|  |  |
| --- | --- |
| logo ngan.png | MINISTRY OF EDUCATION AND TRAINING |

|  |
| --- |
| **FPT UNIVERSITY** |
| Capstone Project Document |
| Build a Web Application  for manages all activities of  delivery service system by coach |
|  |
| |  |  | | --- | --- | | **Group 18** | | | **Group Members** | Đào Bảo Long – Team Leader – SE60690  Lê Phúc Lữ – Team Member – 60563  Nguyễn Thanh Tùng – Team Member – SE60609  Nguyễn Tấn Đức – Team Member – SE60660 | | **Supervisor** | Nguyễn Trọng Tài | | **Ext Supervisor** | N/A | | **Capstone Project Code** | i-Deliver | |
| Hồ Chí Minh City, April 2014 - |

*This page is intentionally left blank*

# Table of contents

[Table of contents 3](#_Toc385424823)

[Table of figures 5](#_Toc385424824)

[Terminology 5](#_Toc385424825)

[1. Introduction 6](#_Toc385424826)

[1.1. Introduction 6](#_Toc385424827)

[1.2. The initial idea of group 6](#_Toc385424828)

[1.3. Overview of existing methods 6](#_Toc385424829)

[1.1.1. Requests management 6](#_Toc385424830)

[1.1.2. Packages arrangement 6](#_Toc385424831)

[1.1.3. Packages management 6](#_Toc385424832)

[1.4. Limitations of existing system 6](#_Toc385424833)

[1.1.1. Requests management 6](#_Toc385424834)

[1.1.2. Packages management and arrangement 7](#_Toc385424835)

[1.1.3. Schedules and trips management 7](#_Toc385424836)

[1.5. Benefits of expected system 7](#_Toc385424837)

[1.6. Business outline 7](#_Toc385424838)

[1.7. Approaches 7](#_Toc385424839)

[1.8. Group of functions 8](#_Toc385424840)

[2. Software Project Management Plan (SPMP) 9](#_Toc385424841)

[2.1. Problem Definition 9](#_Toc385424843)

[2.1.1. Name of this Capstone Project 9](#_Toc385424846)

[2.1.2. Problem Abstract 9](#_Toc385424847)

[2.1.3. Project Overview 9](#_Toc385424848)

[2.2.3.1. The Current System 9](#_Toc385424855)

[2.2.3.2. The Proposed System 9](#_Toc385424856)

[2.2.3.3. Boundaries of the System 10](#_Toc385424857)

[2.2.3.4. Development Environment 11](#_Toc385424858)

[2.2. Project organization 11](#_Toc385424859)

[2.3.1. Software Process Model 11](#_Toc385424862)

[3. Software Requirement Specifications (SRS) 12](#_Toc385424863)

[3.1. User Requirement Specification 12](#_Toc385424865)

[3.1.1. Guest Requirements 12](#_Toc385424868)

[3.1.2. Logged User Requirements 13](#_Toc385424869)

[3.1.3. Customer Requirements 13](#_Toc385424870)

[3.1.4. System Admin Requirements 13](#_Toc385424871)

[3.1.5. Staff Requirements 13](#_Toc385424872)

[3.2. Entity Relationship Diagram 13](#_Toc385424873)

[4. Software Design Description (SDD) 14](#_Toc385424874)

[4.1. System Architectural Design 14](#_Toc385424876)

[4.2. Component Diagram 15](#_Toc385424877)

# Table of figures

[Figure 1. An overview of the i-Deliver system 10](#_Toc385451126)

[Figure 2. The waterfall software lifecycle model 11](#_Toc385451127)

[Figure 3. Overall usecase 14](#_Toc385451128)

[Figure 4. Model-View-Controller Architecture 18](#_Toc385451129)

[Diagram 1. Actor overview diagram 12](#_Toc385451123)

[Diagram 2. Entities Relationship Diagram 15](#_Toc385451124)

[Diagram 3. Component Diagram 19](#_Toc385451125)

# Terminology

|  |  |  |
| --- | --- | --- |
| No. | Terminology | Explanation |
| 1. | Customer | A person who needs to book rooms in hotels at their destination |
| 2. | Staff | A person in charge of managing their hotels |
| 3. | System Administrator | A system that provides hotel data including hotel information, room availability, and price changes |
| 4. | Delivery Request | A request sent by a customer to the staff to ask for a delivery request with the I-DELIVER system |

# Introduction

## Introduction

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Project Title:** | | *Build a web application for manage the activities of delivery service system by coach* | | | |
| **Start Date:** | | Jan 6, 2014 | | **Finished Date:** |  |
| **No** | **Full name** | | **Role** | **Position** | **Contact** |
| 1. | Nguyễn Trọng Tài | | Supervisor | Lecturer | [TaiNT@fpt.edu.vn](mailto:PhuongLHK@fpt.edu.vn) |
| 2. | Đào Bảo Long | | Developer, Tester | Team Leader | [LongDBSE60690@fpt.edu.vn](mailto:LongDBSE60690@fpt.edu.vn) |
| 3. | Lê Phúc Lữ | | Developer, Tester | Member | [LuLP60563@fpt.edu.vn](mailto:DucBM60324@fpt.edu.vn) |
| 4. | Nguyễn Thanh Tùng | | Developer, Tester | Member | [TungNTSE60609@fpt.edu.vn](mailto:TungNTSE60609@fpt.edu.vn) |
| 5. | Nguyễn Tấn Đức | | Developer, Tester | Member | [DucNTSE60660@fpt.edu.vn](mailto:DucNTSE60660@fpt.edu.vn) |

## The initial idea of group

Nowadays, freight traffic between cities in a day is huge. We have the passenger transportation service providers like Mai Linh or the goods delivery service providers like Tin Thanh, Hop Nhat. And we also have the combination like Phuong Trang. Before working on this project, our team conducted a survey in a passenger transportation service provider. Here are the major findings during the survey:

**Phuong Trang Travel & Transportation Company:**

This company provides passenger transport service using coach. Beside this service, this company also provides goods delivery service. They receive a lot delivery requests per day but all of them still managed by staff manually using papers/books. It makes planning and scheduling for goods delivery become complicated beside passenger transportation arrangement, also makes hard to manage all goods to ensure integrity.

From the problems above, our team decided to develop a delivery service system for passenger transportation service providers which using coach. It operates difference from professional delivery service which using cars specially made for goods delivery.

## Overview of existing methods

### Requests management

The goods delivery service’s staffs have to write down all of delivery requests, monitoring – planning – scheduling manually using papers/books and their memory.

### Packages arrangement

They just simply fill-up empty cargo compartments of each coach as much as possible.

### Packages management

Tracking, searching … is temporary unavailable

## Limitations of existing system



### Requests management

By using papers/books and memory, staffs cannot ensure information accuracy; controllable requests, delivery status of requests.

### Packages management and arrangement

They don’t have any specific fee calculating formula, it’s just estimation.

They don’t have any plan to arrange package on each coach so it’s hard to balance the freight on each route.

### Schedules and trips management

They don’t have detail plans to schedule for each trip, all still managed by demands on real-time.

## Benefits of expected system

The project aims to develop a web-based application that

* Create an easy way for customers to make goods delivery request by allow them to post a new requests on website with registered account then they can tracking their package to ensure delivery.
* Mainly assists service providers’ staff to manage all goods delivery request by using optimized planning and scheduling algorithms, manage the fee of the delivery requests
* Allow customers to comment and rating for the service.

## Business outline

Following the project objectives above, the scope of the project is constrained to the following statements:

* The users of the web-application should interact with the web-application itself through a friendly and attractive user interface.
* *For* ***customers***, this web-application should provide fundamental functions such as register, login then post, edit and cancel goods delivery requests. Through integrated e-payment services, they can pay for their delivery requests. They also can search for posted requested, tracking their package. After using service, customers can comment and rating for the service.
* *For* ***system administrators***, the web-application should allow them to monitor fundamental information about customers, staffs, coaches, routes, and fee.
* *For* ***staffs****,* i-Deliver is a management system allows them to import request, fee calculating, invoice making, planning and scheduling for goods delivery, assign packages for specific coach, specific route.

## Approaches

* Adopt the perspective of a customer during the whole development process, in order to develop a web-based application that promotes usability and interactivity as much as possible.
* Negotiate with web service providers to use their services, and then agree upon how the application communicates with the web services, what information should be retained and what should be eliminated.
* Try to provide staffs the customized tools for manage requests easily.
* Conduct research on how to determine the most appropriate fee based on the average fee of the same service providers in the market.
* Conduct research on how to optimize planning and scheduling for goods delivery.

## Group of functions

|  |  |
| --- | --- |
| Functions for customers | * Allow customers register, login, post/edit/cancel requests, search for, and tracking requests; * Allow customers to rate by different criteria and write comments/reviews on service; |
| Functions for staffs | * Allow staffs to manage fundamental information about requests, fee calculating, invoice making; * Allow staffs to planning and scheduling for goods delivery; * Allow staffs to monitoring and assigning for coaches and routes; |
| Functions for system admins | * Allow system admins to manage all relevant information about the system, customers, staffs, coaches and routes; * Allow system admins to grant access rights to other users of the system; * Allow system admins to decide fee calculating formulas; * Allow system admins to collect and export data to statistic. |

# Software Project Management Plan (SPMP)



## Problem Definition



### Name of this Capstone Project

|  |  |
| --- | --- |
| **Official name** | Building a web-based application that manages the activities of delivery service system by coach |
| **Vietnamese name** | Xây dựng ứng dụng web quản lý các hoạt động cho dịch vụ vận chuyển hàng hóa thông qua hệ thống xe khách đường dài |
| **Abbreviation** | i-Deliver |

### Problem Abstract

The idea of the project is to develop a web-based application that assists delivery service staffs/administrators in manages goods delivery requests, package arrangement, and planning and scheduling, fee management. It also helps customers to make goods delivery request, searching and tracking their package, make a payment online.

### Project Overview



#### The Current System

The idea of this project is developing a delivery service system for passenger transportation service providers which using coach. Before working on this project, we conducted a survey about traditional goods delivery process.

**Traditional goods delivery process:**

Customers will pick their package to delivery service station and provide the name of receiver, destination, and type of goods, weight and size. Then staffs of delivery service will calculate transport fee and give the customer a package invoice which contain provided information and destination station address.

#### The Proposed System

By working on this project, we will develop a service system that assists delivery services providers in Ho Chi Minh City to be closer to their customers. The system has some significant features:

**Create user-oriented interfaces for administrators to simplify management:**

The i-Deliver system support administrators to manage delivery fee like define or edit fee factor. It also helps them to manage related activities of a journey, included trips, stages, stations and routes. Of course the system will provide mechanisms to manage users/staffs.

**Support staffs in delivery requests management, package arrangement, planning and scheduling:**

This system provide easy-to-use interface for requests management like make a new request, edit request information, fee calculating, invoice making, packages arrangement and planning and scheduling, assign packages for specific coach/route.

**Easy-to-use tool for customer to make goods delivery requests, searching and review/edit requests, tracking their package:**

The i-Deliver system is integrated with some technique to help making goods delivery requests online, searching then review or edit requests information, tracking customer’s packages by using request code.

**Optimize arrangement, planning and scheduling for package delivery process**

In traditional way, delivery service staffs have to planning, scheduling and arrange packages for each coaches manually. They just simply fill-up empty cargo compartments as much as possible. This system helps them do their works easily, efficiently by using tools, which optimized by arrangement, planning and scheduling algorithms.

**The i-Deliver system’s users:**

1. **Guests**: non-authorized members can register new account of the i-Deliver website to be granted full access permission or they just can search for routes which delivery service providers operating.
2. **Members:** guests had an authorized account can login to the i-Deliver website to

* Make goods delivery requests;
* Search for posted requests;
* Edit posted requests;
* Cancel posted requests (also have constraints);
* Tracking packages;
* Comment and rating for service;
* Statistics

1. **Administrators**: owners of the i-Deliver website who have highest permission can

* Create new staff account;
* Edit staff account information;
* Delete staff account;
* Define fee calculating formular and fixed value;
* Edit fee calculating formular and fixed value;
* Statistics

1. **Staffs**: users who has account which created by adminstrators have right to

* Manage requests: approve, reject, update status, assign and scheduling;
* Search for packages, routes or members;
* Manage journeys: include add new, edit or delete routes, stations, trips and coaches;
* Manage comments and rates: delete violated/spam comments

Figure . An overview of the i-Deliver system

#### Boundaries of the System

There is no previous version of this system. The product will be developed from scratch, independent of any current system.

As said previously, the system under development is not a delivery service management system. It does not provide mechanisms to manage all activities related to goods delivery. In our team’s scope, the i-Deliver is a system that manage related activities of transportation service providers which using coaches. It means that the service provider only working on 2-stations routes (likes Saigon-Nha Trang, Saigon-Vung Tau, Saigon-Hanoi), they don’t provide mechanism to deliver packages to stations between starting point and destination. That is our team’s future plan for this system.

It focuses mainly on providing easy-to-use interfaces and tools, which support both customers and delivery service staffs.

The final product of this Capstone Project includes

* A service portal which helps customers make goods delivery requests;
* A management module for staffs/administrators of delivery service;
* All the documents involved in the development process.

#### Development Environment

Below is the list of hardware and software requirements needed for the development environment of the project.

**Hardware requirements**

* Personal computers for developing with the minimum configuration: 2 Gb of RAM, 100GB of hard disk, Core 2 Duo 2.0 Ghz;
* A server computer for testing with the minimum configuration: 4 Gb of RAM, 100GB of hard disk, Core 2 Duo 2.0 Ghz;
* All computers must be connected to the Internet.

**Software requirements**

* Operating system: Windows 7 or above;
* Web Server: IIS Express 8;
* Framework: .NET Framework 4.5;
* IDE: Visual Studio 2012;
* DBMS: SQL Server 2008 R2;
* Source Control: Tortoise SVN 1.8.4.

## Project organization



### Software Process Model

The waterfall software lifecycle model will be used to guide the development of the system. The waterfall model includes five major phases as in the figure below, enforcing moving to the next phase only after completion of the previous phase.

Requirement Specifications

System and Software Designs

Implementation and Unit Testing

Integration and System Testing

Operation and Maintenance

Figure . The waterfall software lifecycle model

# Software Requirement Specifications (SRS)



## User Requirement Specification

The system should allow 4 types of actors, namely Guest, Customer, Staff, and System Administrator, and an abstract actor named Logged User, to interact with. Each of these types of user is granted a set of functions as specified below.

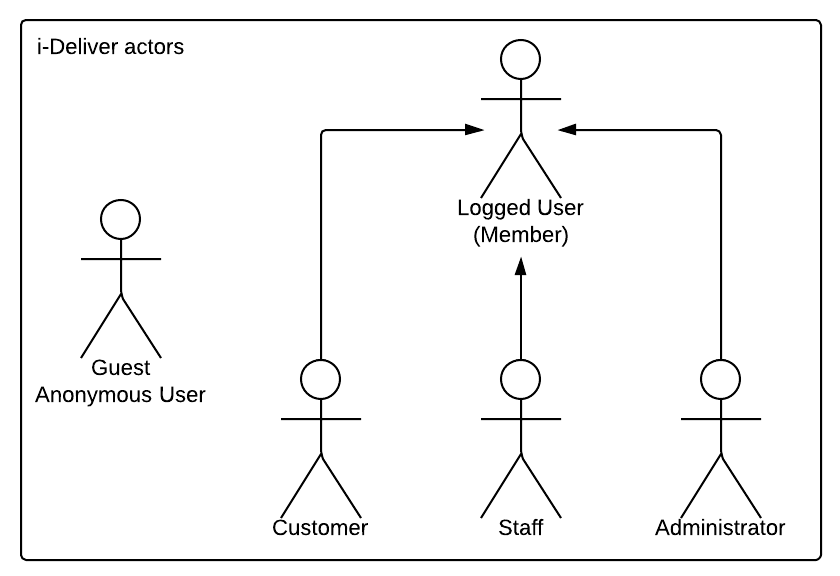


Diagram . Actor overview diagram



### Guest Requirements

A guest is an unauthenticated user of the website. He or she can:

* Log in with his authorized account;
* Register a new account;
* View, search for, and filter routes by a number of criteria that suit his or her needs;
* View details of an arbitrary routes.

### Logged User Requirements

A logged user is an authenticated user of the website (e.g., a user who logged on to the system with a valid username and password). They can:

* Log out;
* Change their own password;
* View and update their account details.

Note that that this actor is an abstract one and does not represent a real actor in practice.

### Customer Requirements

A Customer is a logged user (see 3.1.2. Logged User Requirement) and is given all the functions of a logged user. In addition, he or she is granted all the functions of a guest (see 3.1.1. Guest Requirement) except for the Log in and Register functions. A Customer also has his or her own set of functions:

* Create and submit the goods delivery requests;
* Cancel unwanted requests if they weren’t paid or approved;
* Edit submitted delivery information in a limit duration;
* Leave reviews about the service;
* Rate hotels by multiple criteria;
* Tracking for package (when the package left station, time left to be delivered, …);
* Make an online payment via third-party service.

### System Admin Requirements

A System Administrator is a logged user (see 3.1.2. Logged User Requirement) and is given all the functions of a logged user. In addition, a system Administrator also has his or her set of functions:

* Manage Staff accounts;
* Add more System Administrator accounts;
* Manage fee value and calculating formula;
* Statistics: get data about revenue, performance.

### Staff Requirements

A Staff is a logged user (see 3.1.2. Logged User Requirement) and is given all the functions of a logged user. In addition, a system Administrator also has his or her own set of functions:

* Manage all requests: approve, reject online requests, create new, edit and cancel offline requests;
* Manage all related information about routes, stations, trips and coaches;
* Arrange and schedule for packages delivery;
* Searching for information about requests, routes, stations, trips and coaches;
* Manage comments and rating: delete violated comments and clear spam rating;

## System Requirement Specification (Specified Requirements)

### External Interface Requirements



#### User Interfaces

The i-Deliver website should adopt an attractive and user-friendly interface so that the users of the system can get a good experience browsing the site.

The pages should be well linked together to promote seamless navigation between them. The instructions should be specific and suggestive, so that the users will not be confused about what to do next.

Client postbacks should be limited wherever possible, in order to boost performance and ease of use.

Vietnamese should be the official language of the website as its target customers are mainly Vietnamese people. Vietnam Dong should be the official currency. Moreover, all other localizable information, such as date and time, should be Vietnamese-styled.

#### Hardware Interfaces

The i-Deliver website can be reached by personal computers that support Internet connection and web browsers. In order to get the best experience, the following conditions should be satisfied:

* 50 Kbps Internet connection or faster;
* 1 gigahertz (GHz) processor or faster;
* 512 megabytes (MB) RAM or more;

#### Software Interfaces

The software listed below is needed for the system to operate normally:

|  |  |  |  |
| --- | --- | --- | --- |
| **Software Name** | **Version** | **Cost** | **Provider** |
| Web Browser   * Internet Explorer * Google Chrome * Mozilla Firefox * Safari * Opera | 7.0 or higher | Free  Free  Free  Free  Free | Microsoft  Google  Mozilla  Apple  Opera |
| SQL Server Express | 10.0 | Free | Microsoft |
| .NET Framework | 4.5 | Free | Microsoft |

### Functional Requirements

#### Overall use case diagram

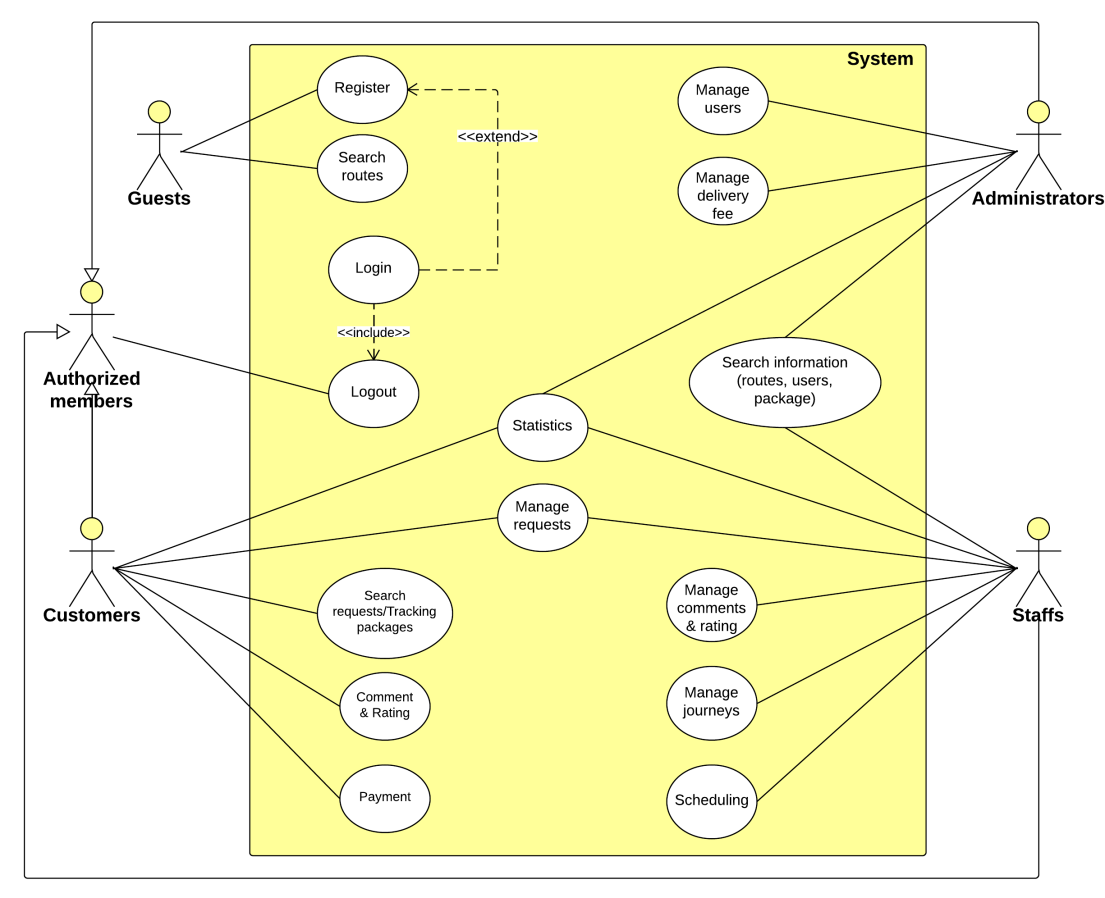


Figure . Overall usecase

## Entities Relationship Diagram

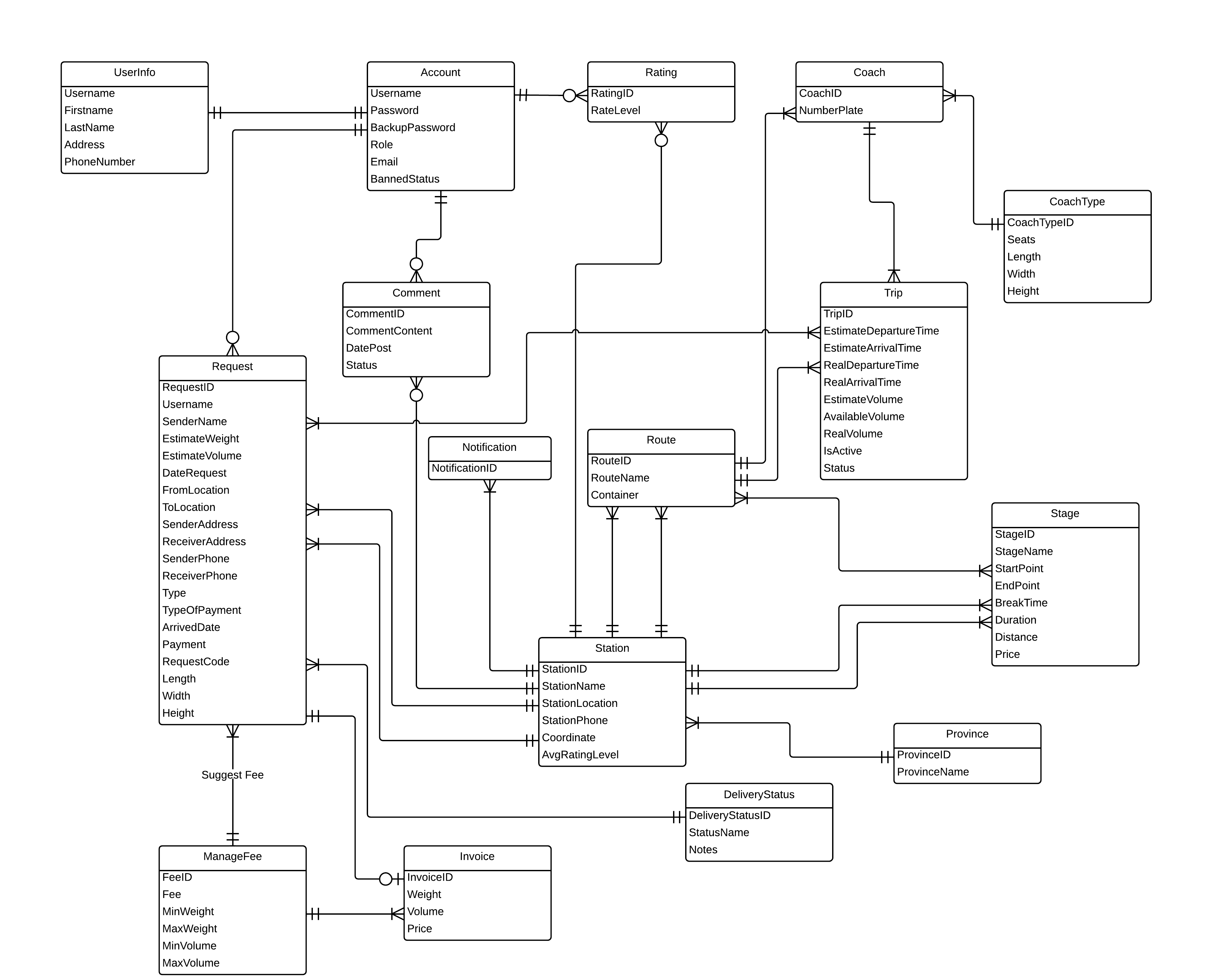
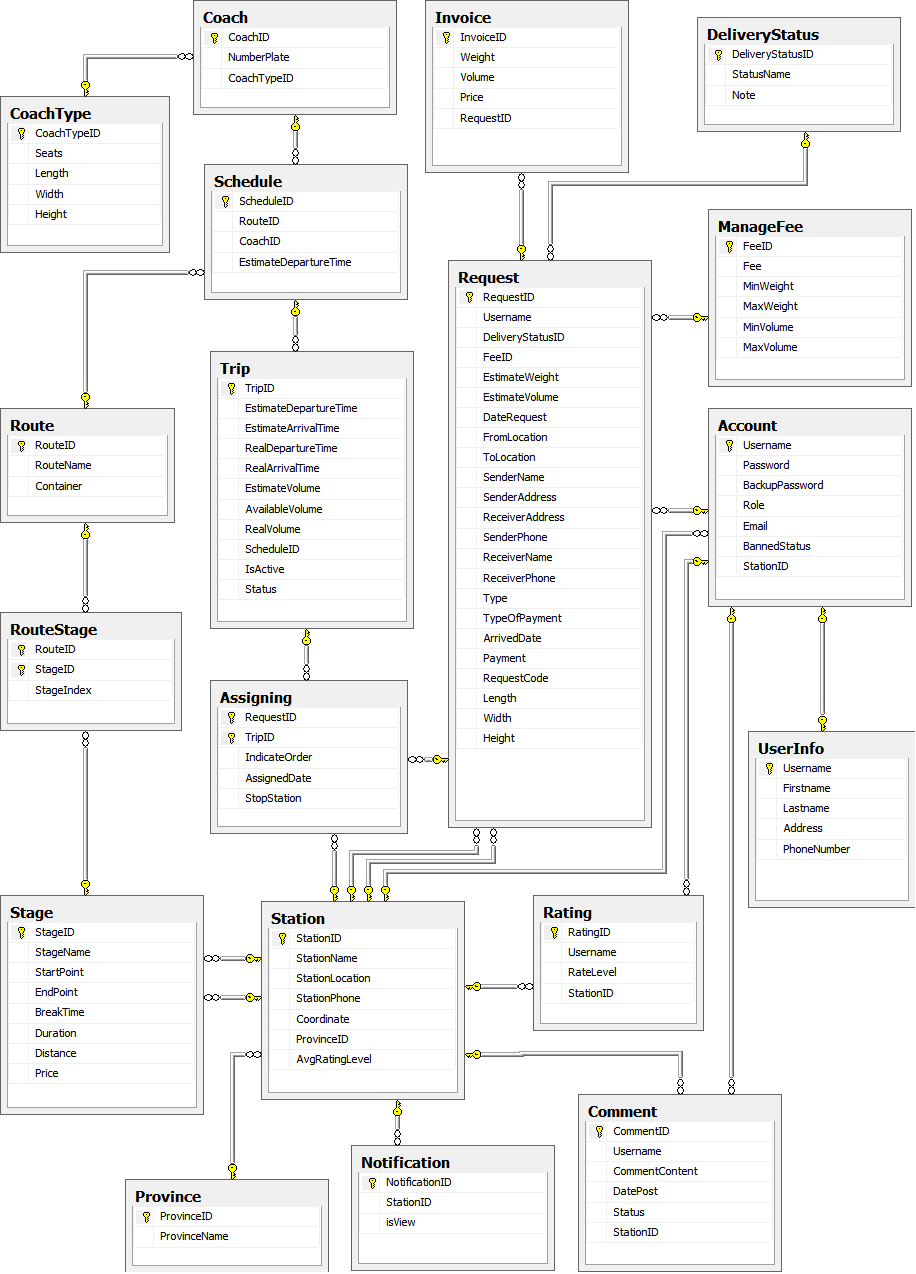


Diagram . Entities Relationship Diagram

Table of database description:

|  |  |  |
| --- | --- | --- |
| **Index** | **Table Name** | **Description** |
| 1 | Account | List of user that registered to the system with valid username and password. |
| 2 | UserInfo | Detail information of each user. |
| 3 | Coach | List of all coach the center has. |
| 4 | CoachType | List of type of coach base on the number of seat that the coach has. |
| 5 | Route | List of route that center can deliver to, including the route name. |
| 6 | Trip | Information of trip of each coach the center have everyday. |
| 7 | Station | The list of all stations the center has. |
| 8 | Schedule | The schedule of coach for run on the specific route during a day. |
| 9 | Request | Information of each request that user posted to the system. |
| 10 | DeliveryStatus | Status of the request of base on the action of user and staff. |
| 11 | Invoice | Detail of invoice corresonding to the approve request. |
| 12 | ManageFee | The fee of service base on the range of volume and range of weight. |
| 13 | Comment | Content of comment of user posted for each station. |
| 14 | Rating | Rating level that user rate for each station. |
| 15 | Assigning | Table of mapping between coach and request. |
| 16 | Stage | The partial of the way of route, each stage contains the start and end station and addition information of this stage |
| 17 | RouteStage | Mapping between route and stage |
| 18 | Notification | Check whether user has seen the notification or not yet. |
| 19 | Province | List of provinces that the system has some station in there. |



# Software Design Description (SDD)



## System Architectural Design

The MVC III (Model – View – Controller) pattern is used as the overall system architecture, because it specifies a clear distinction between the responsibilities of the components and is appropriate for developing web applications.

3. Exchange data

4. Results

**CONTROLLER**

* Intercepts user input;
* Coordinates the view and model;
* Handles communication between the model and data layer.

**VIEW**

* Binds to the model;
* Renders the UIs (HTML, CSS, JavaScript);
* Allows navigating between controllers.

**MODEL**

* Communicates with data source;
* Exposes functionalities, business logics, and data validation.

Database   
Server

Application Server

1. Request

8. Response

5. Select views

7. Changes made

2. Invoke methods

6. Query states

Method invocations

Change notifications

Figure . Model-View-Controller Architecture

## Component Diagram

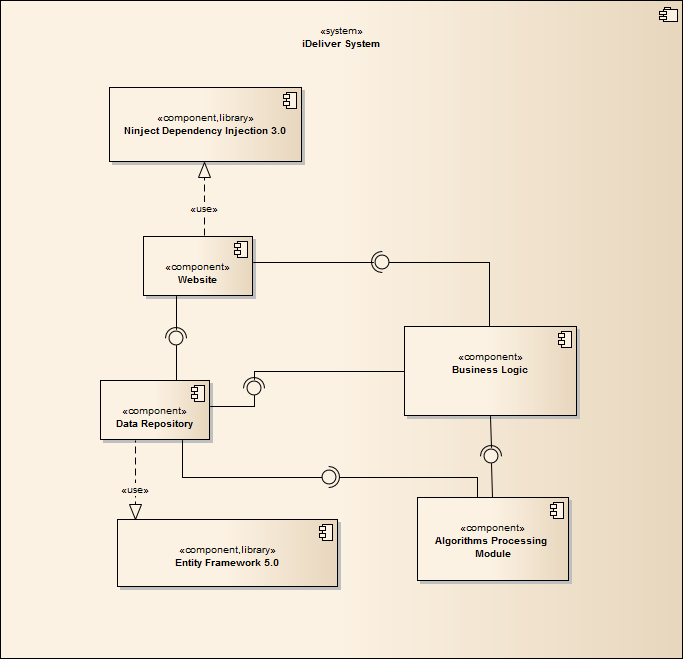


Diagram . Component Diagram

The i-Deliver system includes the following major components:

* **Entity Framework 5.0:** An object-relational mapper that enables working with relational data persistence using domain-specific objects;
* **Data Repository:** A data store that provides data access functionality, using the Entity Framework to communicates with the database;
* **Business Logic:** A module that uses interfaces exposed by the Data Repository to encapsulate important businesses of in the i-Deliver system;
* **Ninject Dependency Injection 3.0:** A lightweight dependency injection framework for .NET applications which helps split the application into a collection of loosely-coupled, highly-cohesive pieces, and then glue them back together in a flexible manner.
* **Website:** A web-application that helps exposes all the functionalities to end-users.

From the point of view of the MVC-II pattern, the components Entity Framework 5.0, Data Repository, Business Logic, and Ninject Dependency Injection 3.0 all belong to the Model part. The Controller and the View parts are wrapped in the component Website.

## Algorithm description

### Introduction

Scheduling problem is very popular in real life and appears in a lot of scopes, such as: education, human resource, transport, etc. For most of the scheduling problems, it has been shown that they are NP – complete problem, and they cannot be solved in polynomial time using a deterministic algorithm or find out the optimize solution within considered space in short time.

Our problem points to a type of this problem, which is scheduling the vehicles to deliver some package from a specific location to another location. The detail description of the problem:

At the transport center using coaches, they want to do an addition task for each coach as deliver the package that some customer sends to center along to each trip. Customers post request to the system to ask for transport the package with some specific information as:

* The time that user post request to the center.
* Volume of request (in dm3).
* Weight of request (in kilograms).
* Begin and end location of request.

The center has some resource as follow:

* Some stations with given address, exact coordinate with longitude and latitude.
* Some stages and each stage connect exactly two stations.
* Some routes and each route include one or multiple stages with the given order.
* Some coach with specific capacity (about volume can serve).
* Some schedule infers that which coach travels on which route and the coach start at what specific time of day.
* Some trip and each trip is an instant of schedule in a specific day.

The problem turns out is one type of scheduling problem: arrange resource for satisfying the given constraint. For detail, we want to assign each request for some trips such that:

- The total volume of requests assigned to trip not more than this coach’ volume.

- The time of delivery for each request not more than 5 days and the time for deliver each request as min as possible.

- In the list of trips, the departure time of each trip not earlier than the time for right previous trip depart to the middle station and the delta time between two point times (arrival time of previous trip and depart time of current trip) is 45 minutes.

- The way of deliver is straight forward, not go backward.

### Detail solution

We name the begin station of request is  and the end station of request is 

1. **Step 1:**

First, build an indirect and single graph  in which:

*  is the set of stations of the system.
*  is the set of stage connects two stations.

List all data from database and sort the list of request base on 3 criterions:

* The time of user post request (like the principles: first come, first served of queue).
* The volume of request (follow the greedy rules: to gain the most, first try to process the biggest one).
* The weight of request (the higher weight of request, the more money system can get).

1. **Step 2:**

Using Breath First Search algorithm to find some way connect  and . The original of this algorithm is just finding one satisfied solution and that solution travels on the smallest number of edges of graph. We extend it a little bit to find not only one solution but also multiple solutions. The maximum number of solutions allowed is  (to avoid the loop in graph with thick graph situation that leads to the number can be up to ).

In some cases, the direction of the partial of way is not always forward so we should reject the inappropriate way (the path in graph usually fallen into that case). To check the way not go backward, we use the mathematical idea as:

The point  with longitude and latitude are  respectively, we have the coordinate of its in space is  with  and  is the radius of Earth (we can assume that Earth is a quite sphere).

The angle of two vectors  is

.

with  is the dot product of two vectors and we can find this value easily by the found coordinate as above.

Assume that we have the list of stages:



with  is the number of stages. So we iterate through the list of stages to check whether two consecutive stages form an obtuse angle with value bigger than the allow angle (in this case, we choose  as the upper value). If all pairs of consecutive stages are satisfy the condition, we accept this way, otherwise, we reject it and find another way.

1. **Step 3:**

For each found way, try to perform 3 greedy ideas as following:

* Check one trip:
* If the is such route, find list of trip travel on that route: this can be obtained by iterate through the list of route.
* If this trip satisfies the condition, take it as the good solution immediately (because it is easy to deliver and manage a package when it is only in one trip than multiple trips).
* Check multiple trip:
* Check to find some routes connect the consecutive stages of each list of found way list and the package hasn’t been get off the coach at the middle way, this can be obtained by brute-force way as following:
* Cut the list of stages into 2 parts and find route for each part. If all well, save the current solution to list.
* Cut the list of stages into 3 parts and perform action the same as above.
* Cut the list of stages into 4 parts and perform action the same as above.
* For each list of found route, find trip and check condition to find some solution and add them to the list.
* Sort the solution bases on 2 criteria: the duration and the distance. Then choose the first of list mean the best solution for this case and stop the finding method.
* Check middle trips:
* Check to find some trips travel on some routes and the package can be gets on and gets off each trip at the middle of its journey, this can be done by the brute-force way as above that try to cut the list of into 1, 2, 3 or 4 (not consider more) parts and find the route.
* For each found list, we check the time that previous trip will comes to the break station that earlier than the next trip 45 minutes or more. If all are well, save the current list as one of solution in this case.
* Sort the solution bases on 2 criteria: the duration and the distance. Then choose the first of list mean the best solution for this case and stop the finding method.

Notes that the condition including some criterions:

- The total request assigned to trip not more than available volume of trip.

- If the list of trip contains more than one trip, the departure time of each trip is 45 minutes later than the arrival time of previous trip on the list.

### Pros and cons of algorithm

|  |  |
| --- | --- |
| **Pros** | **Cons** |
| Use greedy method for almost steps of algorithm so it is fast to find out the solution and meet the local optimize of the given optimization problem. | The step of algorithm are separately performed so in some case, it cannot meet the absolute optimize solution (it occurs in some tricky or special case). |
| Some list of needed data listed for the first request also can be used for next requests. | Data list all from database so when the system works in a long time, the amount of item can be very large so reducing the time of action. |
| It is easy to see the correctness of this solution and because all condition are check step by step so if a solution is found, it also meet all constraint. | Some steps of algorithm can be implemented better to reduce the complexity but still not have done. |

The complexity of algorithm is  with  are the number of trip, route, stage and requests, respectively.

### References

[1] <http://www.codeproject.com/Questions/164691/Retrieve-all-simple-paths-in-a-graph>

[2] <http://math.stackexchange.com/questions/441182/how-to-check-if-three-coordinates-form-a-line>