Lappeenranta University of Technology
Department of Information Technology
Code Camp on Communications Engineering CT30A9300

Project Final Report

Sale Assistant

Group 05:
Saeed Mirzaeifar
Yasir Ali
ville Arola
Gergely Zachar

Table of Contents

Introduction

Due to the growth of competence in market, businesses are trying to provide better services to their customers. These services can be special offers to a system can bring customers ease of shopping. Therefore, the need of a service that can bring the shopping stuff to the customer location can be a good idea to have an advantage rather than other competent. To provide these kinds of services, businesses should utilize from business application that assist them to attract the satisfaction of customers in the beneficial and also optimized way. Our project for the .NET Code Camp was a multifaceted web application called Sale Assistant.

This report describes the idea that we had developed in .NET Code Camp. The first part indicates the motivation which causes to develop this application. Then, the main technical functionalities and properties of this application are discussed and finally the last part includes the reflection and conclusion.

Motivation

Our project for the .NET Code Camp was a multifaceted web application called Sale Assistant. As the name suggests, the primary purpose and function of the application is to support sales activity by offering a unified interface between stores, delivery companies and consumers and by optimizing all activities therein. The direct benefits of our application for each party can be summarized as follows:

- Consumers can buy goods from various stores through a single web interface and have them delivered to their doorstep.
- Stores are able to reach potential customers who are otherwise hesitant if not unable to visit them.

- Stores which do not have their own online shopping portals can utilize this interface for that purpose.
- Delivery companies can increase the demand of their services by delivering orders which have been placed through the application.

There can also be indirect benefits for realizing our application. Since the idea implies a centralized and optimized management of (everyday) shopping and delivery for multiple consumers, it might save resources such as fuel and also help decrease traffic, pollution, etc. On the other hand, for many consumers the obvious benefit would simply be saving time for other activities than shopping.

Features and Functionalities

The purpose of this system is as follows:

New Business Opportunity:

We have analyzed in Finland there are many supper market but most of them don't provide order delivery service. We have also design this system in such a way if any third party company wants to start its business as order delivery service, it can start using our service.

Optimization:

We have a special route optimizations feature for delivery companies. This system provides optimize result by using different optimization algorithms. This system is very helpful to plan trips, and delivery routes. We consider time, distance, and fuel consumption while drawing optimize route.

Customization:

We have adopted modular approach to make higher cohesive module. Our idea was if some one wants to implement or customize our solution he can easily do it. We

have exposed services if some one wants to develop mobile solution he can use our services.

Efficiency:

This system provides efficient addition, deletion, updating or search for various types of data which are a part of the system. This provide the easy way to create order, get optimize route and secure payment facilities. This decrease the process time for employees or customers to search product that could take hours can be done in just a few seconds or minutes.

Accurate Records:

This system is secured; there is no fear of manipulation of records by unauthorized personnel. All record stored in cloud which reduces these risks.

Accountability:

We make sure that any transaction made through the system is kept track of and in case anything goes wrong with customer dealings, the fault can be traced back to the origin very easily.

Generating Reports:

This is capable of producing reports for system users.

High points of your technical design

Easy To Use:

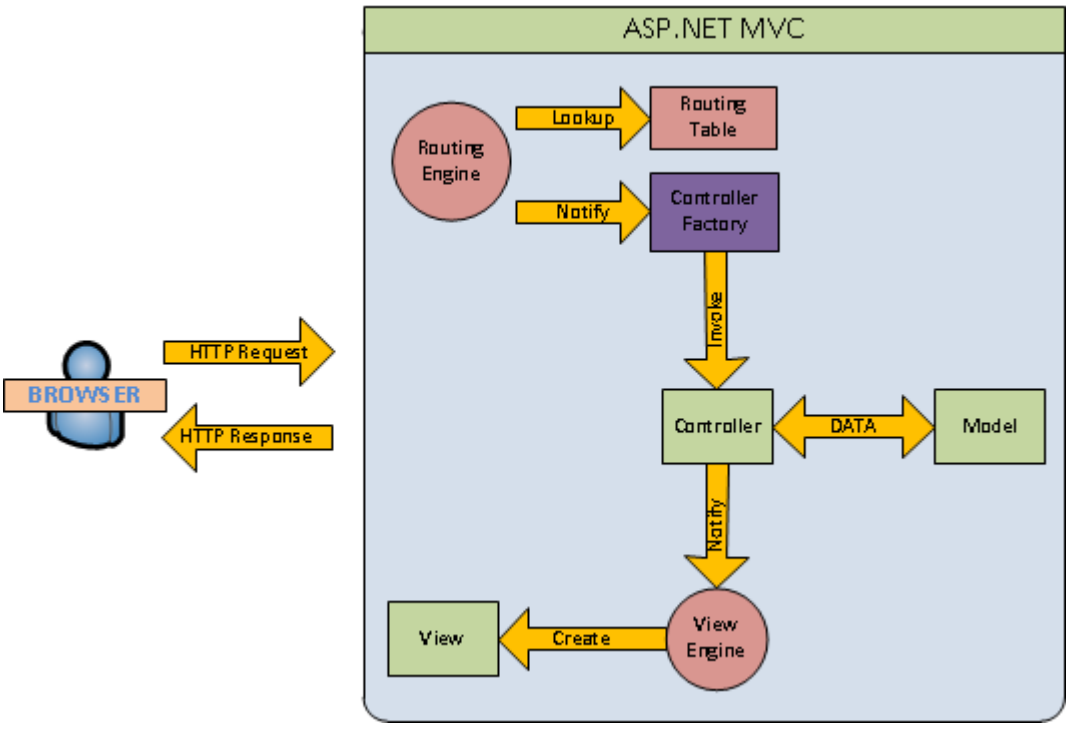
We have designed very simple user interface. We have visited many shopping portal they all using shopping cart feature but to show cart products they have separate design. We provided this feature on the same screen where user selects products so he can check products if he wants to remove any product he can easily do it. We have also separate admin and employee portion to keep simple design.

MVC4:

MVC pattern divide software development into three layers. These layers are model, view and controller. Model is responsible operation related to data. View is responsible how components should render and how data should bind with these component on presentation layer. Controller layer deals navigation and logical operations. It receives event from view and carried out these operation on data and responded to view. The main advantages of MVC4 are:

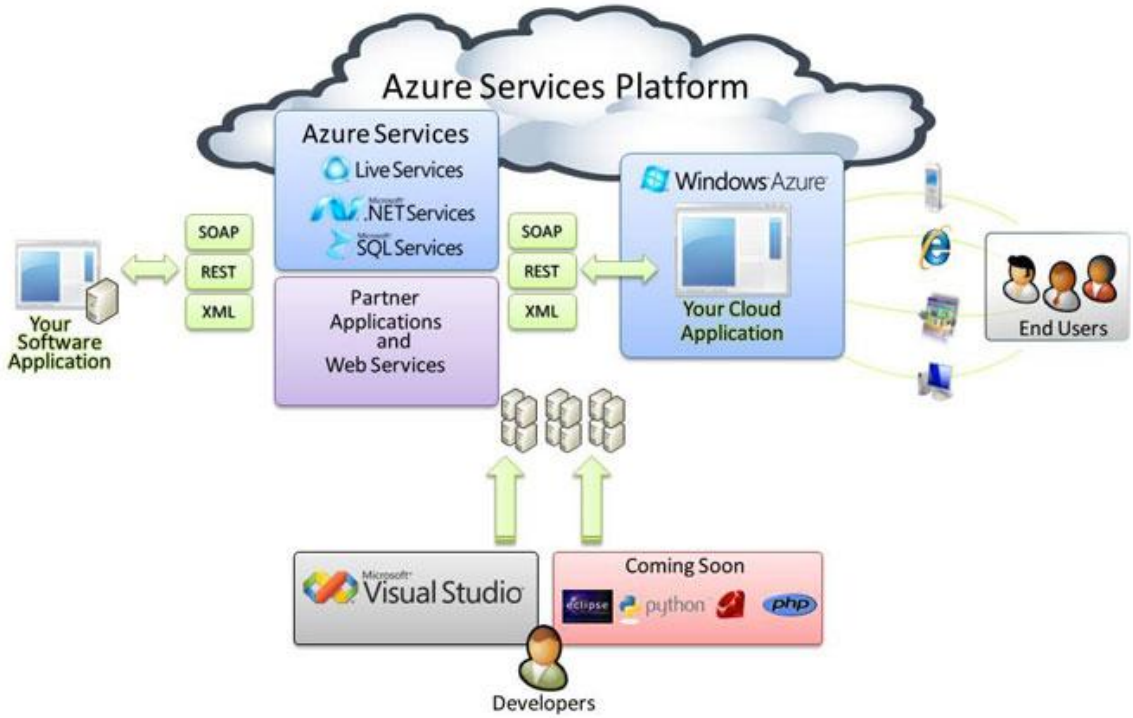
1. Full control over the rendered HTML.
2. Separation of concerns (SoC).
3. Test Driven Development (TDD).
4. Easy integration.
5. Stateless nature of the web design.
6. No View State and Post Back events

MVC4 asp.net application can run on any browsers especially on mobile devices. It mean your can run on multiple platforms.



Azure cloud:

The reason to choose azure cloud is Scalability and flexibility of application. We can create application very easily that run reliably and can scale with out any addition coding effort. Azure Storage provides scalable, secure, performance-efficient storage services in the cloud.

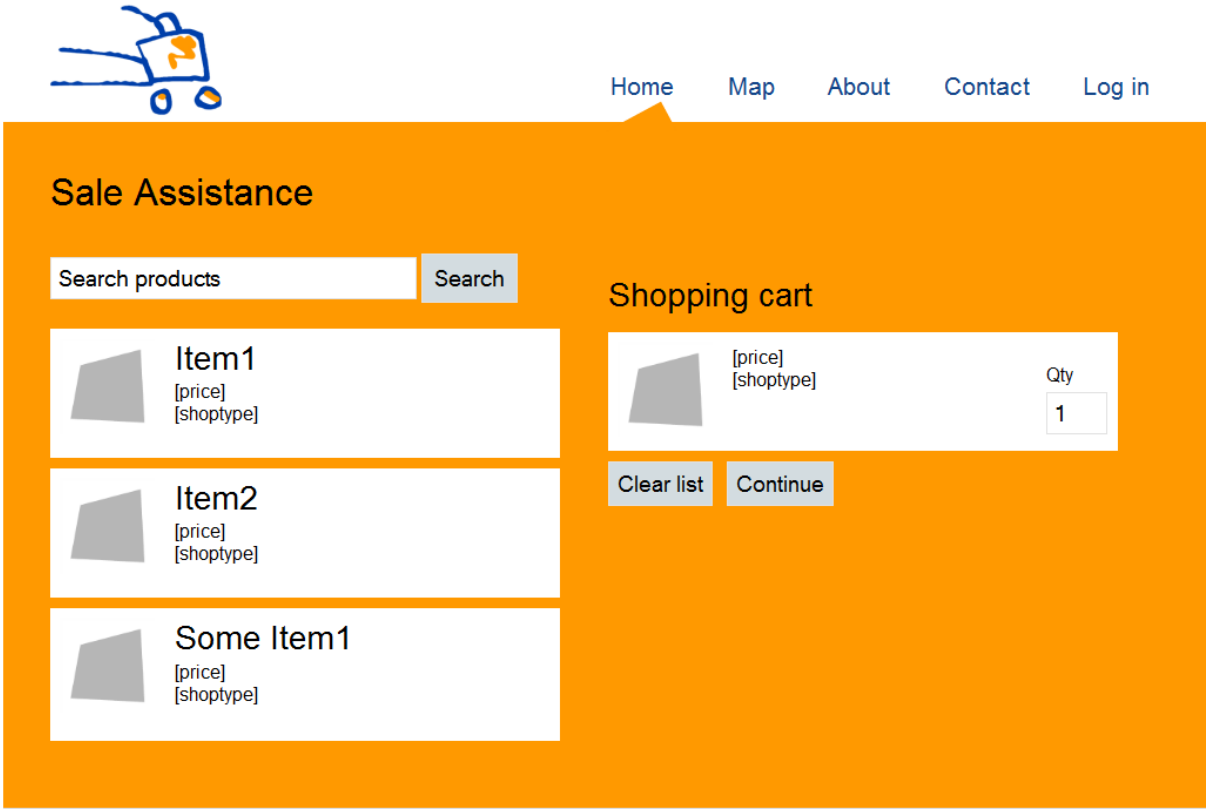


User interface

In this chapter the implemented user interfaces of our application and design principles adopted in creating them are described. Since our application serves three different kinds of users in logically isolated contexts, three different user interfaces were needed. We managed to identify and create some of the views for the three user types (consumer, delivery service and store) but all of them were left in a quite rudimentary state. We also didn't have time to organize and separate them accordingly during the project. Nor did we

manage to implement much of the functionality behind these interfaces, apart from some basic database connections.

The consumer view consists of a product search field, which provides the user a clean and simple interface for finding any product from any of the stores that are registered into the system. Unlike ordinary online shopping portals, we assume that the consumer has adequate knowledge about the product she wants to find and thus, no product listings e.g. by category or by manufacturer are provided in the view. Instead, products matching best with the search string are listed under the search field. From the search results the user can then pick the desired products into a shopping cart, which is in the right-hand side of the view. Having gathered all desired products into the shopping cart, the user can then proceed into another view where the order can be confirmed. The consumer view is shown in the figure below.



For delivery services we devised an interface which shows a list of received orders in a similar fashion as in the consumer view's product listing shown above. The user of the

interface can click on these orders to reveal more information about them, such as list of ordered products and the optimal route for retrieving and delivering them to the customer. The views for showing detailed order information and the route were not implemented during the project.

The third interface for stores required a form for adding new products, a product list and controls for editing and removing existing products. We only managed to create the form during the project, but the other parts could have been adopted with minor changes from the delivery service view.

Visual style

The visual style of the user interfaces was inspired by the currently revered design principles followed e.g. in Windows 8. These include using colours and padding boldly and coherently, focusing on content and avoiding unnecessary decoration with gradients, textures and skeuomorphism. Following these principles also made it much easier and faster to create the design, since all of it could be done with just few CSS style definitions. The only images created for the user interface were the logo of the service and a placeholder image for products

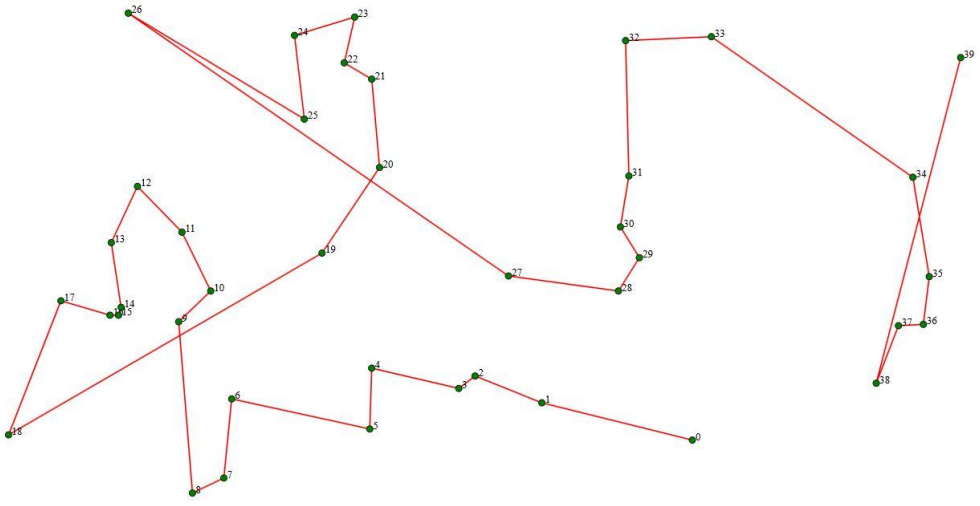
Optimization

The main goal of the program is to provide the optimal route, which means the less time and fuel consuming path between the stores and customers. In the basic problem there are several stores where the salesman can pick up goods and several customers where the salesman can leave the already bought goods. Between all of the customers and stores the exact distance and fuel consumption have to be defined. Then the problem can be described as a complete graph and a mathematical problem.
This kind of problem called as the VRP (vehicle routing problem) which has many variations like with time or fuel constraint or the limitation of the pickup and dropdown order. The most common solutions are based on genetic algorithms, linear programming and branch-and-bound techniques.

In our solutions we simplifies the problem (because the time limitations). In all of the trips first the salesman visit each of the stores and after that delivers the bought items to the customers. In this way we separate the problem to two mathematically equivalent parts. Then if we extract the interested part from the original graph, we get another complete graph where each of the vertices has to be visited only once with the less time and fuel consumption. This problem is called as TSP (traveling salesman problem) which is NP-hard. (Notice that, the complexity of the problem increased, because we simplified it and leaved most of the restrictions from the original VRP problem) Therefor in our solution we can't guarantee the globally optimal solution just local optimality or near optimal solution.

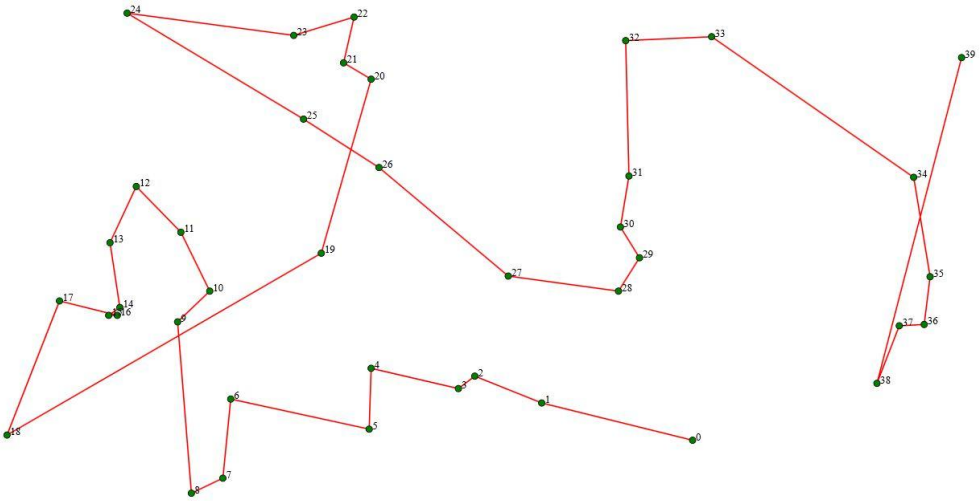
Optimization – Implementation

In our implementation we create a combined algorithm. The first part is a greedy like algorithm and the second part is a genetic algorithm. For testing purposes we firstly implement the algorithms with randomly generated input places and with hand coded SVG generator for visualizing the result. The algorithm calculates the direct Euclidean distance between two points, which can be easily replaced with real distance data later. (This can be done for example with Google Maps distance and fuel calculator.)

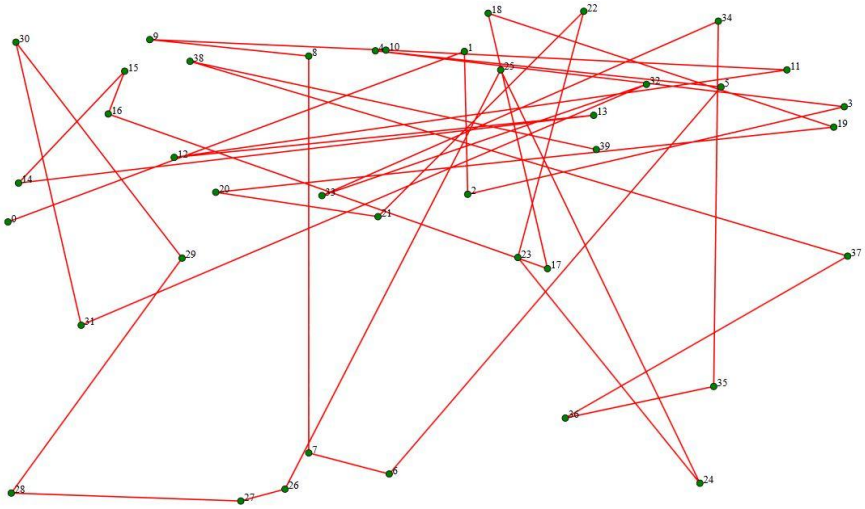


The first greedy like algorithm just picks the next nearest point in the map. The result can be seen on the above figure with 40 vertices.

In the second part of the process we create the initial population for the genetic algorithm. A part of the population consists of the result of the greedy like algorithm, and the rest of the data is randomly generated. This trick can helps to orientate the genetic algorithm to relatively good solution. The bottom figure shows the result of the genetic algorithm with the previous vertices.

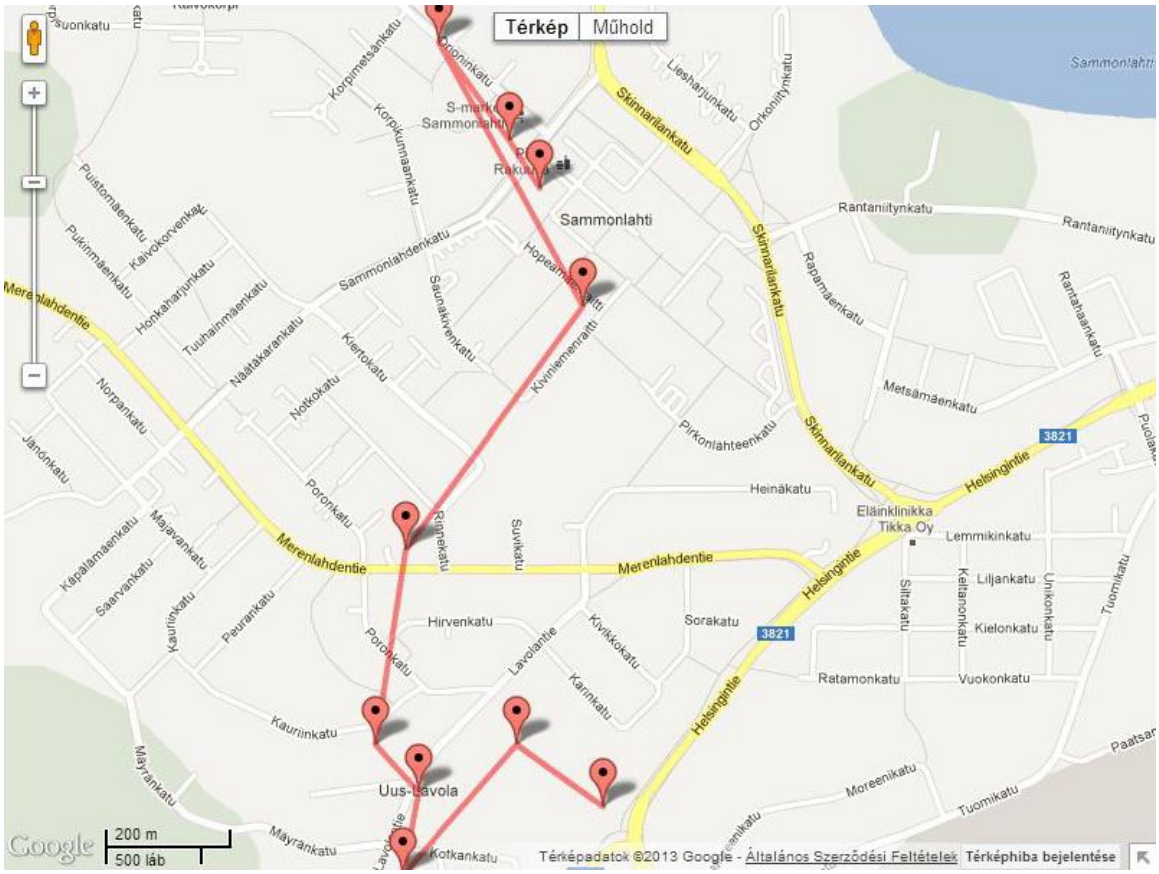


In numbers the distance reduced from 4839 to 4513, about 6,73%. Without the greedy like algorithm the result can lead to local optimum, but sometimes with unacceptable result, like in the under figure.



Notice that the genetic algorithm can be more efficient after properly tuning the parameters.

I our implementation we can also show the result on Google Map without considering the real paths as we mentioned before. This can be seen on the under figure.



Reflection

We had only few ideas in the brainstorming phase. The ideas were discussed briefly and the final decision was made by voting. We ended up selecting Sale Assistant as our project because the other ideas seemed either too vague or unambitious, although choosing Sale Assistant also raised some concerns because of its scale and complexity.

Our development process was constantly disturbed by our insufficient knowledge of the .NET framework and lack of coding experience with C#. Despite the preliminary tutorials, we had to use alot of time getting to know how to work with them. Initially we also lost some time setting up our version control system.

The fast-paced Code Camp method was not the optimal approach for our project (or the other way around), even though team members worked and communicated well together. One week was simply not enough time for designing and implementing this application.

Conclusion

Since we had to jump right into implementation from the initial brainstorming phase in order to get at least some code done within the week, it would be wise to continue the project by taking some steps back.

Things that should be done first:

- Requirements and use cases should be clearly defined and documented so that a clear consensus about the features is established within the development team.
- Other optimization areas which the application should address (in addition to route optimization) and their relationships should be identified and studied.

References

http://www.windowsazure.com/en-us/develop/net/architecture/

http://stackoverflow.com/questions/102558/biggest-advantage-to-using-asp-net-mvc-vs-web-forms