SMART TASK SCHEDULING FOR A TRUCK COMPANY

A Paper
Submitted to the Graduate Faculty
of the
North Dakota State University
of Agriculture and Applied Science

By

Pooja Gautam

In Partial Fulfillment of the Requirements
for the Degree of
MASTER OF SCIENCE

Major Department:
Software Engineering

November 2014

Fargo, North Dakota

North Dakota State University
Graduate School

Title

SMART TASK SCHEDULING FOR A TRUCK COMPANY

By

Pooja Gautam

The Supervisory Committee certifies that this ***disquisition*** complies with North Dakota State University's regulations and meets the accepted standards for the degree of

MASTER OF SCIENCE

SUPERVISORY COMMITTEE:

Dr. Wei Jin
Chair

Dr. Saeed Salem

Dr. Na Gong

Approved:

11/20/14	Dr. Kenneth Magel
Date	Department Chair

ABSTRACT

The truck scheduling system is a web-based application which will facilitate online truck activities. The objective of the application is to provide the truck system with an automated computerized tool, which will be helpful for the associated users to manage their requirements to find the shipper's best routes and alternative routes from the available carriers. The system will be a highly robust and user-friendly tool that aims to achieve maximum user satisfaction.

Text mining has been used to extract the data from email, i.e., an unstructured data source, and to store the data in a structured form, i.e., in RDBMS, and to apply the data-mining technique to discover knowledge from the data.

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my adviser Dr. Wei Jin for her continuous guidance, support, patience, encouragement and motivation. I would also like to thank all my committee members for their support.

I also thank my family and my friends for their continuous support.

DEDICATION

I dedicate this paper to my father, Mr. Ramesh Prasad Gautam, my mother, Mrs. Sabita Gautam; and my husband, Mr. Pranav Raj Sharma, for being the inspiration of my life.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1. INTRODUCTION

Data-mining is the process of analyzing a large amount of data and finding meaningful information that can be understood by people and that can be used to analyze business growth, web analytics, research and development, etc. It is a process of automatically extracting knowledge from a large amount of data (stored data, warehouse data, or information repositories) where new patterns can be found using several of data-mining tools [1].

Data-mining is the process of analyzing a large amount of data and finding meaningful information that can be understood by people and that can be used to analyze business growth, web analytics, research and development, etc. It is a process of automatically extracting knowledge from a large amount of data (stored data, warehouse data, or information repositories) where new patterns can be found using several of data-mining tools [1].

Emails are essential for communication and cooperation; they contain a large amount of data sources. An email is the most used service on the internet. Recent research shows that, generally, people send and receive about 133 emails a day and spend 21% of the work time on email management. It is important to manage emails efficiently because we are spending a lot of our work time on emails. Email data could be very noisy because they contain headers, signatures, quotations, program codes, line break, extra spaces, special characters, badly cased words, spelling mistakes, etc. Within text mining, it is a challenge to extract accurate information from the noisy text data [1, 2].

The data can be stored as various types, such as flat files, spreadsheets, database tables, email storage system, etc. The data have to be collected, cleaned, transformed, and stored to make better decision making. Relational databases have structured data (a tabular database) which can mismatch when a program written in an object oriented programming language is trying to access, which is known as “impedance mismatch.”[3] The ADO.NET Entity Framework can reduce impedance-mismatch issues because of the following reasons:

1) The entity model (EDM) defines the entity framework which operates entity SQL on the instances of the entity model.
2) The model supports bidirectional mapping (EDM-Relational), queries, and updates.
3) The model provides object-relational mapping (ORM) functionality.
4) The Entity Framework has Microsoft’s new language integrate query (LINQ) technologies that extend programming to reduce impedance mismatch for the application [3].

Web applications for scheduling truck routes have been built to automate the transportation-decision process. It enables the users to access the system through email and to obtain a response quickly as per the shipper’s requirements and the carrier’s availability. The system only allows registered users to utilize it so that the shared information is private. The application’s purpose is to automate the system so that users spend less time querying for various possibilities. An auto-generated solution gives the best solution and alternative solutions based on the requirements provided by shippers and the availability of carriers.

In this paper, we provide the software-development process for the truck scheduling system and explain the application’s importance as well as how the application could be used to make transportation decisions. The application automates the email data and schedules the truck route based on the shipper’s requirements and the carrier’s availability. We discuss how we did the email mining for the registered user (shippers and carriers). Then, we illustrate how we cleaned the unstructured (emails) data to create structured data by using a regular expression (pattern matching) technique and stored result into the database tables. Furthermore, we discuss how the data-mining technique can be used to find meaningful information about multiple dimensions to give suggested routes (Good routes/alternative routes are the acceptable routes, and the best route is the best solution among the acceptable routes). We also explained how the extracted information is converted into a human-readable format, such as map view and tabular view, using the jQuery and JavaScript libraries.

For a truck dispatch company to save time and costs, it is important to automate emails when the primary means of communication is through emails. Throughout each day, a lot of emails are received by the truck scheduling system. To study the email data, the web application is developed as a graphical user interface (GUI) based solution that automates the emails from different users (shippers or carriers) and extracts the information and store it in the Microsoft SQL server database tables. With the help of the ADO.NET Entity Framework, LINQ queries could be performed to extract the information, providing the best route and the acceptable routes in map view and tabular view, for the customer based on various scenarios. The web application automates the registered users' process of reading emails, it filters the emails as being from the carrier or shipper based on the email subject; then, it reads the email body or attachment extracts the information, and stores the information in the appropriate database tables, and creates queries that give the best and alternative solutions in the tabular and map-view format.

Given that the system requirements are all met, the user can interact with the system using a GUI. The system shall be able to read the email's subject and filter as available (carriers) or request (shippers) or irrelevant emails. For example, the table named “SubjectEmail” contains the synonyms for “request the service”; the system checks to see if the subject of email occurs in the database, and if the subject matches a synonym for “request the service” in the database, then the email would be tagged as “request route.” The system shall be able to read the email’s body for messages sent from registered users. The system is currently able to filter email sent by the following companies: a) Brake Bush, b) Schwan's Global Supply Chain, c) Lodestar Transport Services, and d) Northland Express Transport.

The system shall be able to extract information from the message and be able to change the abbreviations to full descriptions used in the message, storing final data in the database. For example, if the message has “RGN”, the acronym would be converted into its full form, “Removable Goose Neck.” The distance can be found between two coordinates (available and request) to find the best and alternative carriers and to convert the geo coordinates into addresses.

The application shall get and store the list of good routes (the match between available (carrier) routes and requested routes that is queried based on distance, start date, load type, clearance height, mileage, required equipment, etc.) for each requesting shipper. The application computes and stores the best route (the best among the good route options). The user shall be able to register, login, and see the available or requested routes on the truck scheduling system. Hence, the user should be able to see the best and alternative routes for each requested service.

In addition to these user-interface design requirements, the truck scheduling system has several other desired functionalities, such as the administrator user being able to add Excel format abbreviation files to the system and the information from the excel is stored in the database.

1.1. Paper Organization

Chapter 1 introduces the topic and the objective of this paper. Chapter 2 provides some related work and background of the motivation of this project. Chapter 3 describes the details of system's design and implementation. Chapter 4 presents the system's evaluation, and is followed by the conclusion and future work.

2. RELATED WORK

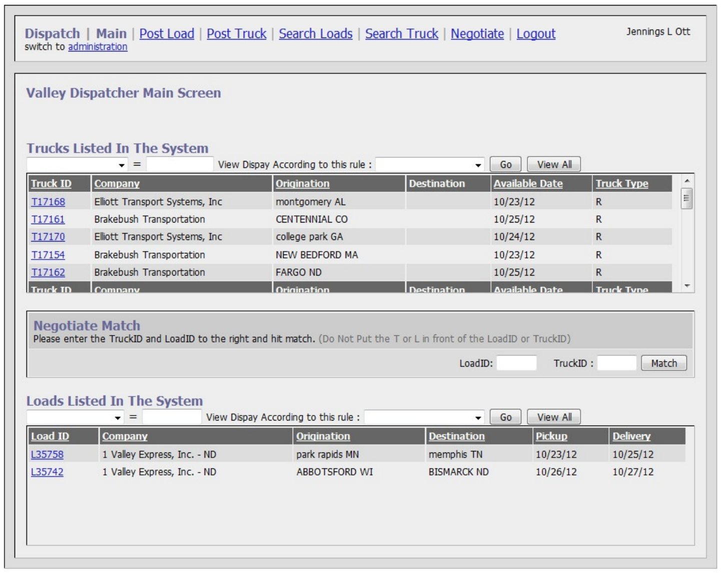
The truck scheduling system is a web-based application which will facilitate truck scheduling activities. The system's objective is to provide automate, computerized tool which is helpful for the users who manage the shipper's requirements with carrier's availability. The system allows the users to be able to keep track of the truck routes and to give suggestions for the request for availability for tracking the nearest possible route. The application gives the best route and alternative routes. The related works for developing the application are discussed in this chapter.

2.1. Valley Logistics Member Services

Valley Logistics is a freight and equipment management company which offers transportation and logistics services. It works as a bridge between shippers and carriers which are, in fact, a highly fragmented industry. The company claims to have solved challenges in the trucking industry by bridging the gap between carriers and shippers [4]. Some of the features they provide are as follows:

- Main Screen (Figure 1) shows the list of trucks in the system and adding rules. Each truck will have a Truck ID, company name, start address, truck type, etc. According to the business rule the list gets modified as per the requirement.
- Add Carrier Information (Figure 2) shows how carriers and shippers input their information into the system which gets saved in the database
- View load (Figure 3) that was added to the system.

- Manually assign carrier to a shipper or vice versa (Figure 4) shows how a carrier is assigned to the shipper.



Dispatch | Main | Post Load | Post Truck | Search Loads | Search Truck | Negotiate | Logout
switch to administration
Jennings L Ott

Valley Dispatcher Main Screen

Trucks Listed In The System

= View Dispay According to this rule :
Go
View All

Truck ID	Company	Origination	Destination	Available Date	Truck Type
T17168	Elliott Transport Systems, Inc	montgomery AL		10/23/12	R
T17161	Brakebush Transportation	CENTENNIAL CO		10/25/12	R
T17170	Elliott Transport Systems, Inc	college park GA		10/24/12	R
T17154	Brakebush Transportation	NEW BEDFORD MA		10/23/12	R
T17162	Brakebush Transportation	FARGO ND		10/25/12	R
Truck ID	Company	Origination	Destination	Available Date	Truck Type

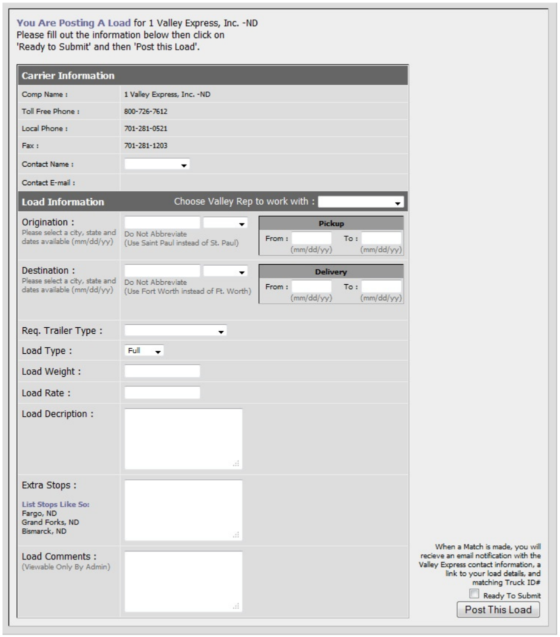
Negotiate Match

Please enter the TruckID and LoadID to the right and hit match. (Do Not Put the T or L in front of the LoadID or TruckID)

Loads Listed In The System

Load ID	Company	Origination	Destination	Pickup	Delivery
L35758	1 Valley Express, Inc. - ND	park rapids MN	memphis TN	10/23/12	10/25/12
L35742	1 Valley Express, Inc. - ND	ABBOTSFORD WI	BISMARCK ND	10/26/12	10/27/12

Figure 1. Truck dispatcher main screen



You Are Posting A Load for 1 Valley Express, Inc. -ND
Please fill out the information below then click on
'Ready to Submit' and then 'Post this Load'.

Carrier Information	
Comp Name :	1 Valley Express, Inc. -ND
Toll Free Phone :	800-726-7612
Local Phone :	701-281-0521
Fax :	701-281-1203
Contact Name :	
Contact E-mail :	

Load Information — Choose Valley Rep to work with :

Origination : Please select a city, state and dates available (mm/dd/yy) — Do Not Abbreviate (Use Saint Paul instead of St. Paul) — Pickup From : (mm/dd/yy) To : (mm/dd/yy)

Destination : Please select a city, state and dates available (mm/dd/yy) — Do Not Abbreviate (Use Fort Worth instead of Ft. Worth) — Delivery From : (mm/dd/yy) To : (mm/dd/yy)

Req. Trailer Type :

Load Type : Full

Load Weight :

Load Rate :

Load Decription :

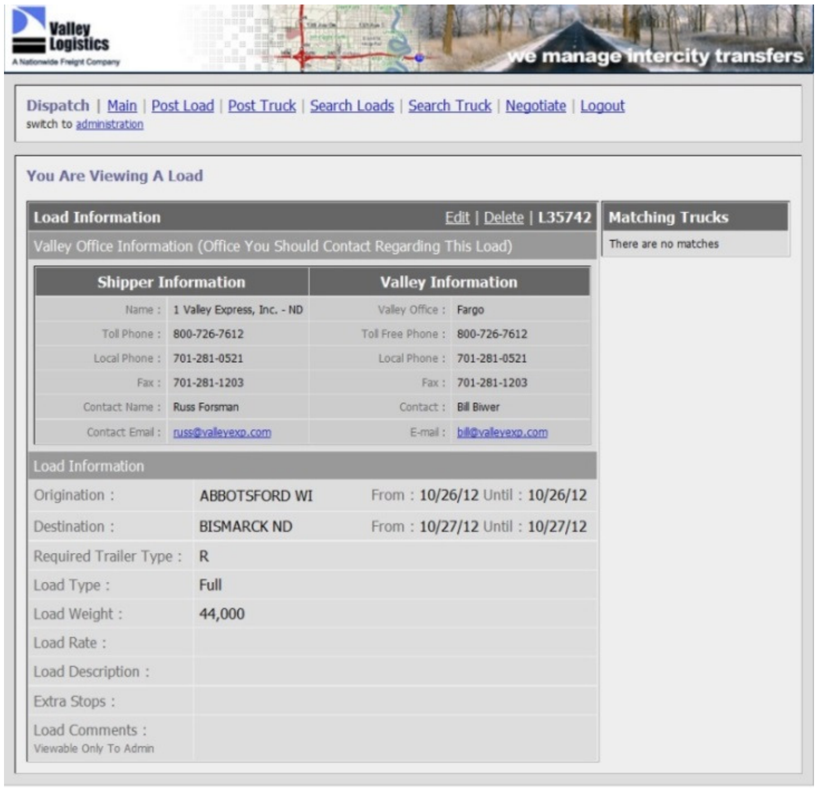
Extra Stops :
List Stops Like So:
Fargo, ND
Grand Forks, ND
Bismarck, ND

Load Comments :
(Viewable Only By Admin)

When a Match is made, you will recieve an email notification with the Valley Express contact information, a link to your load details, and matching Truck ID#

☐ Ready To Submit

Figure 2. Add carrier information



Dispatch | Main | Post Load | Post Truck | Search Loads | Search Truck | Negotiate | Logout
switch to administration

You Are Viewing A Load

Load Information — Edit | Delete | L35742

Valley Office Information (Office You Should Contact Regarding This Load)

Shipper Information		Valley Information	
Name :	1 Valley Express, Inc. - ND	Valley Office :	Fargo
Toll Phone :	800-726-7612	Toll Free Phone :	800-726-7612
Local Phone :	701-281-0521	Local Phone :	701-281-0521
Fax :	701-281-1203	Fax :	701-281-1203
Contact Name :	Russ Forsman	Contact :	Bill Biwer
Contact Email :	russ@valleyexp.com	E-mail :	bill@valleyexp.com

Load Information

Origination :	ABBOTSFORD WI	From : 10/26/12 Until : 10/26/12
Destination :	BISMARCK ND	From : 10/27/12 Until : 10/27/12
Required Trailer Type :	R	
Load Type :	Full	
Load Weight :	44,000	
Load Rate :		
Load Description :		
Extra Stops :		
Load Comments : Viewable Only To Admin		

Matching Trucks

There are no matches

Figure 3. View each load

Valley Logistics works as a bridge between shippers and carriers where each shipper is manually assigned to carrier. They do not have a system to automate request sent by a shipper to automatically assign it to the available carrier. A lack of those features for the Valley Logistics system causes their web application not to be very functional. This is the motivation of development of our truck scheduling system that automatically finds the best possible route for the shipper's request with respect to the available carrier's information, as well as giving sub-optimal alternative routes. We expect the application would be helpful for the end user as the best route and alternative routes are given to the end user by the system. Table. 1 shows the difference between Valley Logistics' System and our developed truck scheduling system.

Table 1. Differences between Valley Logistics and the Truck scheduling system

Valley Logistics	Truck Scheduling System
• Does not automate email • User has to manually input the data in the database tables. • User has to manually input queries according to business logic. • Does not give the best route and the alternative routes. • Output the results in tabular format.	• Email data are parsed and stored in the database. • Automatic extraction of email data and import to the database tables. • System auto-generates the best and alternative routes. • Gives the best route and the alternative routes. • Output the result in map view and tabular format.

2.2. Click Software

Click software is a flexible scheduling system for any small-to-big business to make advanced decisions. It has mobile and chatting service, creating schedules based on various criteria. It works online and offline. The software can be used for various scheduling industries, such as oil and gas, retail, insurance, etc. Scheduling information is provided by the users, and based on the information provided, the system give the solution and performance [5].

Table 2. Differences between Click software and the Truck scheduling system

Click Software	Truck Scheduling System
• Does not automate email. • Manually populated manually database. • Can work for any user provided scheduling request. • Gives the visualization in map view.	• Email data are parsed and stored in the database. • Automatic extraction of email data and import to the database tables. • Specific to truck scheduling. • Gives the visualization in map view.

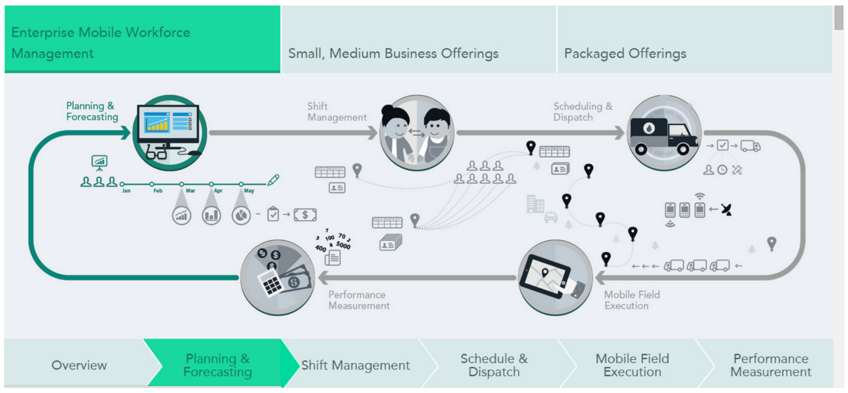


Figure 4. Click software's features

2.3. K Means Clustering Algorithm

The approach for the truck routing system to find the best routes has been influenced by K means clustering. Cluster analysis, or clustering in data-mining, refers to identifying groups or clusters of similar objects. It is non-supervised learning in which we do not know the characteristics that will determine to which clusters the object should belong because there are no labels associated with the objects. We do not have pre-defined classes in clustering [6, 7].

The K means clustering algorithm is a partitioning algorithm that is used to divide the dataset into different parts. The objects are separated into k clusters, where k is a number which is known a priori. We partition the dataset into clusters such that each member of the dataset is assigned to some cluster. There is no item in the dataset which is not assigned to any cluster, and each item is assigned to exactly one cluster. The concept of centroid, or the center of gravity, is introduced which the arithmetic mean of the objects in that cluster. Each point is assigned to the closest cluster based on the defined distance measure. To measure the distance between the object and the centroid, we can use the Euclidean Distance Formula or any other distance formula [6, 7].

- Centroid: The centroid, or the center of gravity, is the arithmetic mean of the objects in that cluster.
- Each point in the cluster is closer to the centroid of that cluster than to the centroids of the other clusters.

2.3.1. Algorithm

Input

We need a dataset that needs to be partitioned, D, containing n objects and the number of clusters to be formed, k, as input to the algorithm

Output

Output would be k clusters satisfying the partitioning objective (similar to objects within the cluster and dissimilar to objects from other clusters).

Method

The “centroid” is the variable that first stores the seed value along with the mean value for the objects within the cluster in the following steps:

(1) Arbitrarily choose k objects from D as the initial cluster centers or seeds;

(2) Centroid = seed;

(3) For each of the k clusters

a) (Re) allocate each object to the cluster to which the object is the most similar, based on the centroid value in the cluster;

b) Update the cluster centroid, i.e., compute the mean value of the objects and assign that value to the centroid.

(4) If there is change in the centroid value, Go to step (3); [6, 7]

2.3.2. Finding the Best-Route Algorithm for Shippers

To find the best route, we have followed some ideas motivated from the K-means algorithm. Because the raw data had multiple dimensions, Euclidian distance was calculated.

On an n-dimensional Euclidian space, let vectors $\boldsymbol{x} = (x_1, x_2, ..., x_n)$ and $\mathbf{y} = (y_1, y_2, ..., y_j)$ The formula to find multi-dimensional distance is as follows:

$$d(x,y) or\ d(y,x) = \sqrt{(x1-y1)^2 + (x2-y2)^2 \dots \dots + (xn-yn)^2}$$

<u>Input</u>

We need carrier's dataset (x) that contains n objects and, k clusters which is equal to number of shippers as well as a shipper's dataset (y) with j objects

<u>Output</u>

The output is the shortest distance between dataset (x) and dataset (y)

<u>Method</u>

The following steps are implemented for the dataset (x) and dataset (y)

1) For each of dataset(y)
2) For each of dataset(x)
3) Match Criteria of y with x
4) IF Matched
5) Get distance $d(x,y) or\ d(y,x) = \sqrt{(x1-y1)^2 + (x2-y2)^2 \dots \dots + (xn-yn)^2}$
6) Store the distances for each y
7) For each of the distances of each y
8) Find the Min value

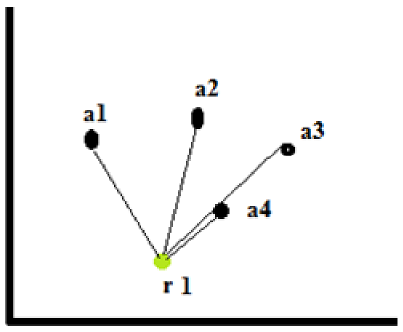


Figure 5. Shortest route

In Figure 4, r1 is the shipper (y), and a1, a2, a3, and a4 are the dataset for the carrier (x). The shortest path is r1-a4, which is the best route.

In the next chapter, truck scheduling system design and development are discussed in detail.

3. SYSTEM DESIGN

According to the Standish Report, one of the major reasons that projects fail is due to a lack of user involvement as well as incomplete requirements and specifications [8]. It is very important that projects are flexible enough for user input and changes during the software lifecycle. We choose an agile methodology because it promotes user input during the different phases of software development and would be flexible with requirement changes.

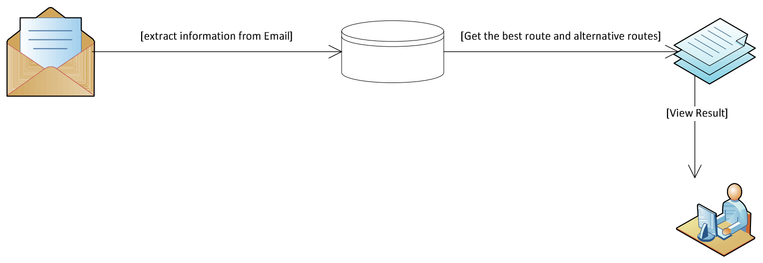

[extract information from Email]
[Get the best route and alternative routes]
[View Result]

Figure 6. Overall picture of the application

3.1. Design

3.1.1. Used Technologies

Microsoft Visual Studio is an integrated development environment (IDE) for C# and related web technologies. The ASP.NET Model View Controllers (MVC) framework is defined and supported in the System.Web.Mvc and System.Web namespace for creating web applications [9].

On MVC where Model is the part of the application implements the domain's data logic. Model objects retrieve information from the database which can be operated and updated. On MVC, View displays the application's user interface (UI) which is populated from the model data. On MVC, Controller handles the business logic and; user interaction, and renders the selected view to display the user interface [9].

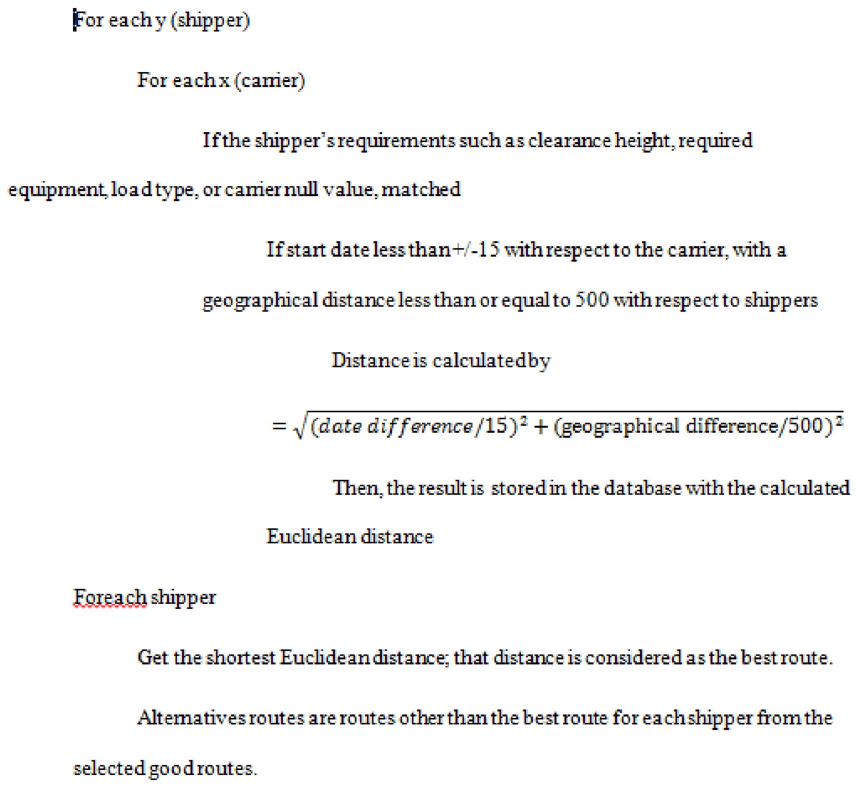
The truck scheduling system was created using Microsoft Visual Studio 2012 and the database used was MS SQL Server 2014. With the help of various C# libraries and web technologies, the application was developed. The application is built on the .NET framework, ASP.NET provides the application programming interface (API) and uses Microsoft's Internet Information Services (IIS) as a platform. The truck scheduling system application runs on several operating systems: Windows, Linux/Unix and Mac OS on these web browsers: Chrome, Firefox, and Internet Explorer.

3.1.2. Designing the Best and Alternative Routes

After the email is read and parsed by the system, it is then stored in the database table named "Email." The matching requirements for each shipper such as difference days (<=15), distance (<=500 miles), mileage, clearance height, required equipment, load type, and etc are found by querying between shippers and carriers. The found matches are called "good routes" and are saved in the database table named "DistanceCalculator." Null values for carriers are considered as adjustable values; for example, the shipper needs a clearance height of 50 feet, and the carrier has not specify any clearance height (null), which means that the carrier can give a truck that meets shipper's height requirements. From these good routes for each shipper, the Euclidian distance between carrier and shipper is calculated. Let carrier has (x_1, x_2, ..., x_n) dimension and shipper have (y_1, y_2, ..., y_n) dimension and between shipper and carrier is as

D (x,y) or d(y,x) = $\sqrt{(x1-y1)^2+(x2-y2)^2 \ldots\ldots + (xn-yn)^2}$.

From the calculated distances, the shortest distance is called the “best route” for each shipper. Other than the best route, other good routes are called “alternative routes” for the shipper. Shortest Euclidian distance between shipper and carriers give the best route for the shipper.



```
For each y (shipper)
    For each x (carrier)
        If the shipper's requirements such as clearance height, required
equipment, load type, or carrier null value, matched
            If start date less than +/-15 with respect to the carrier, with a
        geographical distance less than or equal to 500 with respect to shippers
                Distance is calculated by
            $= \sqrt{(date\ difference/15)^2 + (\text{geographical difference}/500)^2}$
                Then, the result is stored in the database with the calculated
            Euclidean distance
Foreach shipper
    Get the shortest Euclidean distance; that distance is considered as the best route.
    Alternatives routes are routes other than the best route for each shipper from the
selected good routes.
```

Figure 7. Finding the best route

3.1.3. Use Case Diagram

The use-case diagram (Figure 8) shows the interaction between operations and actors in the truck scheduling system’s application. Actors are the system’s user and admin user. Admin users have special privileges such as adding abbreviation to the database table.

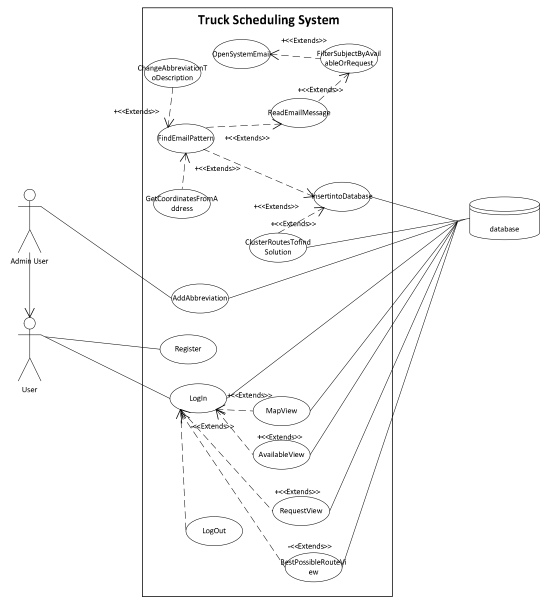


Figure 8. Use case diagram for the application

3.1.4. Activity Diagram

The activity diagram (Figure 9) and (Figure 10) show the activity of filtering emails and storing it in the database and finding the best and alternative routes for the shipper.

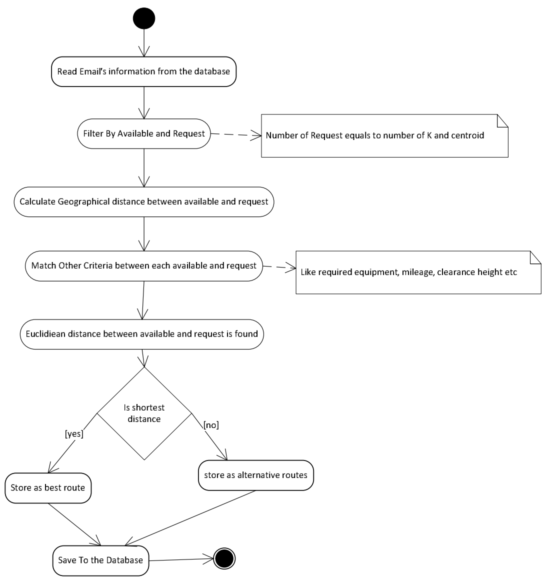

Read Email's information from the database
Filter By Available and Request
Number of Request equals to number of K and centroid
Calculate Geographical distance between available and request
Match Other Criteria between each available and request
Like required equipment, mileage, clearance height etc
Euclidiean distance between available and request is found
Is shortest distance
[yes]
[no]
Store as best route
store as alternative routes
Save To the Database

Figure 9. Activity diagram finding the best routes for each requesting shipper route

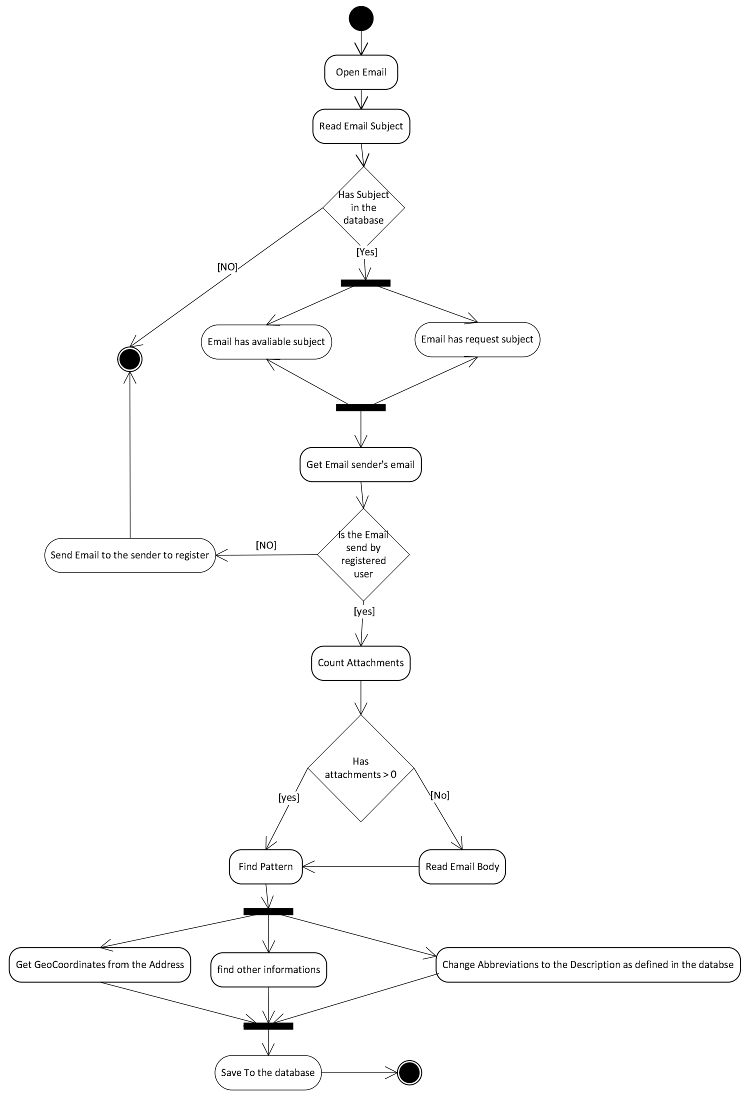

Open Email
Read Email Subject
Has Subject in the database
[NO]
[Yes]
Email has avaliable subject
Email has request subject
Get Email sender's email
Send Email to the sender to register
[NO]
Is the Email send by registered user
[yes]
Count Attachments
Has attachments > 0
[yes]
[No]
Find Pattern
Read Email Body
Get GeoCoordinates from the Address
find other informations
Change Abbreviations to the Description as defined in the databse
Save To the database

Figure 10. Activity diagram for reading and storing email data

3.1.5. Database Design

An MS SQL database was used for persistent data storage with Microsoft Internet Information Services (IIS) as the web server. This web-based application was designed in

ASP.NET; .Net platform was used as a programming environment with C# as the programming language. The tables schema used in the project are as shown in (Figure 11).

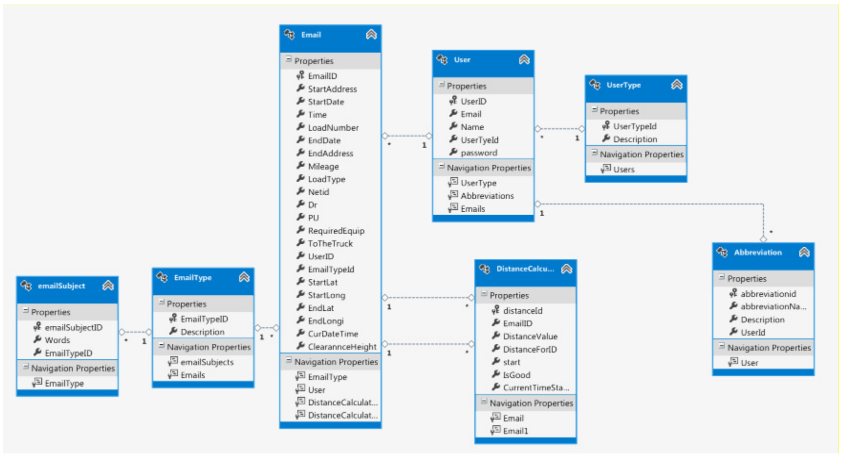


Figure 11. Database design

3.2. Development

This section describes each step of the general workflow of the application using pseudo code, block diagrams and examples. Read Email message describes how the emails is read and stored in the database tables, development of best and alternative routes describe the process of getting the best and the alternatives routes, and finally it also describes how the Excel file is saves in the database table.

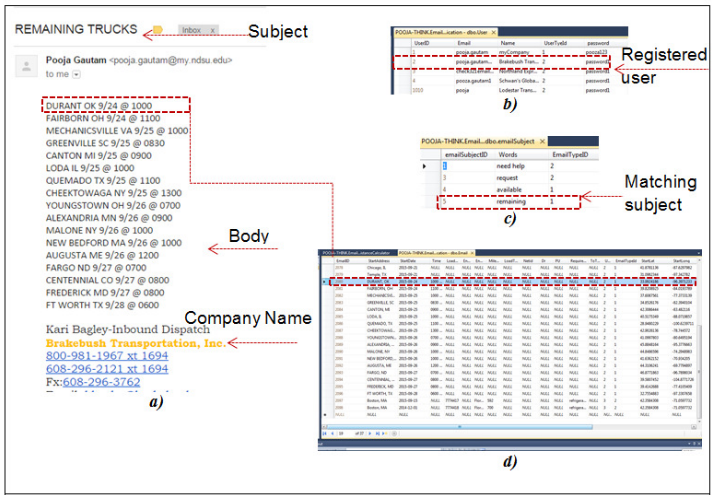
3.2.1. Read Email Message

When the registered user logs in with a valid credential, the system checks for the new email messages, and if there are any new emails, then it checks the messages' subject. If the subject of the email message matches the one in the database table, the system checks to see if the email is being sent by a registered user. If the email is not sent by a registered user, then the system sends a response to the email sender that he/she needs to register before the sending request. If the email is sent by a registered user, the system checks which email pattern the message belongs to and also checks to see if the message has attachment.

```
INPUT: EmailMessages
BEGIN
   FOREACH message in EmailMessage
      SET EmailSubject to be message subject
      DECLARE List of string as  availableSubLine
      DECLARE List of string as RequestSubLine
      SET availableSubLine to be SubjectLine.AvailableSubjects()
      SET RequestSubLine to be SubjectLine.RequestSubjects()
      IF availableSubLine count > 0 or RequestSubLine count > 0
                  IF  Any availableSubLine contains EmailSubject
                       SET requestType to be 1 this is to set
the message as carrier
                  CALL Filteration.FilterEmails with message,
requestType
                    ELSE IF  Any RequestSubLine contains
EmailSubject
                        SET requestType to be 2this is to set
the message as a shipper
                  CALL Filteration.FilterEmails with message,
requestType
               END IF
      END IF
   END LOOP
 END
FilterEmails(message, requestType)
{
   SET string SenderEmail to message.From.Email
   SET User user to context.Users where emailaddress is equal to
SenderEmail
   IF user != null
      SET int messageAttCount to be attachment count
      SET int userID to be user.UserID
      IF messageAttCount > 0
         CALL WithAttachment with msg, userID, and requestType
      ELSE
         CALL WithoutAttachment with msg, userID, and
requestType
      END IF
   ELSE
      CALL sendEmailToUnregisteredUser("You are not registered
to the system")
   END IF
}
```

Figure 12. Pseudo code-from email message to database table

3.2.1.1. Email from the Brakebush Company to Database Table. An email from Brakebush Transportation is shown in Figure 13.a. It's subject is "Remaining trucks" with the key word "remaining" which matches the data in the database table shown in Figure 13.c. with "EmailTypeID" equals 1 which means that the email is from a carrier about truck availability . The company, "Brakebush Transporataiton, Inc.," is a registered user which is shown in Figure 13.b. with "UserType" equals 2 which means that the email is from a general user. The email body, as shown in Figure 13.a. has a repetitive text pattern of "address", "date", and "time" example "DURANT OK 9/24 @ 1000" from the email text, which was send on the date 10/26/2014. Here, the starting address is "Durant OK", the date is "9/24", and the time is 10:00 am. From the starting address, the geo coordinates are generated as a latitude of 33.9924186 and a logitude as -96.3971233. To the extracted information from email is stored in the database table as shown in Figure 13.c.



REMAINING TRUCKS Inbox

Pooja Gautam <pooja.gautam@my.ndsu.edu>
to me

DURANT OK 9/24 @ 1000
FAIRBORN OH 9/24 @ 1100
MECHANICSVILLE VA 9/25 @ 1000
GREENVILLE SC 9/25 @ 0830
CANTON MI 9/25 @ 0900
LODA IL 9/25 @ 1000
QUEMADO TX 9/25 @ 1100
CHEEKTOWAGA NY 9/25 @ 1300
YOUNGSTOWN OH 9/26 @ 0700
ALEXANDRIA MN 9/26 @ 0900
MALONE NY 9/26 @ 1000
NEW BEDFORD MA 9/26 @ 1000
AUGUSTA ME 9/26 @ 1200
FARGO ND 9/27 @ 0700
CENTENNIAL CO 9/27 @ 0800
FREDERICK MD 9/27 @ 0800
FT WORTH TX 9/28 @ 0600

Kari Bagley-Inbound Dispatch
Brakebush Transportation, Inc.
800-981-1967 xt 1694
608-296-2121 xt 1694
Fx:608-296-3762

POOJA-THINK.Email...ication - dbo.User

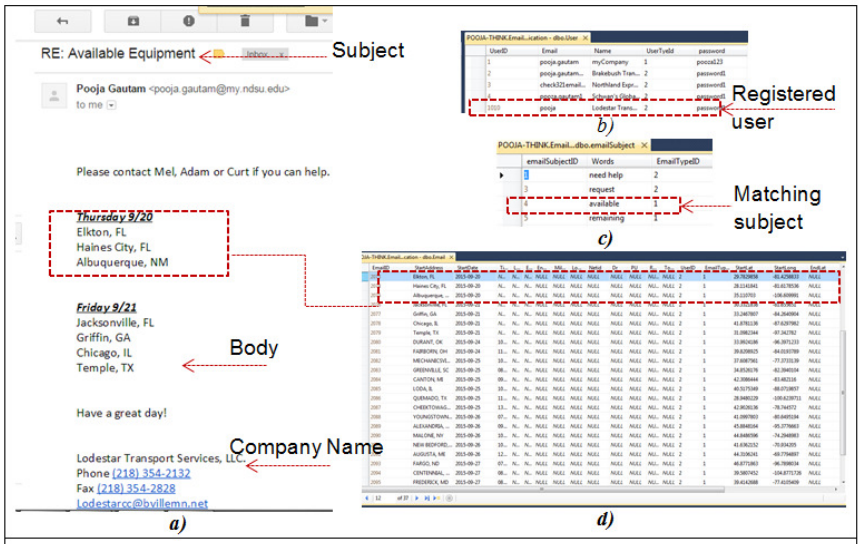
UserID	Email	Name	UserTypeId	password
1	pooja.gautam	myCompany	1	pooza123
2	pooja.gautam...	Brakebush Tran...	2	password1
3	check321email...	Northland Expr...	2	password1
4	pooza.gautam1	Schwan's Globa...	2	password1
1010	pooja	Lodestar Trans...	2	password1

POOJA-THINK.Email...dbo.emailSubject

emailSubjectID	Words	EmailTypeID
1	need help	2
3	request	2
4	available	1
5	remaining	1

Figure 13. From the Brakebush Company's email to the database table

3.2.1.2. Email from Lodestar Transportation Services to the Database Table. Email text from Lodestar Transportation Services is shown Figure 14.a. It has subject as "Available Equipment", with the key word "available" which matches data in the database table shown in Figure 14.c. "EmailTypeID" equals 2 which means that the email is from the carrier about the truck availability. The company "Lodestar Transportation Services" is a registered user which is shown in Figure 14.b. "UserType" equals 2 which means that the email is from a general user. On the email body as show in Figure 14.a. has a repetitive text pattern of "date", and, "addresses" example "Thursday 9/20 and following lines "Elkton, FL", "Haines City, FL" and "Albuquerque, NM" which was sent on the 10/26/2014 as shown in Figure 14.a. Here, the starting address is "Elkton, FL" and the date is "9/20." From the starting address, geo coordinates are generated as a latitude of 29.7829858 and a logitude of -81.4258833. The extracted information from the email was stored in the database table as shown in Figure 14.d.



RE: Available Equipment

Pooja Gautam <pooja.gautam@my.ndsu.edu>
to me

Please contact Mel, Adam or Curt if you can help.

Thursday 9/20
Elkton, FL
Haines City, FL
Albuquerque, NM

Friday 9/21
Jacksonville, FL
Griffin, GA
Chicago, IL
Temple, TX

Have a great day!

Lodestar Transport Services, LLC.
Phone (218) 354-2132
Fax (218) 354-2828
Lodestarcc@bvillemn.net

POOJA-THINK.Email...ication - dbo.User

UserID	Email	Name	UserTyeId	password
1	pooja.gautam	myCompany	1	pooca123
2	pooja.gautam...	Brakebush Tran...	2	password1
3	check321email...	Northland Expr...	2	password1
4	pooja.gautam1	Schwan's Globa...	2	password1
1010	pooja	Lodestar Trans...	2	password1

POOJA-THINK.Email...dbo.emailSubject

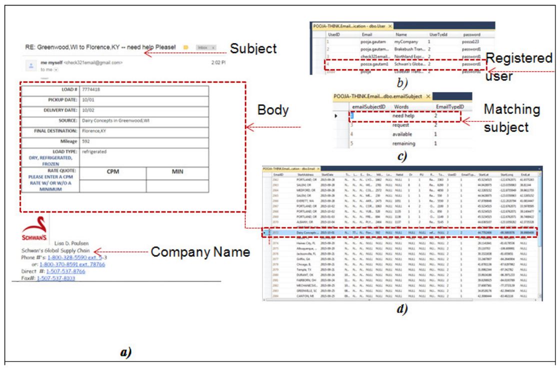
emailSubjectID	Words	EmailTypeID
1	need help	2
3	request	2
4	available	1
5	remaining	1

...-THINK.Email...cation - dbo.Email

EmailID	StartAddress	StartDate	Ti...	L...	E...	En...	Mil...	Lo...	Netid	Or	PU	R...	To...	UserID	EmailTyp...	StartLat	StartLong	EndLat
[illegible]	Elkton, FL	2015-09-20	N...	N...	N...	NULL	NULL	NULL	NULL	NULL	NULL	NU...	NULL	2	1	29.7829858	-81.4258833	NULL
[illegible]	Haines City, FL	2015-09-20	N...	N...	N...	NULL	NULL	NULL	NULL	NULL	NULL	NU...	NULL	2	1	28.1141841	-81.6178536	NULL
[illegible]	Albuquerque, ...	2015-09-20	N...	N...	N...	NULL	NULL	NULL	NULL	NULL	NULL	NU...	NULL	2	1	35.110703	-106.609991	NULL
[illegible]	Jacksonville, FL	2015-09-21	N...	N...	N...	NULL	NULL	NULL	NULL	NULL	NULL	NU...	NULL	2	1	[illegible]	[illegible]	NULL
2077	Griffin, GA	2015-09-21	N...	N...	N...	NULL	NULL	NULL	NULL	NULL	NULL	NU...	NULL	2	1	33.2467807	-84.2640904	NULL
2078	Chicago, IL	2015-09-21	N...	N...	N...	NULL	NULL	NULL	NULL	NULL	NULL	NU...	NULL	2	1	41.8781136	-87.6297982	NULL
2079	Temple, TX	2015-09-21	N...	N...	N...	NULL	NULL	NULL	NULL	NULL	NULL	NU...	NULL	2	1	31.0982344	-97.342782	NULL
2080	DURANT, OK	2015-09-24	10...	N...	N...	NULL	NULL	NULL	NULL	NULL	NULL	NU...	NULL	2	1	33.9924186	-96.3971233	NULL
2081	FAIRBORN, OH	2015-09-24	11...	N...	N...	NULL	NULL	NULL	NULL	NULL	NULL	NU...	NULL	2	1	39.8208925	-84.0193789	NULL
2082	MECHANICSVL...	2015-09-25	10...	N...	N...	NULL	NULL	NULL	NULL	NULL	NULL	NU...	NULL	2	1	37.6087561	-77.3733139	NULL
2083	GREENVILLE, SC	2015-09-25	08...	N...	N...	NULL	NULL	NULL	NULL	NULL	NULL	NU...	NULL	2	1	34.8526176	-82.3940104	NULL
2084	CANTON, MI	2015-09-25	09...	N...	N...	NULL	NULL	NULL	NULL	NULL	NULL	NU...	NULL	2	1	42.3086444	-83.482116	NULL
2085	LODA, IL	2015-09-25	10...	N...	N...	NULL	NULL	NULL	NULL	NULL	NULL	NU...	NULL	2	1	40.5175349	-88.0718657	NULL
2086	QUEMADO, TX	2015-09-25	11...	N...	N...	NULL	NULL	NULL	NULL	NULL	NULL	NU...	NULL	2	1	28.9480229	-100.6239711	NULL
2087	CHEEKTOWAG...	2015-09-25	13...	N...	N...	NULL	NULL	NULL	NULL	NULL	NULL	NU...	NULL	2	1	42.9026136	-78.744572	NULL
2088	YOUNGSTOWN...	2015-09-26	07...	N...	N...	NULL	NULL	NULL	NULL	NULL	NULL	NU...	NULL	2	1	41.0997803	-80.6495194	NULL
2089	ALEXANDRIA, ...	2015-09-26	09...	N...	N...	NULL	NULL	NULL	NULL	NULL	NULL	NU...	NULL	2	1	45.8848164	-95.3776663	NULL
2090	MALONE, NY	2015-09-26	10...	N...	N...	NULL	NULL	NULL	NULL	NULL	NULL	NU...	NULL	2	1	44.8486586	-74.2948083	NULL
2091	NEW BEDFORD...	2015-09-26	10...	N...	N...	NULL	NULL	NULL	NULL	NULL	NULL	NU...	NULL	2	1	41.6362152	-70.934205	NULL
2092	AUGUSTA, ME	2015-09-26	12...	N...	N...	NULL	NULL	NULL	NULL	NULL	NULL	NU...	NULL	2	1	44.3106241	-69.7794897	NULL
2093	FARGO, ND	2015-09-27	07...	N...	N...	NULL	NULL	NULL	NULL	NULL	NULL	NU...	NULL	2	1	46.8771863	-96.7898034	NULL
2094	CENTENNIAL, ...	2015-09-27	08...	N...	N...	NULL	NULL	NULL	NULL	NULL	NULL	NU...	NULL	2	1	39.5807452	-104.8771726	NULL
2095	FREDERICK, MD	2015-09-27	08...	N...	N...	NULL	NULL	NULL	NULL	NULL	NULL	NU...	NULL	2	1	39.4142688	-77.4105409	NULL

Figure 14. From the Loadstar Company's email to database table

3.2.1.3. Email from Schwan's Global Supply Chain to the Database Table. Email text from Schwan's Global Suppy Chain is shown in Figure 15.a. The email has the subject as "Greenwood, WI to Florence, KY -- need help Please!", with the key words "need help" which match the data in the database table shown in Figure 15.c with "EmailTypeID" equals 1 which means that the email is from the shipper and is about the truck request. The company "Schwan's Global Suppy Chain" is a registered user as show in Figure 15.b. "UserType" equals 2 which means the email is from a general user. On the email body, as shown in Figure 15.a has folowing information:"Load #", "start date or pickup date", "delivery date", "source or starting address", "final desitination", "mileage", "load type such as refrigerated", and "rate quote." The "Load #" was "7774418", the "start date or pickup date" was "10/1", the delivery date was "10/2", the "source or starting address"was "Dairy Concepts in Greenwood, WI", the "final desitination" was "Florence, KY", the" mileage" was "592", and "load type" was "refrigerated", which was send on the date 08/26/2014. Here the starting address was "Dairy Concepts in Greenwood, WI" and date is "10/2." From the starting address, geo coordinates are generated a latitude of 44.7702406 and a logitude of -90.5993076. The extracted information from the email was stored the database table as shown in Figure 15.d.



RE: Greenwood,WI to Florence,KY -- need help Please!

me myself <check321email@gmail.com> 2:02 PM
to me

LOAD #	7774418	
PICKUP DATE:	10/01	
DELIVERY DATE:	10/02	
SOURCE:	Dairy Concepts in Greenwood,WI	
FINAL DESTINATION:	Florence,KY	
Mileage	592	
LOAD TYPE: DRY, REFRIGERATED, FROZEN	refrigerated	
RATE QUOTE: PLEASE ENTER A CPM RATE W/ OR W/O A MINIMUM	CPM	MIN

SCHWAN'S
Lisa D. Paulsen
Schwan's Global Supply Chain
Phone #'s: 1-800-328-5590 ext. 5-3
or: 1-800-370-8591 ext. 78766
Direct #: 1-507-537-8766
Fax#: 1-507-537-8103

POOJA-THINK.Email...ication - dbo.User

UserID	Email	Name	UserTypeId	password
1	pooja.gautam	myCompany	1	pooza123
2	pooja.gautam...	Brakebush Tran...	2	password1
3	check321email...	Northland Expr...	2	password1
4	pooza.gautam1	Schwan's Globa...	2	password1
1003	pooja	Codestar Trans...	2	password1

POOJA-THINK.Email...dbo.emailSubject

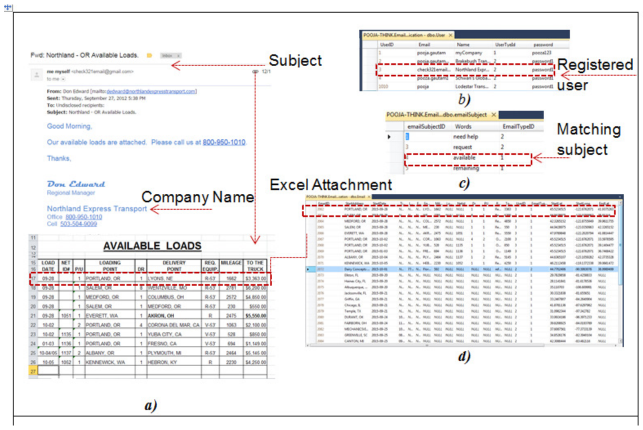
emailSubjectID	Words	EmailTypeID
1	need help	2
3	request	2
4	available	1
5	remaining	1

POOJA-THINK.Email...cation - dbo.Email

Figure 15. From the Schwan's Global's email to the database

3.2.1.4. Email From Northland Express Transport Company to the database table.

Email text from Northland Express Transport is shown in Figure 16.a. The email has the subject as "Northland - OR Available Loads." The key word "available" matches data in the database table shown in Figure 16.c. "EmailTypeID" equals 2 which means the email is from the carrier and is about the available trucks. "Northland Express Transport" is a registered user as show in Figure 16.b. with "UserType" equals 2 which means that the email is from a general user. The email body, as shown in Figure 16.a. has pattern text pattern and has folowing information of "start date or load date," "Net ID," "P/U," "loading point or starting address," "delivery point or final desitination," "mileage,"and "required equipment." For example, the following data are extracted "start date or load date" is "9/28," "Net ID," is null, "P/U" is 1 "loading point or starting address" is "Portland OR," "delivery point or final desitination" is "Lyons, NE," "mileage" is 1662, and "required equipment" is "R-53" that means refrigerated and clerance height as 53 feet. From starting the address geo coordinates are generated as a latitude of 45.5234515 and a logitude of -122.6762071. So the extracted information from email was stored the database table as show in Figure 16.d.



Fwd: Northland - OR Available Loads.

me myself <check321email@gmail.com>
to me

From: Don Edward [mailto:dedward@northlandexpresstransport.com]
Sent: Thursday, September 27, 2012 5:38 PM
To: Undisclosed recipients:
Subject: Northland - OR Available Loads.

Good Morning,

Our available loads are attached. Please call us at 800-950-1010.

Thanks,

Don Edward
Regional Manager

Northland Express Transport
Office 800-950-1010
Cell 503-504-9099

AVAILABLE LOADS

LOAD DATE	NET ID#	P/U	LOADING POINT	DR	DELIVERY POINT	REQ. EQUIP.	MILEAGE	TO THE TRUCK
09-28		1	PORTLAND, OR	1	LYONS, NE	R-53'	1662	$3,363.00
09-28		1	SALEM, OR	8	WENTZVILLE, MO	R-53'	2781	$6,200.00
09-28		1	MEDFORD, OR	1	COLUMBUS, OH	R-53'	2572	$4,850.00
09-28		1	SALEM, OR	1	MEDFORD, OR	R-53'	230	$550.00
09-28	1051	1	EVERETT, WA	1	AKRON, OH	R	2475	$5,550.00
10-02		2	PORTLAND, OR	4	CORONA DEL MAR, CA	V-53'	1063	$2,100.00
10-02	1135	1	PORTLAND, OR	1	YUBA CITY, CA	V-53'	528	$850.00
01-03	1136	1	PORTLAND, OR	1	FRESNO, CA	V-53'	694	$1,149.00
10-04/05	1137	2	ALBANY, OR	1	PLYMOUTH, MI	R-53'	2464	$5,145.00
10-05	1052	1	KENNEWICK, WA	1	HEBRON, KY	R	2230	$4,250.00

POOJA-THINK.Email...ication - dbo.User

UserID	Email	Name	UserTypeId	password
1	pooja.gautam	myCompany	1	pooza123
2	pooja.gautam...	Brakebush Tran...	2	password1
3	check321email...	Northland Expr...	2	password1
4	pooza.gautam1	Schwan's Globa...	2	password1
1000	pooja	Lodestar Trans...	2	password1

POOJA-THINK.Email...dbo.emailSubject

emailSubjectID	Words	EmailTypeID
1	need help	2
3	request	2
4	available	1
5	remaining	1

[illegible]

Figure 16. From the Northland Company's email to the database table

3.2.2. Development of the Best and Alternative Routes

After the email is read and parsed by the system, it is stored in the database table named "Email." Matching requirements for each shipper such as difference days (<=15), distance (<=500 miles), mileage, clearance height, required equipment, load type, and etc are found by querying between shippers and carriers. These matching shippers are the "good" routes for the carrier.

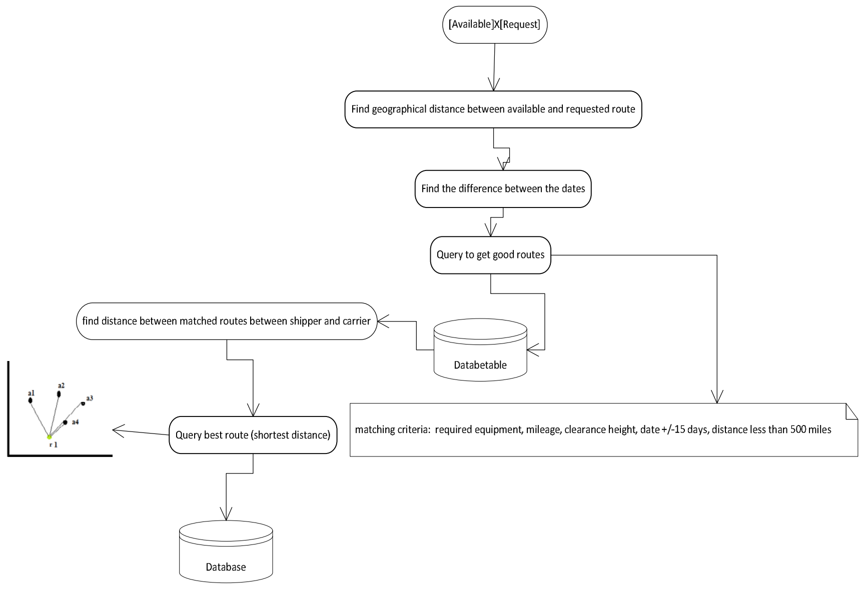

[Available]X[Request]
Find geographical distance between available and requested route
Find the difference between the dates
Query to get good routes
find distance between matched routes between shipper and carrier
Databetable
a1
a2
a3
a4
r 1
Query best route (shortest distance)
matching criteria: required equipment, mileage, clearance height, date +/-15 days, distance less than 500 miles
Database

Figure 17. Finding good and best routes for each requested route

The matches are called "good routes" and are saved in the database table named "DistanceCalculator." From these good routes, the best routes and alternative routes are generated by calculating the shortest distance for carrier and shipper information using the Euclidean distance explained in the Pseudo code, block diagram and example.

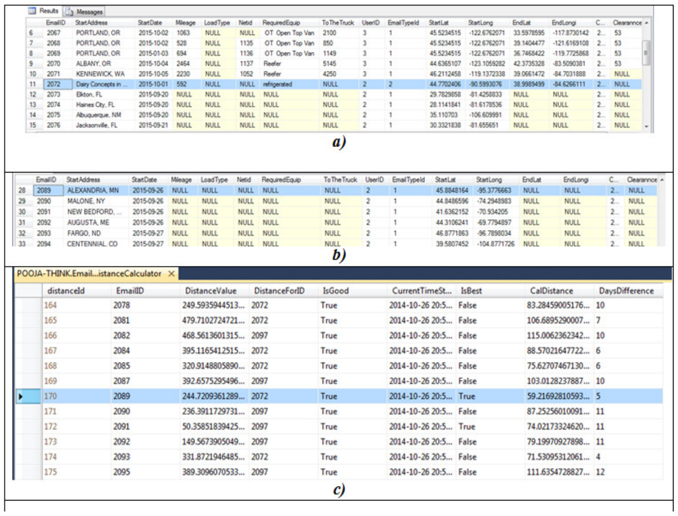
```
// Pseudo code to calculate the distance between the available and requested
routes.
BEGIN
    DECLARE List of type EmailEntity.Email as available
    SET available to be context.Emails.Where EmailTypeId == 1
     DECLARE List of type EmailEntity.Email as request   ==
context.Emails.Where(i => i.EmailTypeId == 2 && i.StartDate >= dt).ToList()
     DECLARE  List of type List of type EmailEntity.DistanceCalculator as
DistanceCal
     FOREACH var x in available
        SET NewDistanceCalculator to be new EmailEntity.DistanceCalculator()
        SET double x1 to 0.0
        SET double y1 to 0.0
          FOREACH var y in request
               SET x1 to be y.StartLat
               SET y1 to be y.StartLong
               SET TimeSpan span to be y.StartDate.Subtract(x.StartDate)
            SET double calDays to be  Math.Abs(span.TotalDays)
             SET double startdis to be CALL Distance(x1, y1, x.StartLat,
x.StartLong)
                SET bool CheckGoodRoute to be getGoodOnces(match criteria)
              if CheckGoodRoute == true
                 Save y.EmailID, x.EmailID,startdis, sqlFormattedDate,calDays
to the database tableDistanceCalculators
           END If
        END LOOP
    END LOOP
    CALL BestResult(maxDistance, daysdifference)
END
// Pseudo code to good routes for each available route. This method gives
all good routes which //match available and request and day difference
within +/- 15 days and geographical distance //less than 500 miles and the
good routes are stored in the database table named //DistanceCalculator.
getGoodOnces(with various parameters)
    Delclare List<Results> as GoodList
       SET DateTime To to be RStartDate.AddDays(daysdifference);
       SET DateTime To be RStartDate.AddDays(-daysdifference);
       SET bool Isgood = false;
    If (matched)
               Isgood = true
     return Isgood
// Pseudo code to get the best route from the list of good routes for each
request route.
BestResult(maxDistance, daysdifference)
    DECLARE List to be Results Mylist
    DECLARE List to be Results newlist
    DECLARE List to be Results Bestlist
    FOREACH var x in GoodList
        SET var y = context.Emails.Where EmailID == x.ForID
        SET TimeSpan span = y.StartDate.Subtract(x.startDate);
        SET double diff = span.TotalDays / diffdays;
        SET double dis = x.Distance / maxDistance;
        SET double cal = Math.Sqrt(Math.Pow(diff, 2) + Math.Pow(dis, 2));
         CALL Mylist.Add(new Results(cal,x.EmailID, x.ForID, x.DistanceID))
    END LOOP
    SET newlist = Mylist.GroupBy EmailID.Select(OrderBy Distance).ToList()
good routes
  SET Bestlist = newlist.GroupBy ForID.Select(OrderBy Distance).ToList()
best routes
    CALL SaveTODatabaseTable with newList and Bestlist
```

Figure 18. Pseudo code- finding the best route

Figure 19.a shows the database table where email informaitons is stored. Here, the "EmailID" 2072 is a shipper that is requesting a truck starting at Dairy Concept in Greenwood, WI on 2015/10/01, mileage more than 592; load type is null; and "netid" is null. The truck needs to be refrigerated as required equipment; the starting latitude is 44.7702406 and starting longitude is -905993076, and the truck does not need any clearance height.

Good matches were found based on requirements such as mileage more than 592 , equipment being refrigerated, days less than or equal to 15, and distance less than or equal to 500 miles from the shippers. Matching routes were called "good routes" and were stored in the database table as shown in Figure 19.c. The geographical distance was within 500 miles, and the day's difference was within 15. Besides day differences and geographical distance, all the other requirements needed an exact match. To find the best route the Euclidian distance was calculated d(x,y) or d(y,x) = $(\sqrt{(\text{date difference}/15)^2 + (\text{geographical difference}/500)^2}) * 100$

The shortest distance was considered as the shipper's best route. For example, "EmailID" 2072 had good routes or matching routes, such as 2078, 2081, 2084, 2085, 2089, and 2093; from all these good routes, the best route available was "EmailID" 2089 as shown in Figure 19.c. the best route has the Euclidean distance of 59.21 (based on days and miles), the shortest distance among the all routes. "EmailID" 2089 had starting address as Alexandria MN, and the start date as 2015/09/26, all other criteria were null. We knew our system assumes that if it does not have criteria then we have considered shipper is flexible to match any criteria as shown in Figure 19.b. "EmailID" 2089 was considered the best route while 2078, 2081, 2084, 2085 and 2093 were considered as alternative routes. The best route and alternative routes is visible on the suggestion page, which can be accessed by the registered users.



Results | Messages

	EmailID	StartAddress	StartDate	Mileage	LoadType	Netid	RequiredEquip	ToTheTruck	UserID	EmailTypeId	StartLat	StartLong	EndLat	EndLongi	C...	Clearannce
6	2067	PORTLAND, OR	2015-10-02	1063	NULL	NULL	OT Open Top Van	2100	3	1	45.5234515	-122.6762071	33.5978595	-117.8730142	2...	53
7	2068	PORTLAND, OR	2015-10-02	528	NULL	1135	OT Open Top Van	850	3	1	45.5234515	-122.6762071	39.1404477	-121.6169108	2...	53
8	2069	PORTLAND, OR	2015-01-03	694	NULL	1136	OT Open Top Van	1149	3	1	45.5234515	-122.6762071	36.7468422	-119.7725868	2...	53
9	2070	ALBANY, OR	2015-10-04	2464	NULL	1137	Reefer	5145	3	1	44.6365107	-123.1059282	42.3735328	-83.5090381	2...	53
10	2071	KENNEWICK, WA	2015-10-05	2230	NULL	1052	Reefer	4250	3	1	46.2112458	-119.1372338	39.0661472	-84.7031888	2...	NULL
11	2072	Dairy Concepts in ...	2015-10-01	592	NULL	NULL	refrigerated	NULL	2	2	44.7702406	-90.5993076	38.9989499	-84.6266111	2...	NULL
12	2073	Elkton, FL	2015-09-20	NULL	NULL	NULL	NULL	NULL	2	1	29.7829858	-81.4258833	NULL	NULL	2...	NULL
13	2074	Haines City, FL	2015-09-20	NULL	NULL	NULL	NULL	NULL	2	1	28.1141841	-81.6178536	NULL	NULL	2...	NULL
14	2075	Albuquerque, NM	2015-09-20	NULL	NULL	NULL	NULL	NULL	2	1	35.110703	-106.609991	NULL	NULL	2...	NULL
15	2076	Jacksonville, FL	2015-09-21	NULL	NULL	NULL	NULL	NULL	2	1	30.3321838	-81.655651	NULL	NULL	2...	NULL

a)

	EmailID	StartAddress	StartDate	Mileage	LoadType	Netid	RequiredEquip	ToTheTruck	UserID	EmailTypeId	StartLat	StartLong	EndLat	EndLongi	C...	Clearannce
28	2089	ALEXANDRIA, MN	2015-09-26	NULL	NULL	NULL	NULL	NULL	2	1	45.8848164	-95.3776663	NULL	NULL	2...	NULL
29	2090	MALONE, NY	2015-09-26	NULL	NULL	NULL	NULL	NULL	2	1	44.8486596	-74.2948983	NULL	NULL	2...	NULL
30	2091	NEW BEDFORD, ...	2015-09-26	NULL	NULL	NULL	NULL	NULL	2	1	41.6362152	-70.934205	NULL	NULL	2...	NULL
31	2092	AUGUSTA, ME	2015-09-26	NULL	NULL	NULL	NULL	NULL	2	1	44.3106241	-69.7794897	NULL	NULL	2...	NULL
32	2093	FARGO, ND	2015-09-27	NULL	NULL	NULL	NULL	NULL	2	1	46.8771863	-96.7898034	NULL	NULL	2...	NULL
33	2094	CENTENNIAL, CO	2015-09-27	NULL	NULL	NULL	NULL	NULL	2	1	39.5807452	-104.8771726	NULL	NULL	2...	NULL

b)

POOJA-THINK.Email...istanceCalculator

distanceId	EmailID	DistanceValue	DistanceForID	IsGood	CurrentTimeSt...	IsBest	CalDistance	DaysDifference
164	2078	249.5935944513...	2072	True	2014-10-26 20:5...	False	83.28459005176...	10
165	2081	479.7102724721...	2072	True	2014-10-26 20:5...	False	106.6895290007...	7
166	2082	468.5613601315...	2097	True	2014-10-26 20:5...	False	115.0062362342...	10
167	2084	395.1165412515...	2072	True	2014-10-26 20:5...	False	88.57021647722...	6
168	2085	320.9148805890...	2072	True	2014-10-26 20:5...	False	75.62707467130...	6
169	2087	392.6575295496...	2097	True	2014-10-26 20:5...	False	103.0128237887...	10
170	2089	244.7209361289...	2072	True	2014-10-26 20:5...	True	59.21692810593...	5
171	2090	236.3911729731...	2097	True	2014-10-26 20:5...	False	87.25256010091...	11
172	2091	50.35851839425...	2097	True	2014-10-26 20:5...	True	74.02173324620...	11
173	2092	149.5673905049...	2097	True	2014-10-26 20:5...	False	79.19970927898...	11
174	2093	331.8721946485...	2072	True	2014-10-26 20:5...	False	71.53095312061...	4
175	2095	389.3096070533...	2097	True	2014-10-26 20:5...	False	111.6354728827...	12

c)

Figure 19. Shortest route calculation

3.2.3. Excel File to the Database Table

The admin user can upload the abbreviation Excel file; which is stored in the database and used while reading the email body, if abbreviations are converted to full description. The admin user upload an Excel file, that is read by the system and the abbreviation and its description is stored in the database table.

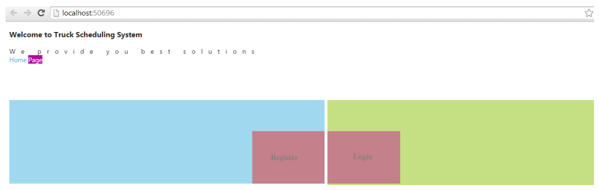
```
//Pseudo code will read the file input(Excel file) from the user and
store it so that it can be //read by the AddAbbreviations (Userid,
path) method
INPUT : Userid,file
BEGIN
   IF (file.ContentLength > 0)
      SET var fileName = Path.GetFileName(file.FileName)
      SET var path =
Path.Combine(Server.MapPath("~/Uploadedfiles"),fileName)
      SET var path1 = "somepath" + file.FileName
      file.SaveAs(path)
      SET string added = AddAbbreviations(Userid, path)

END
```

Figure 20. Pseudo code- from excel file to database table

4. EVALUATION

This chapter demonstrates how the truck scheduling system would handle various criteria, such as login, logout, routes view, etc. When the application starts, the customer either logs in or registers via the system. The layout is shown in Figure 21, Figure 22, and Figure 23.

The user can register with an email, company name and password. After registration the user can log in with an email and password. If the login credentials are correct, then the system directs the user to index page otherwise, the user is redirected to the Login page.



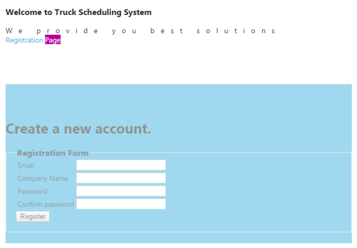
localhost:50696

Welcome to Truck Scheduling System

W e p r o v i d e y o u b e s t s o l u t i o n s

Home Page

Figure 21. Home page



Welcome to Truck Scheduling System

W e p r o v i d e y o u b e s t s o l u t i o n s

Registration Page

Create a new account.

Registration Form

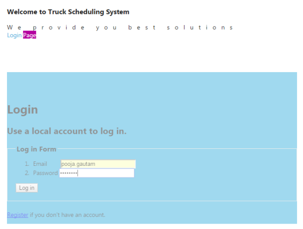
Email

Company Name

Password

Confirm password

Figure 22. Registration page



Welcome to Truck Scheduling System

W e p r o v i d e y o u b e s t s o l u t i o n s

Login Page

Login

Use a local account to log in.

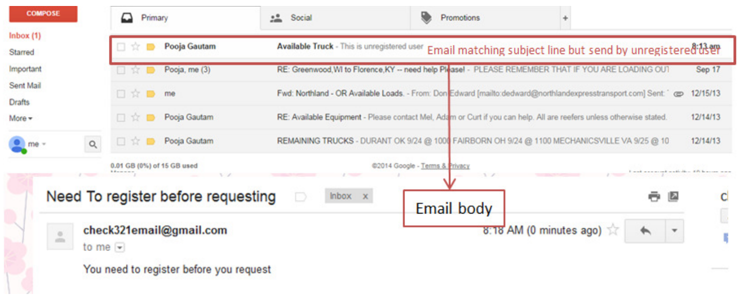
Log in Form

1. Email pooja.gautam
2. Password ••••••••

Register if you don't have an account.

Figure 23. Login page

When the user logs in to the truck scheduling system with valid credentials, the system reads all the new emails sent to the system and checks for the emails' subject lines. If the email subject matches data in the database table named "emailSubject," then the email is categorized as carriers or shippers. If the email is sent by non-registered user and has a matching subject an email response is sent to the user as show in Figure 24.



Pooja Gautam	Available Truck - This is unregistered user		8:13 am
Pooja, me (3)	RE: Greenwood,WI to Florence,KY -- need help Please! - PLEASE REMEMBER THAT IF YOU ARE LOADING OUT		Sep 17
me	Fwd: Northland - OR Available Loads. - From: Don Edward [mailto:dedward@northlandexpresstransport.com] Sent:		12/15/13
Pooja Gautam	RE: Available Equipment - Please contact Mel, Adam or Curt if you can help. All are reefers unless otherwise stated.		12/14/13
Pooja Gautam	REMAINING TRUCKS - DURANT OK 9/24 @ 1000 FAIRBORN OH 9/24 @ 1100 MECHANICSVILLE VA 9/25 @ 10		12/14/13

0.01 GB (0%) of 15 GB used
©2014 Google - Terms & Privacy

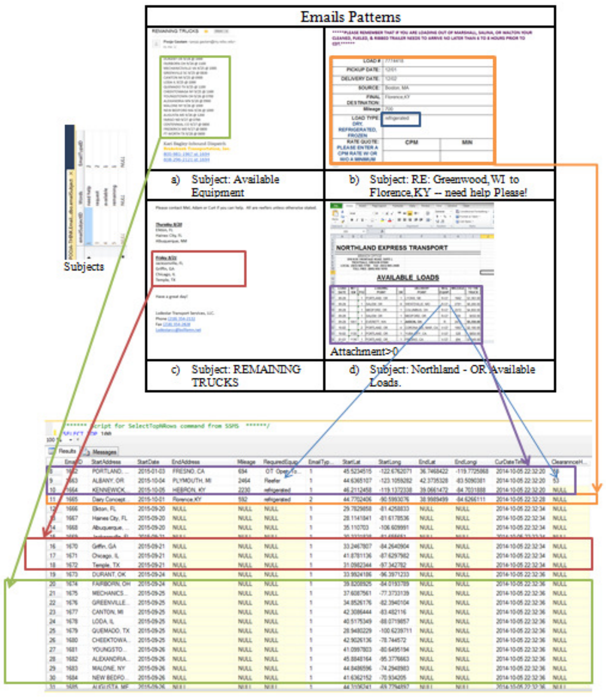
Need To register before requesting

check321email@gmail.com
to me
8:18 AM (0 minutes ago)

You need to register before you request

Figure 24. Email with matching subject line but non-registered user

If any new email is sent by a registered user and has email subject that matches an email subject in the database table named "emailSubject", then the email is categorized into shippers or carriers, after which the email attachments are counted. If the email message contains no attachment then the email body is read by the system; otherwise, the attachment is saved in the location machine, and the attachment is read by the system. From the email text the attachment, the information is gathered and stored it in the database table named "Email" as shown in Figure 25. Figure 25 also shows how an abbreviation is converted to a full description. For example, R-53 is converted to "R as Refrigerated and 53' for clearance height." Based on the shipper's request and the carrier's availability, the best possible routes are listed. From the gathered good routes, Euclidean distance between the shippers and carriers is calculated based on the date and geographical distances.


Emails Patterns
Subjects
a) Subject: Available Equipment
b) Subject: RE: Greenwood,WI to Florence,KY -- need help Please!
Attachment>0
c) Subject: REMAINING TRUCKS
d) Subject: Northland - OR Available Loads.
NORTHLAND EXPRESS TRANSPORT
AVAILABLE LOADS
Results
Messages

```
/****** Script for SelectTopNRows command from SSMS ******/
SELECT TOP 1000
```

	EmailID	StartAddress	StartDate	EndAddress	Mileage	RequiredEquip	EmailTyp...	StartLat	StartLong	EndLat	EndLongi	CurDateTime	ClearanceH...
8	1662	PORTLAND, ...	2015-01-03	FRESNO, CA	694	OT Open To...	1	45.5234515	-122.6762071	36.7468422	-119.7725868	2014-10-05 22:32:20	68
9	1663	ALBANY, OR	2015-10-04	PLYMOUTH, MI	2464	Reefer	1	44.6365107	-123.1059282	42.3735328	-83.5090381	2014-10-05 22:32:20	53
10	1664	KENNEWICK, ...	2015-10-05	HEBRON, KY	2230	refrigerated	1	46.2112458	-119.1372338	39.0661472	-84.7031888	2014-10-05 22:32:20	NULL
11	1665	Dairy Concept...	2015-10-01	Florence,KY	592	refrigerated	2	44.7702406	-90.5993076	38.9989499	-84.6266111	2014-10-05 22:32:28	NULL
12	1666	Elkton, FL	2015-09-20	NULL	NULL	NULL	1	29.7829858	-81.4258833	NULL	NULL	2014-10-05 22:32:34	NULL
13	1667	Haines City, FL	2015-09-20	NULL	NULL	NULL	1	28.1141841	-81.6178536	NULL	NULL	2014-10-05 22:32:34	NULL
14	1668	Albuquerque, ...	2015-09-20	NULL	NULL	NULL	1	35.110703	-106.609991	NULL	NULL	2014-10-05 22:32:34	NULL
15	1669	Jacksonville, FL	2015-09-21	NULL	NULL	NULL	1	[illegible]	[illegible]	NULL	NULL	2014-10-05 22:32:34	NULL
16	1670	Griffin, GA	2015-09-21	NULL	NULL	NULL	1	33.2467807	-84.2640904	NULL	NULL	2014-10-05 22:32:34	NULL
17	1671	Chicago, IL	2015-09-21	NULL	NULL	NULL	1	41.8781136	-87.6297982	NULL	NULL	2014-10-05 22:32:34	NULL
18	1672	Temple, TX	2015-09-21	NULL	NULL	NULL	1	31.0982344	-97.342782	NULL	NULL	2014-10-05 22:32:34	NULL
19	1673	DURANT, OK	2015-09-24	NULL	NULL	NULL	1	33.9924186	-96.3971233	NULL	NULL	2014-10-05 22:32:36	NULL
20	1674	FAIRBORN, OH	2015-09-24	NULL	NULL	NULL	1	39.8208925	-84.0193789	NULL	NULL	2014-10-05 22:32:36	NULL
21	1675	MECHANICS...	2015-09-25	NULL	NULL	NULL	1	37.6087561	-77.3733139	NULL	NULL	2014-10-05 22:32:36	NULL
22	1676	GREENVILLE...	2015-09-25	NULL	NULL	NULL	1	34.8526176	-82.3940104	NULL	NULL	2014-10-05 22:32:36	NULL
23	1677	CANTON, MI	2015-09-25	NULL	NULL	NULL	1	42.3086444	-83.482116	NULL	NULL	2014-10-05 22:32:36	NULL
24	1678	LODA, IL	2015-09-25	NULL	NULL	NULL	1	40.5175349	-88.0719857	NULL	NULL	2014-10-05 22:32:36	NULL
25	1679	QUEMADO, TX	2015-09-25	NULL	NULL	NULL	1	28.9480229	-100.6239711	NULL	NULL	2014-10-05 22:32:36	NULL
26	1680	CHEEKTOWA...	2015-09-25	NULL	NULL	NULL	1	42.9026136	-78.744572	NULL	NULL	2014-10-05 22:32:36	NULL
27	1681	YOUNGSTO...	2015-09-26	NULL	NULL	NULL	1	41.0997803	-80.6495194	NULL	NULL	2014-10-05 22:32:36	NULL
28	1682	ALEXANDRIA...	2015-09-26	NULL	NULL	NULL	1	45.8848164	-95.3776663	NULL	NULL	2014-10-05 22:32:36	NULL
29	1683	MALONE, NY	2015-09-26	NULL	NULL	NULL	1	44.8486596	-74.2948983	NULL	NULL	2014-10-05 22:32:36	NULL
30	1684	NEW BEDFO...	2015-09-26	NULL	NULL	NULL	1	41.6362152	-70.934205	NULL	NULL	2014-10-05 22:32:36	NULL
31	1685	AUGUSTA, ME	2015-09-26	NULL	NULL	NULL	1	44.3106241	-69.7794897	NULL	NULL	2014-10-05 22:32:36	NULL

Figure 25. From the emails to MS SQL tables: “Email” and “EmailSubject”

Then, the user is directed to the Index page. If the user is the admin user, the view is as shown in Figure 26 if the user is a general user, the view is as shown in Figure 27. Both Index pages have map view, request route , available route and logout buttons. The admin user has an admin button to view and modify the user and abbreviation lists.

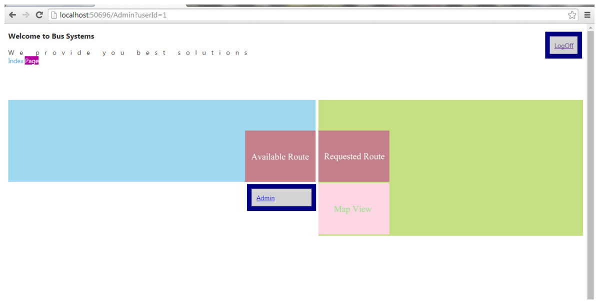


Figure 26. Admin index page

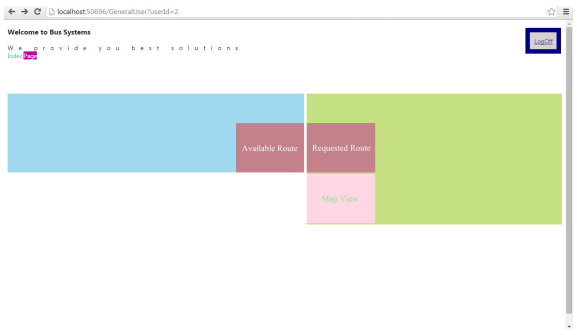
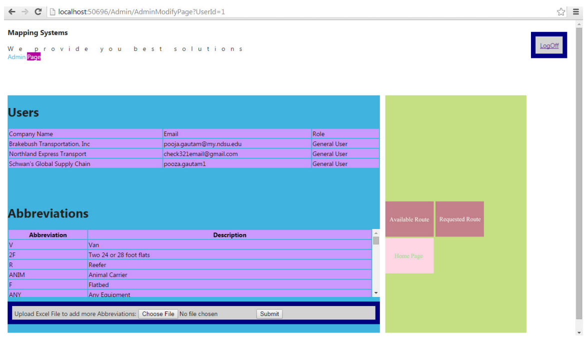


Figure 27. General user index page

The admin user can view and modify the userlist. The user can also view and upload abbreviations and descriptions which should be in an Excel format. The uploaded file is stored in the database table named "Abbreviation" using entity framework and C# programming. The lists are ordered by default as in the database table as shown in Figure 28.


localhost:50696/Admin/AdminModifyPage?UserId=1
Mapping Systems
We provide you best solutions
Admin Page
LogOff
Available Route
Requested Route
Home Page
Upload Excel File to add more Abbreviations:
Choose File
No file chosen
Submit

Users

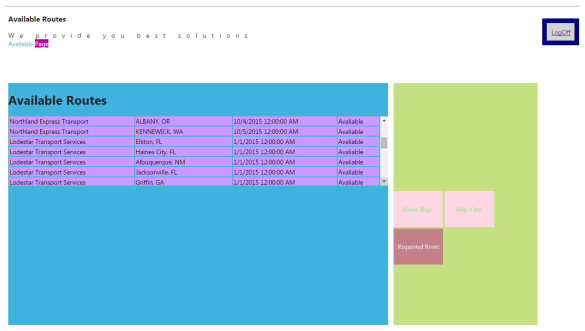
Company Name	Email	Role
Brakebush Transportation, Inc	pooja.gautam@my.ndsu.edu	General User
Northland Express Transport	check321email@gmail.com	General User
Schwan's Global Supply Chain	pooza.gautam1	General User

Abbreviations

Abbreviation	Description
V	Van
2F	Two 24 or 28 foot flats
R	Reefer
ANIM	Animal Carrier
F	Flatbed
ANY	Any Equipment

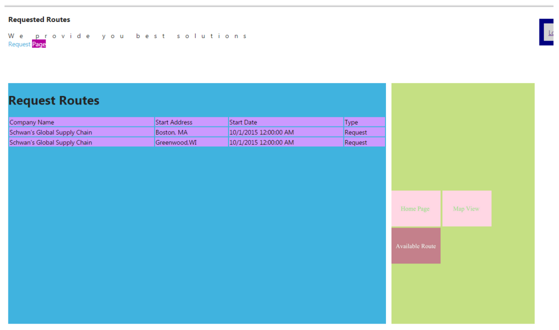
Figure 28. Admin page

When the request route is clicked, all the shipper's information is listed in tabular format. The list is obtained from the database table named "Email" which has the attribute "EmailtypeId" equals 2 as shown in Figure 30. Similary, when available route is clicked, all the carriers' information is listed in tabular format. The list is obtained from the database table "Email" which has the attribute Emailtype equals 1 as shown in Figure 30.



Northland Express Transport	ALBANY, OR	10/4/2015 12:00:00 AM	Avaliable
Northland Express Transport	KENNEWICK, WA	10/5/2015 12:00:00 AM	Avaliable
Lodestar Transport Services	Elkton, FL	1/1/2015 12:00:00 AM	Avaliable
Lodestar Transport Services	Haines City, FL	1/1/2015 12:00:00 AM	Avaliable
Lodestar Transport Services	Albuquerque, NM	1/1/2015 12:00:00 AM	Avaliable
Lodestar Transport Services	Jacksonville, FL	1/1/2015 12:00:00 AM	Avaliable
Lodestar Transport Services	Griffin, GA	1/1/2015 12:00:00 AM	Avaliable

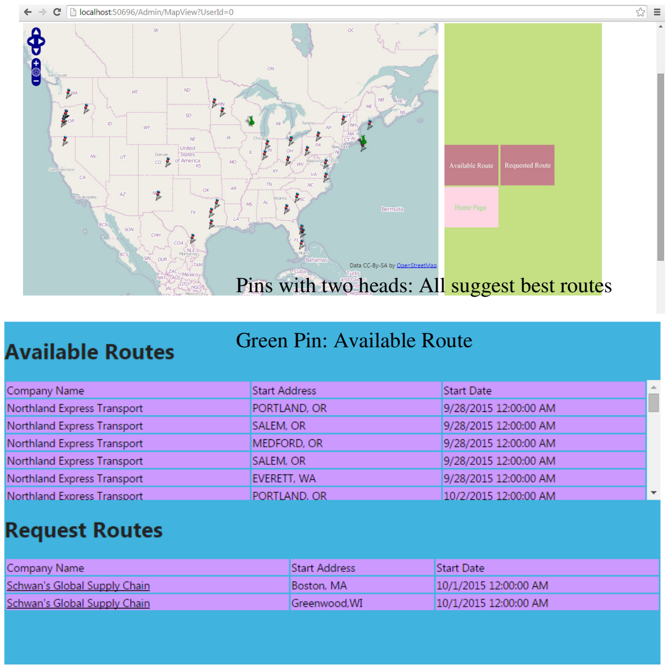
Figure 29. Available route



Company Name	Start Address	Start Date	Type
Schwan's Global Supply Chain	Boston, MA	10/1/2015 12:00:00 AM	Request
Schwan's Global Supply Chain	Greenwood,WI	10/1/2015 12:00:00 AM	Request

Figure 30. Request route

When the user clicks on the Map view button, the Map view page is displayed. This page shows all the requested routes (shippers) and available routes (carriers) in map view and tabular view. The green markers represent requests and two-headed pins represent available routes as shown in the Figure 31. When the user clicks on each request route, the Suggestion page for the route is displayed. The page shows alternative routes and the best route in tabular view as well as map view as shown in Figure 32. If there are no suggestions, the Suggestion page will display the message as shown in the Figure 33.


localhost:50696/Admin/MapView?UserId=0
Available Route
Requested Route
Home Page
Pins with two heads: All suggest best routes
Green Pin: Available Route

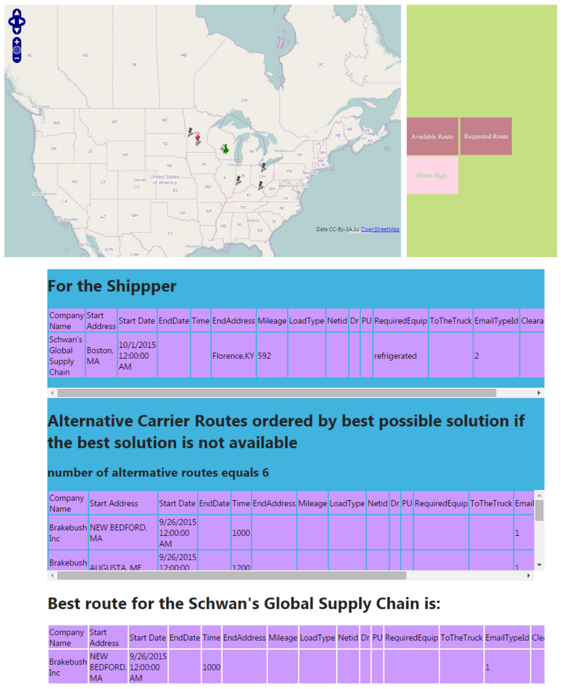
Available Routes

Company Name	Start Address	Start Date
Northland Express Transport	PORTLAND, OR	9/28/2015 12:00:00 AM
Northland Express Transport	SALEM, OR	9/28/2015 12:00:00 AM
Northland Express Transport	MEDFORD, OR	9/28/2015 12:00:00 AM
Northland Express Transport	SALEM, OR	9/28/2015 12:00:00 AM
Northland Express Transport	EVERETT, WA	9/28/2015 12:00:00 AM
Northland Express Transport	PORTLAND, OR	10/2/2015 12:00:00 AM

Request Routes

Company Name	Start Address	Start Date
Schwan's Global Supply Chain	Boston, MA	10/1/2015 12:00:00 AM
Schwan's Global Supply Chain	Greenwood,WI	10/1/2015 12:00:00 AM

Figure 31. Map view



For the Shippper

Company Name	Start Address	Start Date	EndDate	Time	EndAddress	Mileage	LoadType	Netid	Dr	PU	RequiredEquip	ToTheTruck	EmailTypeId	Cleara
Schwan's Global Supply Chain	Boston, MA	10/1/2015 12:00:00 AM			Florence,KY	592					refrigerated		2	

Alternative Carrier Routes ordered by best possible solution if the best solution is not available

number of altermative routes equals 6

Company Name	Start Address	Start Date	EndDate	Time	EndAddress	Mileage	LoadType	Netid	Dr	PU	RequiredEquip	ToTheTruck	Email
Brakebush Inc	NEW BEDFORD, MA	9/26/2015 12:00:00 AM		1000									1
Brakebush	AUGUSTA, ME	9/26/2015 12:00:00		1200									1

Best route for the Schwan's Global Supply Chain is:

Company Name	Start Address	Start Date	EndDate	Time	EndAddress	Mileage	LoadType	Netid	Dr	PU	RequiredEquip	ToTheTruck	EmailTypeId	Cle
Brakebush Inc	NEW BEDFORD, MA	9/26/2015 12:00:00 AM		1000									1	

Figure 32. Suggestion page

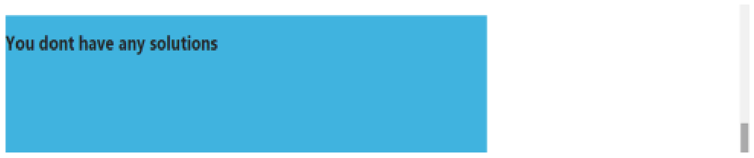

You dont have any solutions

Figure 33. Suggestion page 2

The admin user logs into the system and adds an Excel file for abbreviation. The user has to follow certain steps to obtain the desired result. This query result is as shown in Figure 34.

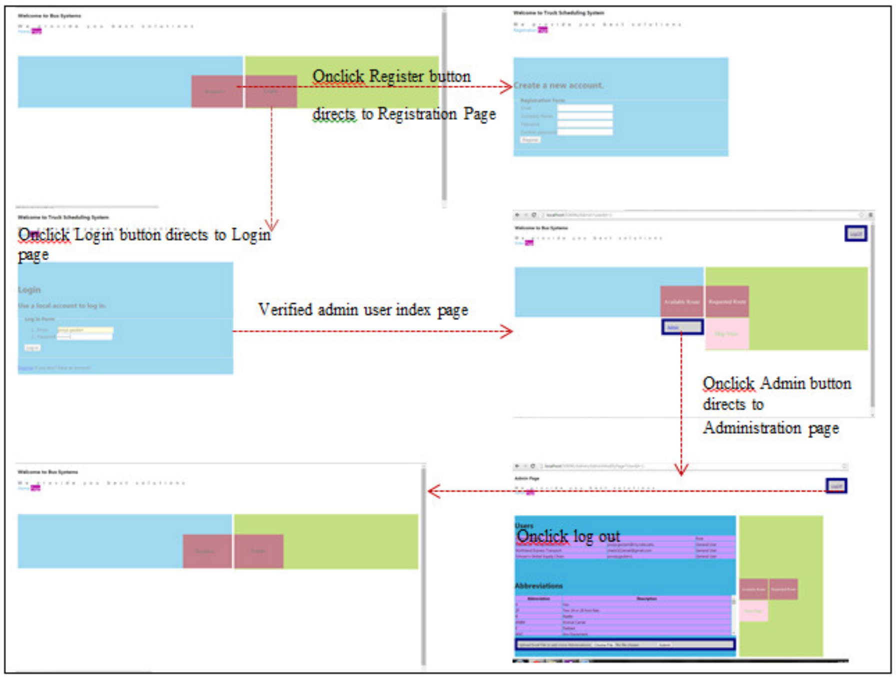

Onclick Register button directs to Registration Page
Onclick Login button directs to Login page
Verified admin user index page
Onclick Admin button directs to Administration page
Onclick log out

Figure 34. Upload file

Suppose the user logs into the system to find the best route. The user has to follow certain steps to obtain the desired result. This query result is shown in Figure 35 to display Suggestion Page.

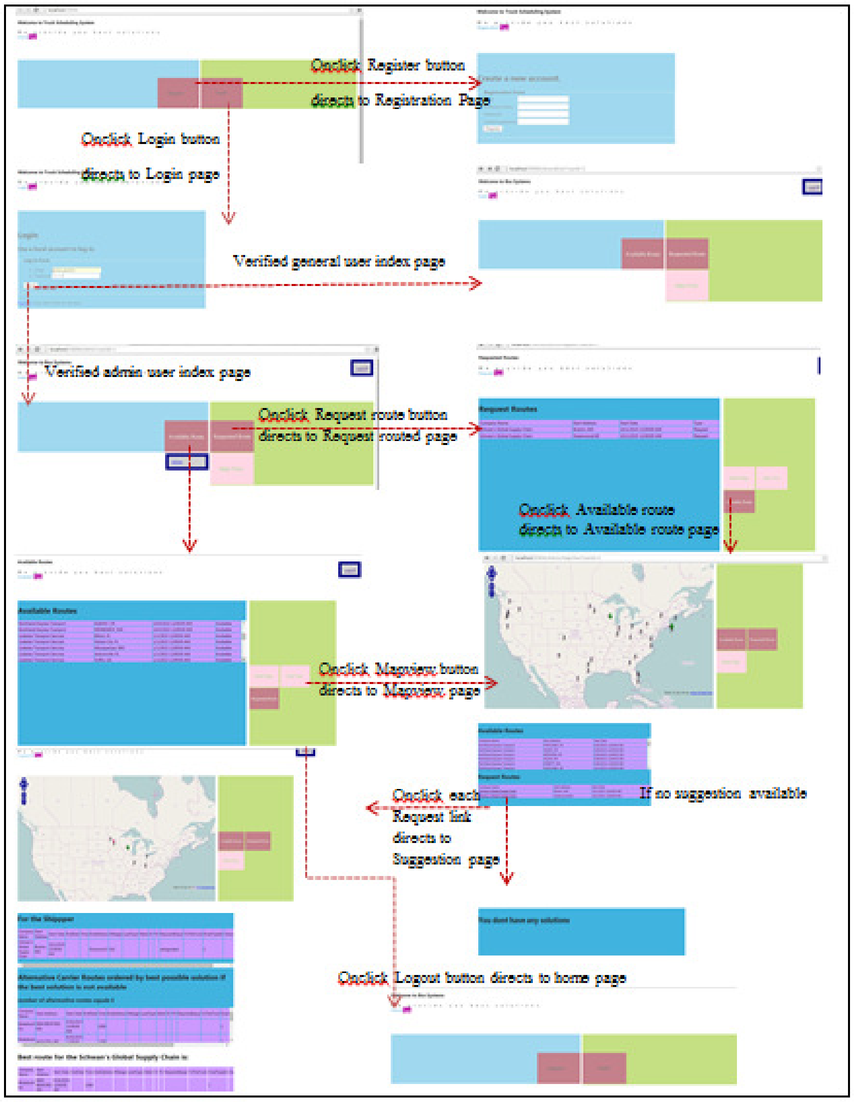
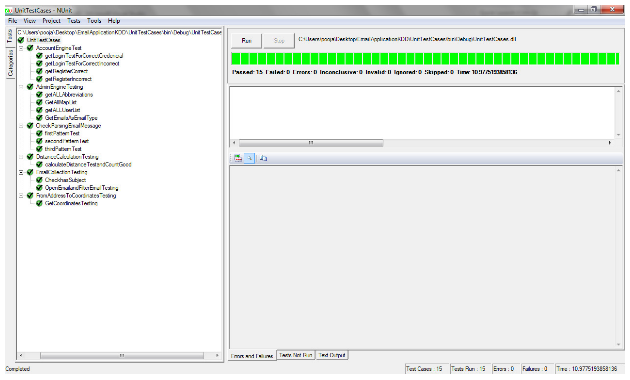

Onclick Register button
directs to Registration Page
Onclick Login button
directs to Login page
Verified general user index page
Verified admin user index page
Onclick Request route button
directs to Request routed page
Onclick Available route
directs to Available route page
Onclick Mapview button
directs to Mapview page
Onclick each
Request link
directs to
Suggestion page
If no suggestion available
Onclick Logout button directs to home page

Figure 35. Best route view page

4.1. Testing

NUnit test cases were developed in Visual Studio and tested in Unit UI. All possible inputs were provided to determine if the software works properly when subject to different inputs. The main testing goal for this paper to make sure that the development works properly with different scenario and displays the correct output. The approach used was Test Driven Development (TDD). NUnit test cases have consisted of four main testing scripts:

1. "EmailCollectionTesting.cs" to test if the system is reading emails, and to test if the subject is read correctly and attachments are counted.
2. "CheckParsingEmailMessage.cs" to check if the email body or attachments are read by the system and stored in the database
3. "FromAddressToCoordinatesTesting.cs" to test if the from the address correct latitude and longitude can be received.
4. "DistanceCalculationTesting.cs" to test if the good distance and the best route is saved in the database table.



```
C:\Users\pooja\Desktop\EmailApplicationKDD\UnitTestCases\bin\Debug\UnitTestCase
UnitTestCases
  AccountEngineTest
    getLoginTestForCorrectCredencial
    getLoginTestForCorrectIncorrect
    getRegisterCorrect
    getRegisterIncorrect
  AdminEngineTesting
    getALLAbbreviations
    GetAllMapList
    getALLUserList
    GetEmailsAsEmailType
  CheckParsingEmailMessage
    firstPatternTest
    secondPatternTest
    thirdPatternTest
  DistanceCalculationTesting
    calculateDistanceTestandCountGood
  EmailCollectionTesting
    CheckhasSubject
    OpenEmailandFilterEmailTesting
  FromAddressToCoordinatesTesting
    GetCoordinatesTesting
```

```
C:\Users\pooja\Desktop\EmailApplicationKDD\UnitTestCases\bin\Debug\UnitTestCases.dll
Passed: 15  Failed: 0  Errors: 0  Inconclusive: 0  Invalid: 0  Ignored: 0  Skipped: 0  Time: 10.9775193858136
```

```
Completed    Test Cases : 15    Tests Run : 15    Errors : 0    Failures : 0    Time : 10.9775193858136
```

Figure 36. NUnit test screen shot

Table 3. Test1

	TestCaseMethod	Test Process	Purpose
1	calculateDistanceTestandCountGood()	Check to see if the distance calculation is correct.	The test was done to check if the distance calculation gave the correct output.

Table 4. Test2

	TestCaseMethod	Test Process	Purpose
1	firstPatternTest()	Check to see if the “firstPattern()” reads emails from the “**b**rakebush” company, extracts the information, and saves it to the database	The test was done to see if the text pattern was read properly, extracted the information, and stored it to the database.
2	secondPatternTest()	Check to see if the “secondPattern()” reads emails from the “Schwan” company, extracts the information, and saves it to the database.	The test was done to see if the text pattern was read properly and extracted the information to store in the database.
3	thirdPatternTest()	Check to see if the thirdPattern() reads emails from the “Lodestar” company, and extracts the information, and saves it to the database.	The test was done to see if the text pattern was read properly and extracted the information stored it to the database.
4	fourthPatternTest()	Check to see if the “fourthPattern()” reads emails with attachments (Excel) and extracts the information, and saves it to the database.	The test was done to see if the text pattern was read properly and extracted the information, storing it to the database.

Table 5. Test3

	TestCaseMethod	Test Process	Purpose
1	GetCoordinatesTesting()	Check to see if the string address was input; the system would give correct output as the latitude and longitude.	The test was done to check if the correct latitude and longitude were delivered when a string address was an input.

Table 6. Test4

	TestCaseMethod	Test Process	Purpose
1	OpenEmailandFilterEmail Testing()	Check to see if the counting on the email was correct.	The test was done to see if the email count was correct. The test passed when the correct parameters were sent.
2	CheckhasSubject()	Check to see if the email subject was read properly.	The test was done to see if the system read the subject properly. e.g. the subject as "Northland - OR Available Loads." was read as available shipper.
3	CheckHasattachment(int[] ids, Mailbox inbox)	Check to see if the email attachment was read properly.	This test was done to see if the system read the email with an attachment properly.

4.2. Time Response

The response time was calculated based how much time each page took to load. If the user is not registered already, the user has to register first. The registration page or the login page's response time is 3-5 seconds. It takes 6-25 seconds to log in to the system. During the login process, email data are read and stored in the database.

After the user logs in, there are 5 options from which to choose from: Available Route, Admin, Requested Route, Map View, and LogOff. To upload an Excel file on the Admin page, it takes 4-7 seconds. Loading Available Route page, takes 3-5 seconds. To load the Requested Route page, it takes 3-5 seconds. Loading the Map View page, it takes 4-7 seconds. Each

Requested Route has a suggestion link; which takes 4-7 seconds to load the page. It takes 3-5 seconds to log out of the system and 6-25 seconds to load index page.

Table 7. Response timetable

Serial Number	Step	Time taken in seconds
1	Login page	3-5
2	Registration page	3-5
3	Available route	3-5
4	Admin	5-7
5	Requested route	3-5
6	Map View	4-7
7	Log Out	3-5
8	Index page	6-25

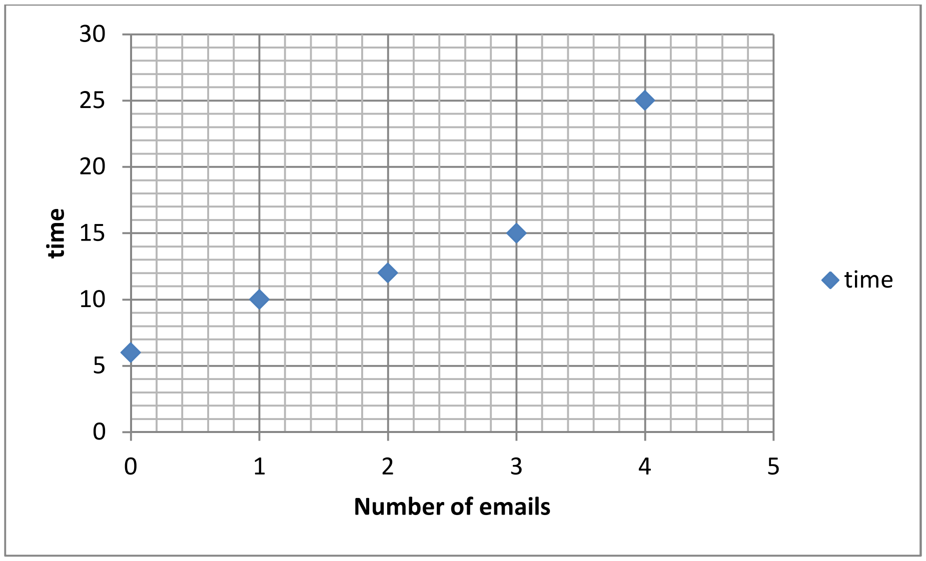


Figure 37. Time vs the number of email

Figure 37 shows the execution time to load the index page, while this page is loading; at the backend the email messages are read by the system. We can see that the time increases with a higher number of email messages. Time increases if the email has an attachment; we can see that there is an increase in the slope from email 3 to email 4 because time taken increases from 15 seconds to 25 seconds. Email 4 has an attachment.

4.3. Alternative Technology

For this project the technologies used are as follows: C#, MS SQL Server, HTML, JavaScript, Razor and CSS. We could have used another object oriented programming language such as Java and Relational database management system such as MySQL.

4.4. Challenges Faced

This project was both learning and implementing experience. Some challenges were faced during the project, such as implementing the Entity Framework (ORM) with C#. Also, parsing the data on the server side was challenging.

5. CONCLUSION AND FUTURE WORK

5.1. Conclusion

With this project, we have designed and developed a web application for a truck scheduling system. We have used C# as an object oriented programming language; MS SQL server as the database management system; and web technologies such as HTML, JavaScript, Razor, and CSS, to create the GUI. Various frameworks and libraries have been used to parse text data, to read and manipulate Excel files, to get latitude and longitude, and to access the database. LINQ expression has been used to query from the database, and the best and alternative routes have been provided. Each shipper requests a route or carrier that advertised for availability, sending an email to the truck scheduling system with the predefined email pattern style. The system reads emails with an attachment that should be in Excel format. Here, the unstructured email data are transformed to the structured data to store in the database. To find meaningful information, such as the best and alternative routes, several queries have been done. Nunit is utilized to test module using unit test cases.

To test the web application, four emails were sent to the system; the email subjects contained "available", "remaining" and "need help." During the test phase, emails from the carrier had subjects such as "available" or "remaining" and emails from the shipper had subjects such as "request," "need help," etc. Also, emails were sent with or without attachments.With the help of regular expression, the emails were parsed and information was stored in the database tables. Then data analysis was done to find the best route and alternative routes for the carrier. After that the result was given to the end user in the tabular format as well as map view.

5.2. Future Work

The truck scheduling system currently gives the best and alternative routes for the carriers and the shippers according to our preset search criteria. We plan to extend it to allow more flexible search condition combinations and involve users in this selection process, in order to provide more user-centered solutions for possible truck scheduling.

6. REFERENCES

[1] Han, J., & Kamber, M. (2006). Data Mining: Concepts and Techniques (2nd ed.). *Morgan Kaufmann publications*.

[2] Introduction to Information Retrieval, Cambridge University (2008) by Christopher D. Manning, Prabhakar Raghavan and Hinrich Schütze,

[3] Atul Adya, José A. Blakeley, Sergey Melnik, and S. Muralidhar. 2007. Anatomy of the ADO.NET entity framework. In Proceedings of the 2007 ACM SIGMOD international conference on Management of data (SIGMOD '07). ACM, New York, NY, USA, 877-888.

[4] Retrieved Oct 15, 2014 from Valley Logistics Website: http://valleyexplogistics.com/services.html

[5] Retrieved Oct 15, 2014 from Click Software Website: http://www.clicksoftware.com/workforce-planning-software-solutions

[6] Jie Tang, Hang Li, Yunbo Cao, and Zhaohui Tang. 2005. Email data cleaning. In Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining (KDD '05). ACM, New York, NY, USA, 489-498.

[7] Clustering by means of medoids by Leonard Koufman & Peter J. Rousseeuw.Vojtech Juhász. 2012. Full-text search in email archives using social evaluation, attached and linked resources. In Proceedings of the 21st international conference companion on World Wide Web (WWW '12 Companion). ACM, New York, NY, USA, 857-860.

[8] Retrieved Oct 15, 2014 from The Standish Group Website: http://www.versionone.com/assets/img/files/CHAOSManifesto2012.pdf

[9] Retrieved Oct 15, 2014 from Microsoft Corp Website: http://msdn.microsoft.com/en-us/library/dd381412(v=vs.108).aspx

[10] Retrieved Oct 15, 2014 from Microsoft Corp Website: http://msdn.microsoft.com/en-us/library/hs600312(v=vs.110).aspx