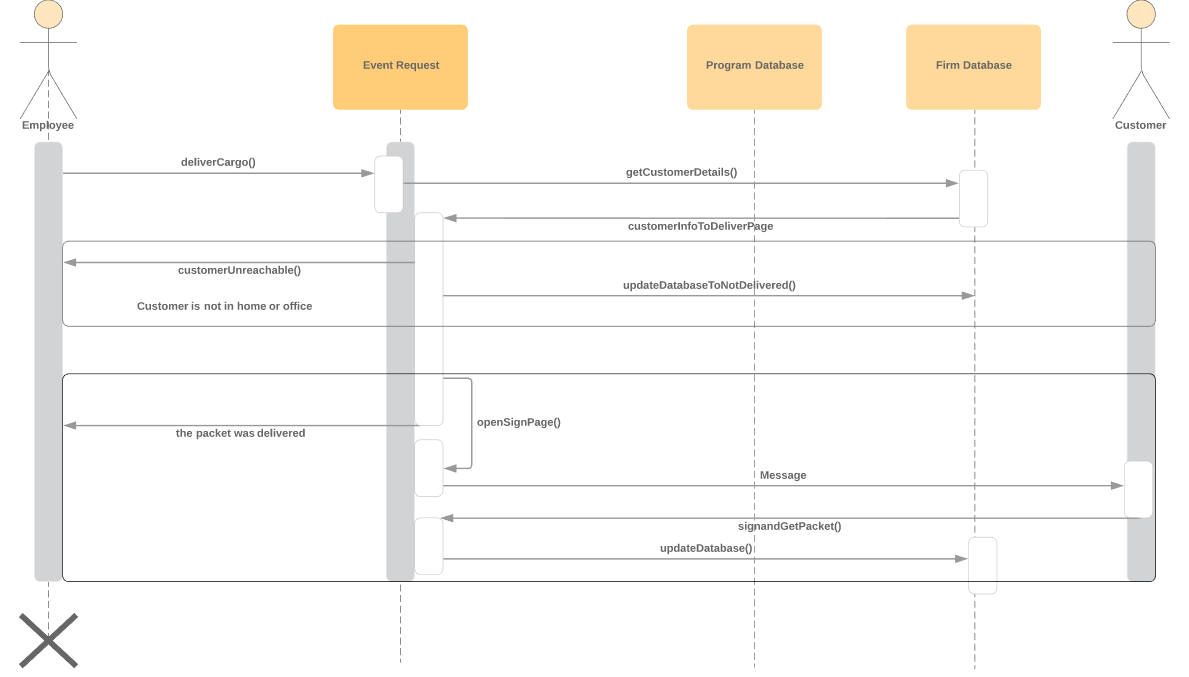


Figure 15 Cargo delivering screen sequence diagram

Customer address visited by employee, if there is the customer in address, the packet delivering process is started. Packet gives the customer; employee waits the signature from customer. And the other scenery, the customer is not in its address. Employee cannot deliver the packet.

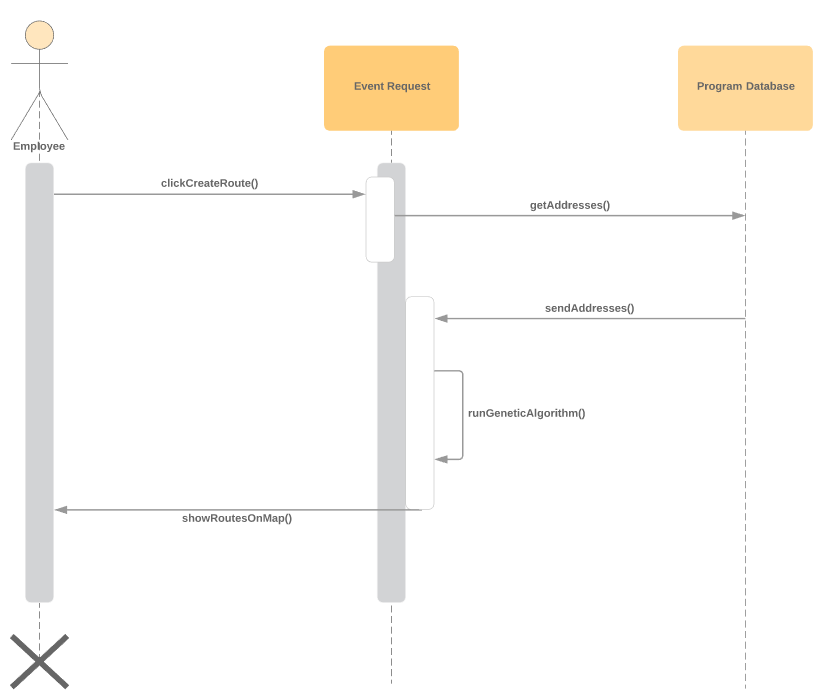


Figure 16 Route creation sequence diagram

All of the packet read from the employee, the genetic algorithm runs, algorithm gets the all address from internal database, and returns a route to employee.

1. Activity Diagram

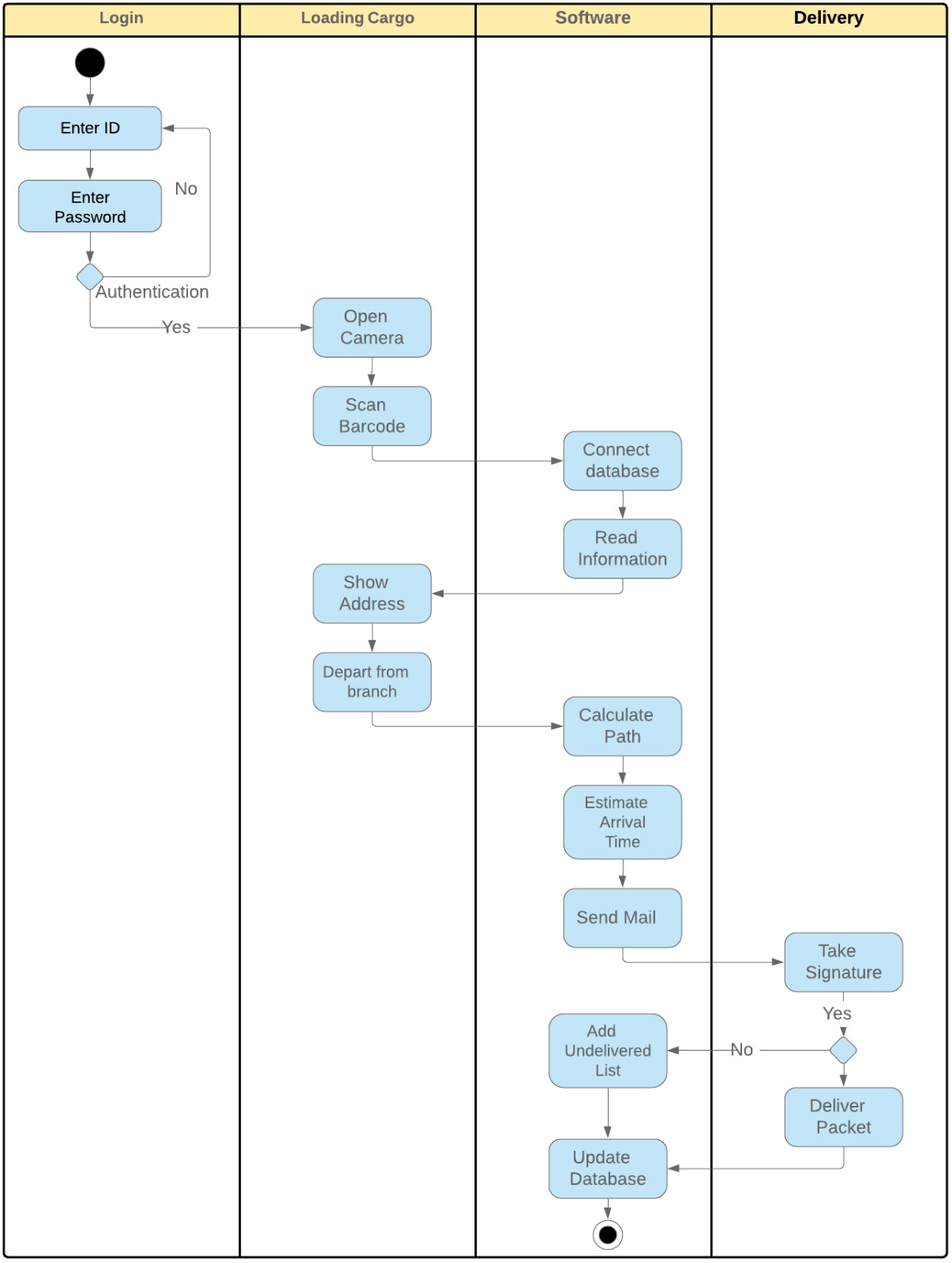


Figure 17 Activity diagram

Activity Diagrams describe how activities are coordinated to provide a service which can be at different levels of abstraction. It is also suitable for modeling how a collection of use cases coordinates to represent business workflows. In Figure 17, login part shows us how to login on software step by step. Also, loading cargo part and delivery part show us workflow step by step and connection of software.

1. Deployment Diagram

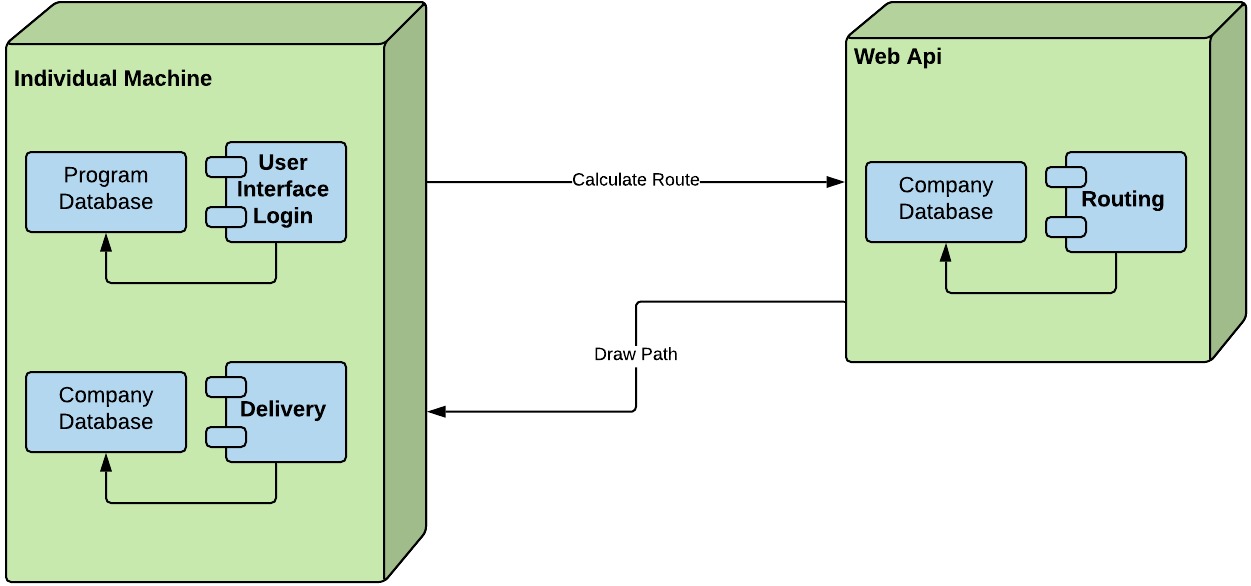


Figure 18 Deployment diagram

Deployment diagrams are typically used to visualize the physical hardware and software of a system. Using it you can understand how the system will be physically deployed on the hardware. Figure 18 shows us how to connecting phone and web server. Also, the figure contains database connections.

**REFERENCES**

Apaydin, O., Gonullu M. T. (2007) . "Route optimization for solid waste collection: Trabzon (Turkey) case study." Global NEST Journal 9.1 : 6-11.

Bošković G.B, Jovičić G.R.,Jovičić N. M., Vujić G.V. (2010) . "Route optimization to increase energy efficiency and reduce fuel consumption of communal vehicles." Thermal Science 14.5: 67.

Bräysy, O. , Gendreau, M. (2002) 10: 211. https://doi.org/10.1007/BF02579017 (2019)

Dantzig, G.B(1962).: Linear Programming and Extensions. Princeton University Press, Princeton .

Diaz F. , Masegosa A. D., Onieva E.,Osaba E.,Perallos A.,Yang X.E. (2017) "A discrete firefly algorithm to solve a rich vehicle routing problem modelling a newspaper distribution system with recycling policy." Soft Computing 21.18 : 5295-5308.

Dijkstra, E.W.(1959): A note on two problems in connexion with graphs. Numerische Mathematik 1, 269–271.

Ester M., Kriegel H., Sander J., and Xu X.(1996). “A density-based algorithm for discovering clusters in large spatial databases with noise,” in Proc. 2nd Int. Conf. Knowledge Discovery and Data Mining (KDD’96), pp. 226–231.

Fernandez M.,Lai F., Michal S., So W (2018) Improving Operations with Route Optimization July 2018 <https://towardsdatascience.com/improving-operations-with-route-optimization-4b8a3701ca39> (2019/15/05)

Fister D., Fister, I., Fister Jr, Yang, X.S.,(2014). Firefly algorithm: A brief review of the expanding literature. In: Cuckoo Search and Firefly Algorithm. Springer 347–360.

Goldberg, A.V., Harrelson, C.(2005). Computing the shortest path: A∗ meets graph theory. In: 16th ACM-SIAM Symposium on Discrete Algorithms, pp. 156–165. ACM Press, New York .

Hanshar F., Ombuki B., Ross B.J.(2006) . "Multi-objective genetic algorithms for vehicle routing problem with time windows." Applied Intelligence 24.1: 17-30.

Hart, P.E., Nilsson, N.J., Raphael, B.(1968). A formal basis for the heuristic determination of minimum cost paths. IEEE Transactions on System Science and Cybernetics 4(2), pp.100–107 .

Meyer, U.(2001). Single-source shortest-paths on arbitrary directed graphs in linear average-case time. In: 12th Symposium on Discrete Algorithms. pp. 797–806.

Nagata Y. (2007). Edge Assembly Crossover for the Capacitated Vehicle Routing Problem. In: Cotta C., van Hemert J. (eds) Evolutionary Computation in Combinatorial Optimization. EvoCOP 2007. Lecture Notes in Computer Science, vol 4446. Springer, Berlin, Heidelberg.

Sanders P.,Schultes D., (2007) Engineering Fast Route Planning Algorithms <http://algo2.iti.kit.edu/schultes/hwy/weaOverview.pdf> (2019/20/05)

Tan X., Zhang J, Zhuo X.. (2006) Ant Colony System for Optimizing Vehicle Routing Problem with Time Windows (VRPTW). In: Huang DS., Li K., Irwin G.W. (eds) Computational Intelligence and Bioinformatics. ICIC 2006. Lecture Notes in Computer Science, vol 4115. Springer, Berlin, Heidelberg.

Thorup, M.(2003). Integer priority queues with decrease key in constant time and the single source shortest paths problem. In: 35th ACM Symposium on Theory of Computing. pp. 149–158.