

TRAFFIC MONITORING SYSTEM

GROUP NUMBER: 7



GROUP MEMBERS:

Kush Patel
Tejas Ravi
Jeffrey Gillen
Shreyas Bhandare
Kisholoy Vinayak Ghosh
Aaditya Shukla

kush.patel@rutgers.edu
tr332@scarletmail.rutgers.edu
jeffrey.gillen@rutgers.edu
shreyas.bhandare@rutgers.edu
kisholoy.v.ghosh@rutgers.edu
aaditya.shukla@rutgers.edu

Table of Contents

I.	Problem Statement.....	3
II.	Algorithm.....	5
III.	Data Collection.....	6
IV.	Functional Features.....	7
V.	Glossary of Terms.....	9
VI.	System Requirements	
	a. Functional Requirements.....	10
	b. Non-Functional Requirements.....	10
	c. On-Screen Appearance Requirements.....	11
VII.	Individual Breakdown.....	11
VIII.	Functional Requirement Specification.....	12
IX.	Use Cases	
	a. Casual Description	13
	b. Use Case Diagram.....	14
	c. Traceability Matrix.....	15
	d. Fully Dressed Description	16
	e. System Sequence Diagram.....	18
X.	User Interface Specification	
	a. Preliminary Design.....	20
	b. User Effort Estimation.....	21
XI.	Domain Model	
	a. Concept Definition for use case 1.....	23
	b. Association definitions for use case 1.....	23
	c. Attribute Definitions for use case 1.....	24
	d. Concept Definition for use case 2.....	24
	e. Association definitions for use case 2.....	25
	f. Attribute Definitions for use case 2.....	25
	g. Traceability Matrix.....	26
XII.	System Operation Contracts.....	27
XIII.	Prediction Model.....	28
XIV.	Plan of Work.....	30
XV.	References.....	31

Problem Statement

The internet has advanced tremendously over the past decade. In today's day and age, our lives have become dependent on using the internet. From checking email to deciding what food to eat, every activity is governed by various websites. Websites also offer map applications that offer services such as navigation and traffic updates. Navigation is very essential if someone wants to go to an unknown location. Live time traffic is also very essential to determine congestion on a particular route and plan routes accordingly.

There are several services that offer this feature; for example, "Traffic.com" and "Yahoo! Maps Live Traffic" etc. All these services only account for current traffic conditions in a given geographical area. Even though current traffic models give a good idea of the congestion on a particular highway, it still is not completely accurate because the prediction can be based off of very few traffic reports. If there are no traffic reports, one cannot know what the traffic levels look like at a specific time of the day. With the help of current traffic intimation models, a user will get to know about current traffic updates dynamically as he travels along the route. However, it would be more meaningful if the user knows what to expect in terms of traffic even before starting from his starting location. This project aims to take traffic prediction to another level. Instead of just using current traffic patterns, we will use historic traffic data to give the user a better estimate of traffic patterns. Below we have proposed a solution to this problem. Several questions one would ask are What? How? Why? Who? Etc. We will explain each aspect below.

This project will create a One Stop Site for a user on a busy weekday or a relaxed weekend. It will give the user the ability to query for traffic information using a zip code, day of the week and time of the day. So when a user does perform a query, the traffic prediction would be based on the current predictions as well as the historic traffic data that the route experienced over the past few months. In this way, the user will be able to see what route is experiencing congestion and hereby decide whether to take another route. As add on, we also will be collecting weather information every time a traffic report is collected. If there was any correlation between bad weather and traffic congestion, this will be used towards our prediction algorithm if the query day has bad weather. The fact that the weather was bad can be used towards our model to make prediction more accurate. Hence we will be using a weightage algorithm that will give output based on current traffic conditions, old traffic data as well as weather. The service will ideally be able to suggest routes that avoid areas with a very high projected concentration of traffic. This will allow the user to find an ideal route with minimal exertion. The assimilated data and traffic information could be used in the future to help in improving road infrastructure or traffic diversion based on traffic patterns and weather conditions.

The front end of the software would include a website where:

1. The user will be able to enter the zipcode[destination or (source & destination)],
2. The day of the week and
3. The time of the day of the area for which the traffic reports are required.

The service will then consult a directions service, which in our case is Google Maps, in order to find the traffic reports. This route will be highlighted with different colors referring to different intensities of traffic congestion, as per our algorithm. The key for the different colors will be available for the user to understand. Along with the traffic report, the user will be able to see the weather predictions based on the input that the user gives. As add-ons to the traffic and weather forecast, the application will provide the user with an option to look for restaurants and parking lots in the area.

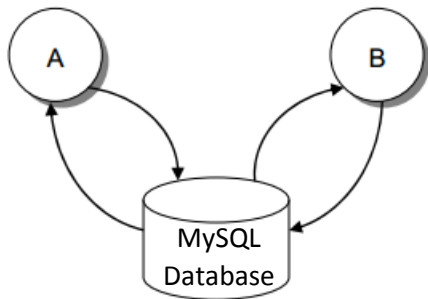
Algorithm:

How does our algorithm predict traffic congestion? A lot of the traffic websites like Yahoo, Bing, Mapquest use only the current traffic incident reports to predict traffic. A typical use case would be a user logging into Yahoo at 7am to check traffic conditions for a particular highway. Due to it being 7am, Yahoo will show no congestions because normally the congestion starts around 8am. If the user has a long commute, they would get stuck in traffic because by 8AM, the traffic has increased tremendously and it's too late for him to take a detour. Hence to handle that use case, our algorithm would also take into account previous day's data. So when a user does login to our website, they will be given an option to select a time and if the time is past the current time (future), the algorithm will query the database for previous historic data for the same time and make a decision accordingly.

Another aspect of traffic prediction is accident conditions. We all know accidents do not occur every day on the same road; if an accident caused traffic congestion today, it would not cause traffic congestion tomorrow. In order to handle this use case, we will be using a weighted algorithm prediction hereby analyzing current traffic data as well as historic traffic data.

Let's assume the user wants to predict traffic for time 9am and the query was executed at 8am. This means that our current database has traffic incident data up to 8am. Our weighted algorithm will analyze the current traffic data and check whether there has been any change in traffic conditions in the past hour (between 7AM-8AM). If there have been any changes, we know traffic congestion is starting to increase. In this case, our traffic predictions will put more weight on the current traffic and less weight on yesterday's traffic. The reason we chose to have more weight on today's traffic is because if there was an accident in the morning, we simply do not want to predict light traffic based on yesterday's traffic. Additionally, we are also using the weather to help with predictions. If the weather is raining or snowing, it will perform a query on the database for snow/rain days and use the traffic incident data for those days. The weather data will have a weight factor associated as well, if and only if the weather was bad; meaning it rained/snowed.

Data Collection:



Our project website will be accessing data via a shared element as shown on the left. To predict traffic conditions, our algorithm will require traffic incident data. This data has been uploaded by users themselves who are out on the road via their smartphones. Our backend will have a datacollection script running continuously that will be pulling data from the Google API and storing them into our local MySQL database. For our prediction, we will be storing data into 3 separate databases, namely historic traffic, current traffic and weather

Database	Data Storage Interval	Total Data Reports
Current	Once every 2 seconds	Max 24 hours
Historic	Once every hour	Max 4 Months
Weather	Once every hour	Max 4 Months

The above table shows the data collection scheme of the three separate databases.

The method we intend to use in realizing this idea is:

- **Current Database:**
The map and 'Traffic information' which will be seen on the site will be gathered from Google Maps. Google maps allow up to 25,000 requests every day. This would be sufficient to make a database with an update on the traffic once every 2 seconds.
- **Historic Database:**
This is similar to our Current Database, however we will be storing once every hour. The major difference between current and historic, besides the update interval, is historic has data upto 4 months whereas current will only have upto 24 hours.
- **Weather Database:**
The 'Weather information' will be gathered from sites like 'wunderground.com' or 'openweathermap.org'

As a side note, please keep in mind "Total Data Reports" mean for how many hours/months of data the data will be stored. If the database has reports for any time greater than that, the newest incident will be saved and oldest incident will be deleted.

Functional features

The previous versions of the project gave traffic predictions for the entire tri-state. They did not have any additional features. This project will bring out some additional features to the application that will enhance the user experience.

- Restaurants and Parking
- Navigation with suggested routes
- Plan a vacation
- Alternative Mode of transportation
- Mobile Android Application

Restaurants and Parking:

There are plenty of applications out there that will give you locations for food on the map. The only issue there is you do not know whether the restaurant is open or not. A lot of times users look up for restaurants late at night or during the night time. Different restaurants have different operating hours. Hence in our application, we will have a layer for restaurants and food, however it will only show you restaurants that are open in the nearby vicinity. No user wants to show up to a restaurant not knowing its closed or will be closed soon. Additionally, when the map points are enabled for the restaurants, it will also display the current estimated time to that place, so a user can make a decision.

Similar to restaurants, we will be adding another layer for parking as well. This can be used to find parking spots in the nearby vicinity. This is different from applications like google because Parking Spots is a layer and can be just enabled easily whereas in google, one has to cancel the current route and perform a search for parking.

Navigation:

Applications with only traffic updates are useless. Hence this application will allow a user to perform a navigation query with an origin and destination address. The user will be able to see the traffic conditions on this navigated route and hereby make a decision whether they want to take the suggested route. The application will also give the user an option to take a detour if they would like.

Vacation Planning:

If a user wants to plan out a vacation in the upcoming weeks, they will have an option to query for weather and possible traffic conditions during the travel days. The application will use the date and the day of the week and perform queries for predicted traffic conditions. This prediction however will be inaccurate due to not having enough traffic data, but this will still be helpful to plan out expected travel time.

Alternate Mode:

If the user's query returns a heavily congested route, and there are no possible detours, the user will be stuck in a limbo. To tackle this condition, our application will also have an alternate mode which will look for closest train stations that the user can take. This will reduce user's overall travel time and hereby tackle traffic.

Mobile Android Application:

Smartphones have become very popular in the past few years and users are very dependent on accessing application on their devices. It would be very convenient for the user to access our map application on their handheld. Hence, we plan to build an android application which will give the user the same features as the desktop application.

Glossary of Terms

1. User: A person who wishes to use the traffic monitoring application to view traffic congestion in a user-entered zip code and be able to choose a faster route to destination
2. ZipCode: A 5 digit number that defines a city or town in the United States. This will be an entry field for the user
3. MySQL Database: A relation database management system that will be used to store traffic incident data as well as weather information
4. Google Maps API: An application programming interface (API) that our system will use to access traffic incident data provided by google. This data will be stored into the MySQL Database
5. Weather Service: A website that contains weather data and will be accessed by our application to store into MySQL Database
6. Traffic Severity: This will depict the different levels of traffic congestion. For the first phase of our project, we will use (Low, Moderate, Moderate High and High)
7. Administrator: A person who is responsible for the management of the traffic monitoring webservice. They also need to ensure the data collection script is also running smoothly
8. Desktop Application: A web application that a user will be able to access on their web browser of a laptop/desktop computer
9. Mobile Android Application: An android application that a user can access on their android smartphone. The application will provide all the features a desktop application provides.

System Requirements

Functional Requirements:

Below are the functional requirements of our application

Identifier	PW	Requirement
REQ 1	5	Upon logging in to the web interface, the application shall prompt the user to enter time of day and zip code and display the most upto date traffic predictions for the entered zip code
REQ 2	5	The application shall display the most upto date weather data.
REQ 3	4	If the user selects Navigation Tab, the application shall prompt the user for a start and end destination and output a navigation route.
REQ 4	5	The application shall collect data in the background for traffic and weather and store them in the database. It also needs to keep the website in sync with the database
REQ 5	3	The application shall give the user to plot the restaurants and parking lots on the map and also display the estimated travel time to each of these applications
REQ 6	3	The application shall give the user an option to select other modes of transportation if there is heavy traffic on the selected route. The output will contain the estimated travel time from the closest train station from the origination to the closest train station from the destination.
REQ 7	4	If there is congestion along the navigation route, the application shall provide the user with an alternate route

Non - Functional Requirements:

Identifier	PW	Requirement
REQ 8	4	The application shall provide the user with a neat and easy to use Interface for the user
REQ 9	3	The application shall overlay the roads with different colors depending on the appropriate traffic conditions experienced at that instant.
REQ 10	2	The application shall provide traffic incident graphing and charting abilities on the webpage (trend over months) for informational purposes
REQ 11	5	The application shall use a local algorithm to predict traffic conditions
REQ 12	2	The application database shall have enough memory to store traffic incident data for over 2 months for different zip codes
REQ 13	3	The administrator shall easily be able to change the time interval of data collection for weather and traffic

On-Screen Appearance Requirements:

Identifier	PW	Requirement
REQ 13	5	Based on the prediction made by the application using the user's input, the application shall display the relevant information in an ergonomic way.
REQ 14	5	The application website shall provide an interface which will display a large size map with pan and zoom functionality.
REQ 15	5	The application website shall provide a key along the side of the map to understand the color code seen on the roads in the map.
REQ 16	3	The application website shall continuously prompt current traffic and weather news of the zip code entered by the user.
REQ 17	5	The application shall provide a drop down menu for all the different features that it provides and for fields that requires the user to make a choice(Ex: time, day of week etc.)

Individual Breakdown

The project can be broken down into 3 mini projects. These projects are parts that can be developed without any dependency. As time passes, these mini-projects will be integrated into our final project

- 1) Kush and Tejas: Kush and Tejas will work on the data collection piece of the project. The historic data collection is very vital since all the functions and algorithms are dependent on it.
- 2) Kisholoy and Shreyas: Kisholoy and Shreyas will be doing the development of the website and designing the user interface. This is very vital since this is the front end of the application and how the user will be accessing the service.
- 3) Jeff and Aaditya: Jeff and Aaditya will be working on the algorithm development and develop functions to integration our data on the website map.

Functional Requirement Specification

Stakeholders

- Users
- Administrators

Actors and Goals

- User:
 - Initiating Actor
 - The user will be accessing the application on their web client to obtain traffic predictions and also navigation between a start and end point. The application will also be providing other features that will go hand in hand with the traffic predictions
- Administrator:
 - Initiating Actor
 - The administrator's primary role is to ensure the traffic and weather data collection modules are working smoothly. The administrator must also ensure the webserver is in sync with the database (display the most up-to date traffic information).
- Backend Script:
 - Initiating Actor
 - The backend script is very vital for data collection. The script will be running continuously on the backend and collect traffic and weather data at periodic intervals. The intervals are defined in the "Data Collection" sections of this document. This data will be store into a MySQL database.
- Database:
 - Participating Actor
 - The database will serve as a central repository for all the traffic and weather data written by the backend script.
- Mapping Service:
 - Participating Actor
 - The goal of the mapping service is to plot the output of the traffic prediction algorithm on the map for the user to display. It will have different colors to depict different congestion levels.
- Weather Service:
 - Participating Actor
 - The goal of the weather service is to provide data to our application to store into the database. The weather service will have the data available and the backend script will be grabbing the data from the website
- Traffic Service:
 - Participating Actor
 - The goal of the traffic service is to provide data to our application to store into the database. The traffic service will have the data available and the backend script will be grabbing the data from the API.

- Restaurant Service:
 - Participating Actor
 - This is one of the enhancements our application will provide. The restaurant service will display nearby restaurants in the nearby vicinity that are open at the time of execution. It will not display restaurants that are closed at the time of the day. It will also display the estimated travel time to each of these restaurants which the user can decide to choose.
- Parking Lot Service:
 - Participating Actor
 - This is one of the enhancements our application will provide. The parking lot service will display the nearby parking lots/ decks in the vicinity.

Use Case Casual Description

UC-1: Get Traffic Predictions and Weather Data

User will enter the time of day and the zip code into the application. Current traffic predictions and weather data will be displayed to the user.

Requirements: 1, 2

UC-2: Navigation

User will enter a starting and ending locations into the application. The application will display a navigation route with restaurants and parking lots marked on the map, with an estimated travel time. If heavy congestion is detected, the application will provide an alternate route. User is also able to select an alternate mode of transportation, and will be directed to the nearest train station.

Requirements: 3, 5, 7

UC-3: Data Collection

A script will be running to continuously collect traffic and weather data and store the data in the database. The script will also keep the website updated with the latest data.

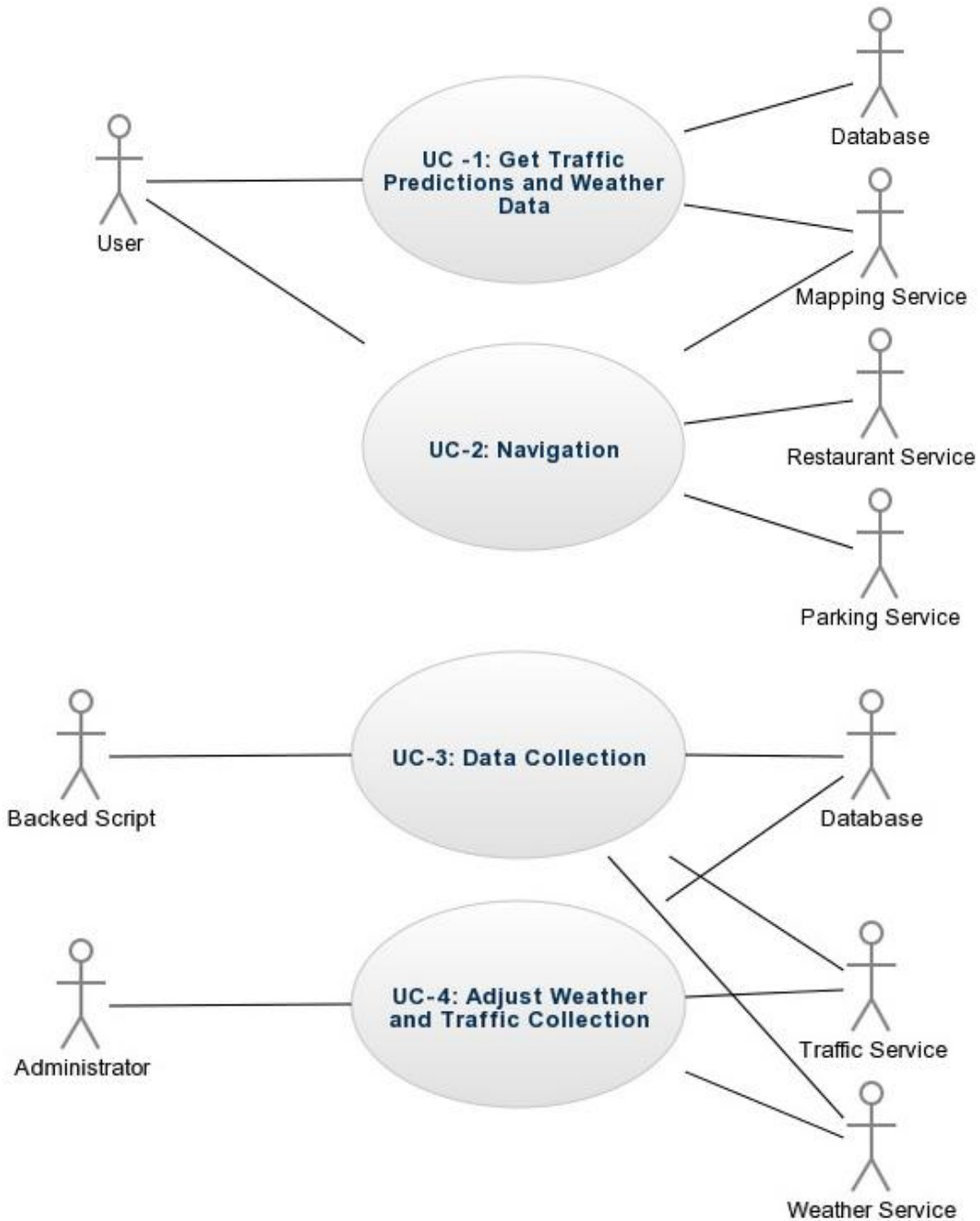
Requirements: 4

UC-4: Adjust Weather and Traffic Collection

The administrator will be able to change the time interval of data collection for weather and traffic.

Requirements: 13

Use Case Diagram



Traceability Matrix

Requirement	Weight	UC-1	UC-2	UC-3	UC-4
REQ-1	5	X			
REQ-2	5	X			
REQ-3	4		X		
REQ-4	5			X	X
REQ-5	3		X		
REQ-6	3		X		
REQ-7	4		X		
REQ-8	4	X	X		
REQ-9	3	X	X		
REQ-10	2				
REQ-11	5	X	X		
REQ-12	2			X	X
REQ-13	3			X	X
REQ-14	5	X	X		
REQ-15	5	X	X		
REQ-16	5	X	X		
REQ-17	3	X		X	X
REQ-18	5	X	X		
Weight		45	46	13	13

Fully Dressed Use Cases

Use Case UC-1: Get Traffic Predictions and Weather Data

Related Requirement: REQ 1, REQ 2

Initiating Actor: User

Actor's Goal: To obtain current traffic predictions and weather data

Participating Actors: Database, Mapping Service

Preconditions: User has logged into the application

Postconditions: Application has displayed traffic predictions and weather data

Flow of Events for Main Success Scenario:

1. **User** enters time of day and zip code into the application
2. **System** requests traffic and weather data from the **Database**
3. **Mapping Service** will display the traffic predictions and weather data

Flow of Events for Alternate Scenarios

1. **User** enters invalid information
 - a. **System** prompts user to re enter zip code and time of day
2. Requested data is unavailable
 - a. **System** will display that the service is currently unavailable

Use Case UC-2: Navigation

Related Requirements: REQ 3, REQ 5, REQ 7

Initiating Actor: User

Actor's Goal: To obtain a navigation route from their starting location to their destination

Participating Actors: Database, Mapping Service, Restaurant Service, Parking Lot Service

Preconditions: User had logged into the application

Postconditions: Application has displayed a navigation route and guided the user to their destination

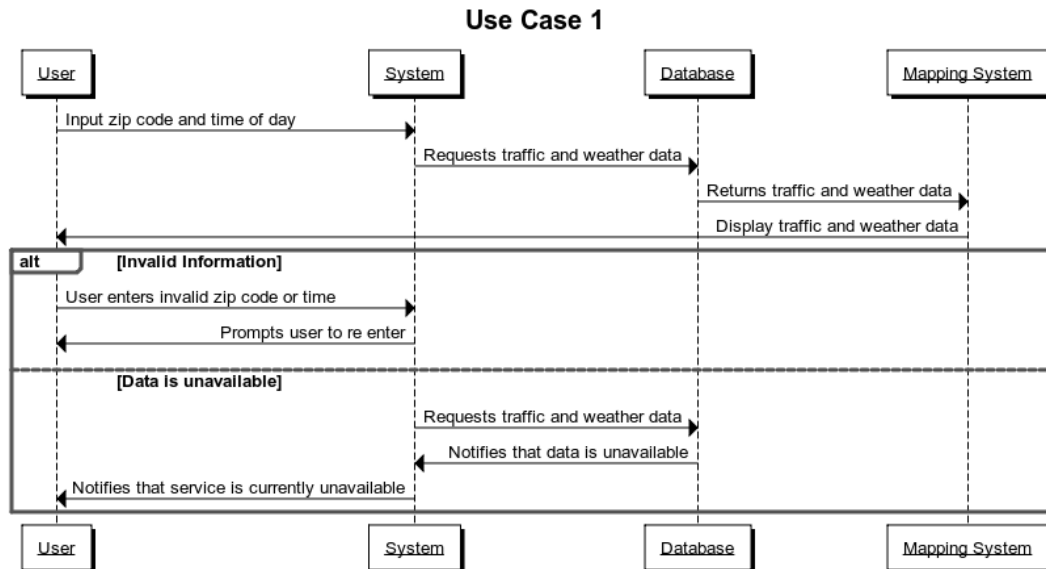
Flow of Events for Main Success Scenario:

1. **User** enters starting location and destination into the application
2. **Mapping Service** displays the most direct route to the destination
 - a. **Restaurant Service** will display restaurants and their estimated travel time
 - b. **Parking Service** will display parking lots and their estimated travel time
3. **System** requests traffic data from **Database** along the suggested route, checking for heavy congestion
4. **User** navigates along the suggested route and reaches the destination

Flow of Events for Alternate Scenarios:

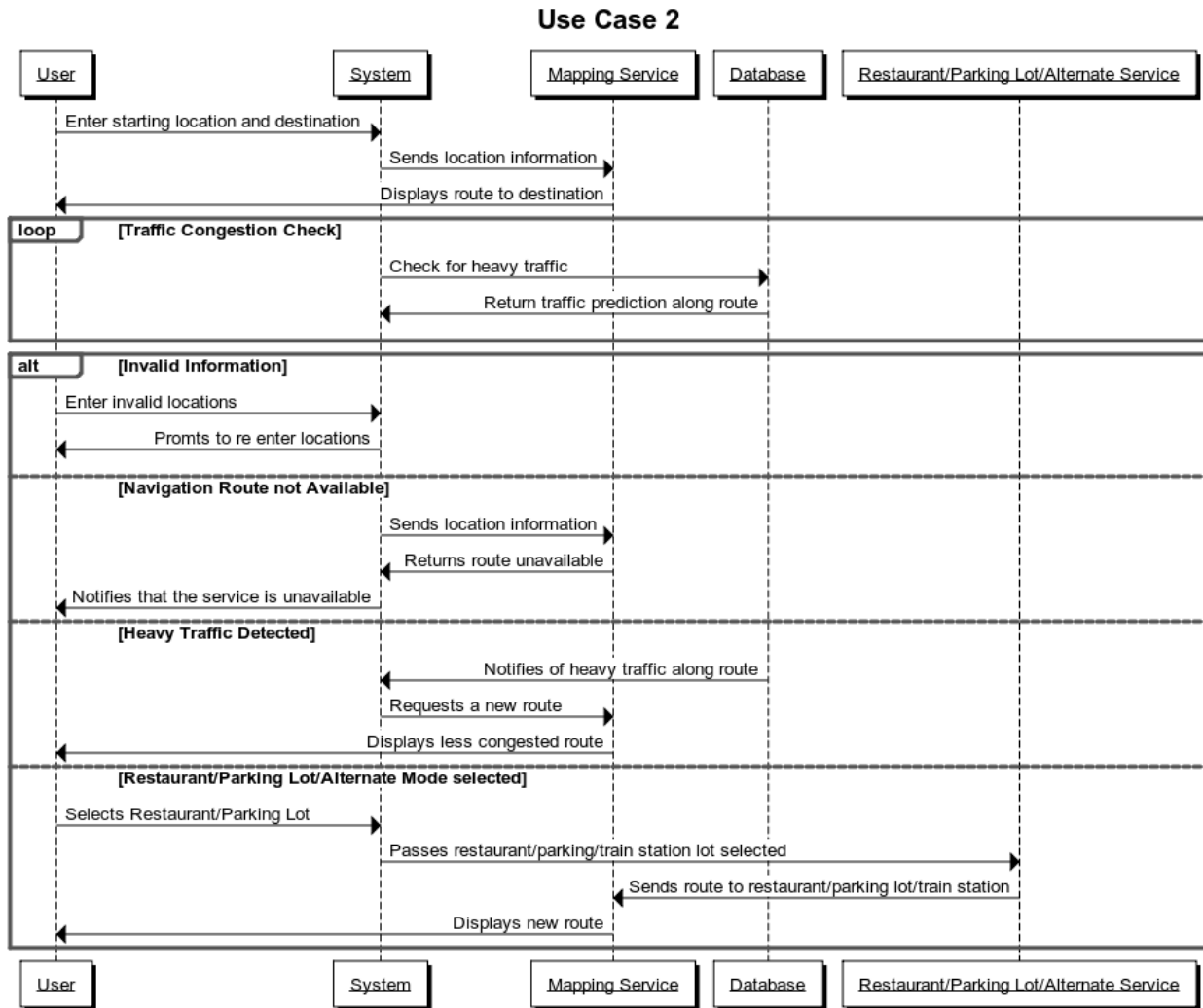
1. User enters invalid information
 - a. **System** will prompt the user to re-enter starting location and destination
2. Navigation route not available
 - a. **System** will display that the service is currently unavailable
3. Heavy traffic detected
 - a. **System** will request a less congested route
 - b. **Mapping Service** will display the route
4. User selects a restaurant, parking lot, or alternate mode of transportation
 - a. **Mapping Service** will display a route to the restaurant, parking lot, or nearest train station with an estimated time. If navigating to a train station, the estimated travel time of the train to the original destination nearest train station will be displayed

System Sequence Diagrams



This system diagram shows the normal flow of **Use Case 1** and the alternate flow for if the array is empty. The Sequence Flow can be summarized as:

- The user will enter the application with his login credentials.
- The User will then enter the data and submit it on the Application.
- The Application queries the database for traffic and weather data within the inputted zip code and time of day.
- The Application then calls the Mapping class to process these inputs, and calls the Database requesting the traffic and weather points in the area desired.
- The Mapping class parses the information and constructs the link to the Mapping Web Service.
- The Mapping Web Service is called and returns a map image to the Application. The Application will display the image for the user.
- For the alternate flow if the array of data for any specific area is empty, the map will show the zip code still, but a text box will appear at top the show that the zip code has no traffic history.



This system diagram shows the normal flow of **Use Case 2** and the alternate flow for if the user selects alternate services or if the array is empty. The sequence can be summarized as

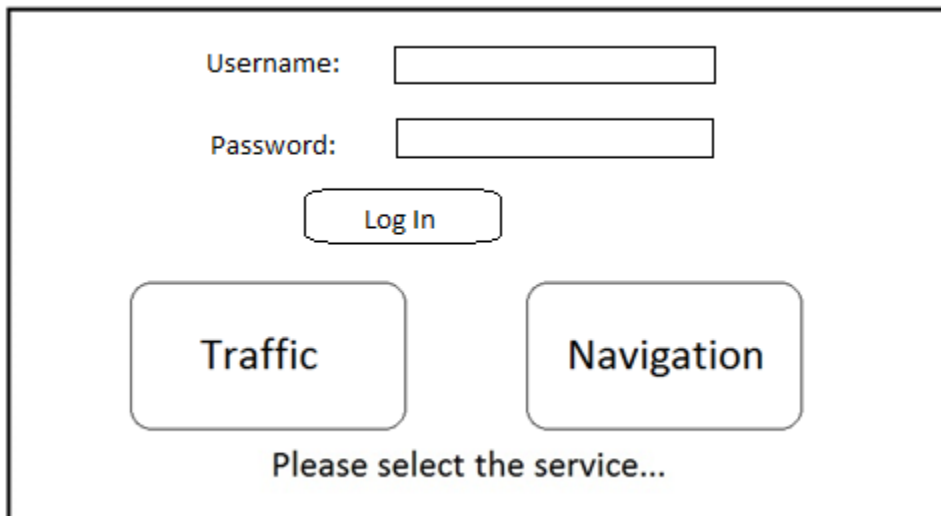
- The user will enter the application with his login credentials.
- The User will then enter the data (start and destination zip codes) and submit it on the Application.
- The Application calls the Mapping Class with the given inputs and initializes the Map.
- The Class then calls the Prediction model that runs the data from the database in a loop with the current input from the user to generate results.
- The Database returns an array of results.
- The Mapping class parses the data from the Mapping Service and the processed data and constructs an image URL to display.
- The Application displays the image of the route along with the directions.
- For alternate flow if the array of data for any specific area is empty, the map will show the zip code still, but a text box will appear at top the show that the route between target zip codes has no traffic history.

- Apart from the above alternate flow, the alternate flow would include flows where the user will enter data and submit to the application and the system will call the RestaurantService or the ParkingLotService or the AlternateService to parse data to the Database. When these services are invoked the final data from the Database is passed to the Mapping Service which is used to display the result on the map in the website.

User Interface Specification

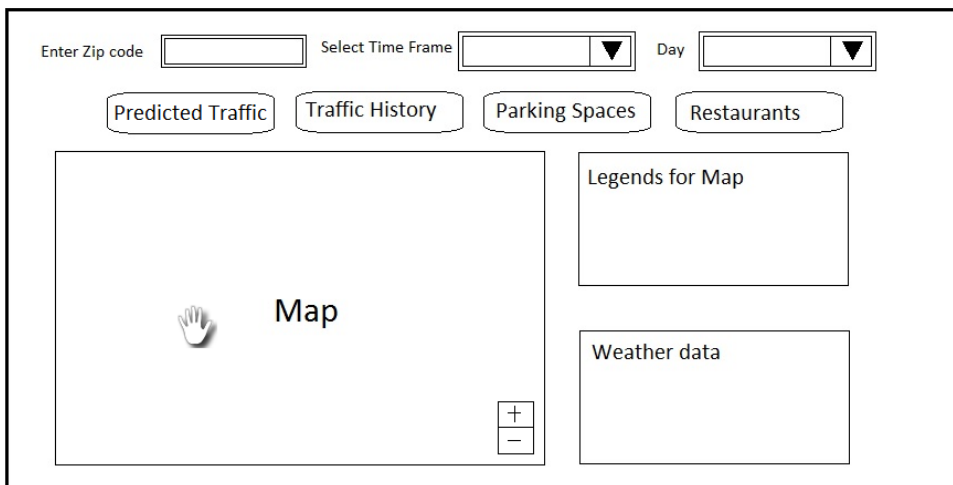
Preliminary Design

Preliminary User Interface design for fully dressed **Use Case - I**



A preliminary user interface design for a login and service selection screen. It features a login section with labels 'Username:' and 'Password:' followed by text input fields. Below these is a 'Log In' button. Underneath the login section are two large, rounded rectangular buttons labeled 'Traffic' and 'Navigation'. At the bottom of the screen is the text 'Please select the service...'.

Fig. 1: Home screen window options.



A detailed preliminary user interface design for a home screen. At the top, there are three input fields: 'Enter Zip code' with a text box, 'Select Time Frame' with a dropdown arrow, and 'Day' with a dropdown arrow. Below these are four buttons: 'Predicted Traffic', 'Traffic History', 'Parking Spaces', and 'Restaurants'. The main area is dominated by a large 'Map' section, which includes a hand cursor icon and zoom-in (+) and zoom-out (-) buttons. To the right of the map are two stacked rectangular boxes labeled 'Legends for Map' and 'Weather data'.

Fig. 2: Basic Service window.

Stepwise data entry and displaying of on-screen results:

- 1) Select the 'Traffic' service on homepage (fig.1) by clicking the appropriate option.
- 2) A New window (fig.2) will appear on the screen.
- 3) Enter the zip code, select the time frame and day of the week in respective fields.
- 4) Select 1 or many out of 4 options, viz. Current Traffic, Traffic history, Restaurant Suggestions and Parking Availability.
- 5) Corresponding information will be shown on the map. Also weather update for given time frame, zip code and day of the week will be shown regardless of selected service.
- 6) User can later zoom in/out and move the map using pan cursor.

User Effort Estimation

The following data shows the effort that needs to be put in by the user to get all the information required, including mouse clicks and keystrokes.

1) NAVIGATION:

The basic requirements to get traffic data of a desired destination would consist of 3 mouse clicks, as follows:

1. Open preferred web browser to load project website.
2. Click on the required service (Traffic Info or Navigation) to enter their designated pages.
3. Click "Show Traffic" to display the traffic in the selected region on entered day (after completing data entries as shown below).

The add-ons to get information about the available Parking lots and Restaurants in the region will be displayed on each of the pages and would require one click each to display the result.

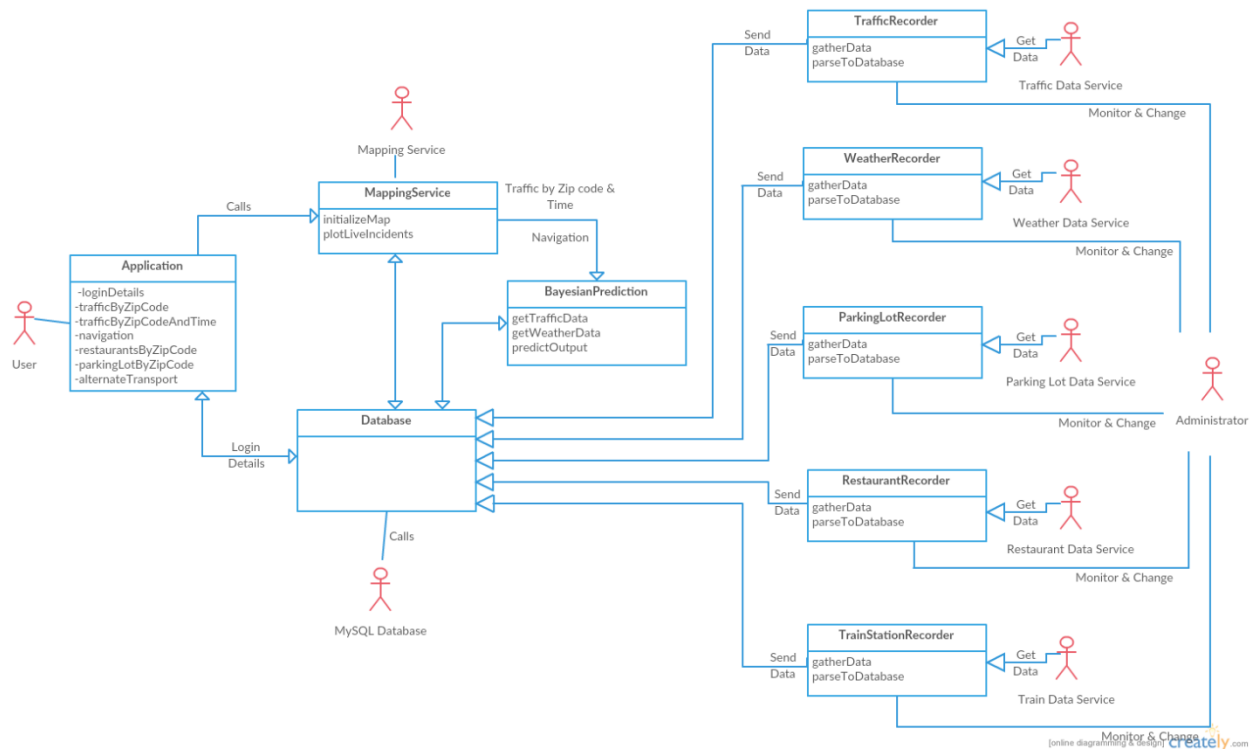
2) DATA ENTRY:

Total of 8 mouse clicks and 5 keystrokes will be required to complete a selected service.

1. Click the Text Box to type zip code of the destination to travel to (assisted by a drop down with prediction of destination name based on user input).
2. Click on the Drop down Menu to Click and select the day of travel.
3. Click on the Drop down Menu to Click and select the time frame.
4. Click the radio button to display the result of whichever service (Traffic History/Restaurant Location/Parking Lot) the user wants to see.
5. The map will be equipped with a zoom and pan functionality which will require a mouse click and drag respectively.

Domain Analysis

Domain Model



The Domain Analysis figure above shows the analysis of related software systems in a domain to find their common and variable parts. From the above diagram, it is clear that the User has access only to the Application. To keep the User experience easy and personalized, the User interface is made basic and has a login to prompt the user based on the history of how they used the application. The login details of the user are verified from the database and his most frequently searched result is prompted. Based on the input from the User, the Application calls Mapping Class. All of the functionality of the application happens through this specified Mapping Class. We are making use of two different paths thereon to provide functionality to the application. One path is where the map has to display values only by taking using entries present in the database. In this path, once the user input is got the Map is initialized and the database is invoked to provide data which is used to display the result on the map. The other path makes use of the Predictive Model to provide the best guess of the data requested by the user. Here, once the user inputs his data, the Map is initialized and then the prediction class is called to use the input and derive a result. The prediction class will invoke the database and use the current data inputs and database entries to compute the result which will be given back to the Mapping service to display.

The Administrator has control of the other features that can receive data from Web Services. The administrator will monitor the data that comes from these Services. These scripts

gather and parse the data received from the Web Services. The Administrator can use these scripts as per the requirement of the application and then pass required information to be stored in the database which in turn is used by the main web application.

To provide a personalized touch to the application, a login feature has been incorporated. This functionality allows the application to access the database at the time of login and prompt or predict the user's general choice location and time to display information based on the users data stored in the database. The application sends and receives data from the database where the User details are stored. Also, a few new features have been added keeping in mind a user who is travelling to an unknown location. The user will be able to find parking lots which are accessible to the general public based on the data that we receive from the Web service which is parsed through the ParkingLotRecorder Class and stored in our Database. The application will also provide restaurant locations at the destination location. The data of the restaurants which are got from the Web Services will also be parsed through the RestaurantRecorder Class and stored in the database. As an add-on, the application will also suggest closest train stations and a train schedules based on the ZipCode of the source and destination addresses fed in by the user. The data for the parking lots and the restaurants will be fed into the database fewer times than the weather and traffic data because it is less prone to frequent changes.

Use Case 1: Get Traffic Predictions and Weather Data

Concept Definitions

Responsibilities	Type	Concept Name
Allows user to login into the application	D	Application
Displays the predicted traffic congestion and weather data based on zip code and time	D	Application
Contains the traffic and weather data	K	Database
Predicts the future traffic congestion	D	BayesianPrediction
Initialize the map and plot the traffic congestion	D	MappingService

Association Definitions

Concept Pair name	Associated Definition	Association Name
Application - MappingService	Application sends the location address.	Plot the graph
BayesianPrediction - MappingService	BayesianPrediction sends the predicted traffic congestion on the route	Provides data

MappingService - Database	MappingService requests for the data from Database and then plots on the graph	Provides data
BayesianPrediction - Database	BayesianPrediction requests for historical data from the database.	Provides data

Attribute Definition

Concept	Attribute	Attribute Description
Application	weather	Weather of the area which user has queried
	time	Time of day
	zipCode	Zip code of source or source
Database	trafficData	Live and historical traffic data
	weatherData	Live and historical weather Data
BayesianPrediction	futuretrafficdata	Predicted congestion data
MappingService	initializeMap	Load the map based on entered zip code

Use Case 2: Navigation

Concept Definitions

Responsibilities	Type	Concept Name
Allows user to login into the application	D	Application
Displays the predicted traffic congestion and weather data based on zip code and time	D	Application
Contains the traffic and weather data	K	Database
Predicts the future traffic congestion	D	BayesianPrediction
Initialize the map and plot the traffic congestion	D	MappingService
Plot the suggested route from source to destination	D	Application

Association Definitions

Concept Pair name	Associated Definition	Association Name
Application - MappingService	Application sends the location address.	Plot the graph
BayesianPrediction - MappingService	BayesianPrediction sends the predicted traffic congestion on the route	Provides data
MappingService - Database	MappingService requests for the data from Database and then plots on the graph	Provides data
BayesianPrediction - Database	BayesianPrediction requests for historical data from the database.	Provides data

Attribute Definition

Concept	Attribute	Attribute Description
Application	weather	Weather of the area which user has queried
	time	Time of day
	zipCode	Zip code of source or source
Database	trafficData	Live and historical traffic data
	weatherData	Live and historical weather Data
BayesianPrediction	futuretrafficdata	Predicted congestion data
MappingService	InitializeMap	Load the map based on entered zip code
	suggestedRoute	Load the map based on entered zip code and plot the suggested route on the map

■ Traceability Matrix

Use Case	Weight	Application	Mapping Service	Bayesian Prediction	Database	Traffic Recorder	Weather Recorder	Parking Lot Recorder	Restaurant Recorder	Train Station Recorder
UC-1	45	X	X	X	X					
UC-2	46	X	X	X	X					
UC-3	13				X	X	X	X	X	X
UC-4	13				X	X	X	X	X	X

System Operation Contracts

1) **Operation:** Writing into database

Precondition:

Memory is allocated for different type of data in database.

Post condition:

New data is inserted in database.

2) **Operation:** Displaying dynamic data on Navigation

Pre-condition:

Traffic is predicted according to weighted algorithm in application.

Post-condition:

Detour/alternate route is suggested according to the predicted traffic.

3) **Operation:** Service request from database

Precondition:

There is enough data in the database provided by Traffic recorder, weather recorder, restaurant recorder and parking recorder.

Post condition:

This operation returns user with markers overlaying on the map. If there is no data in database, it returns nothing to the mapping devices showing only the blank map. That notifies user that there is no data available in the database for particular user request criteria.

4) **Operation:** Calculate results

Precondition:

Criteria provided by the user must be available in the database.

Post condition:

According to weighted algorithm, calculations for traffic predictions are performed and returned to the application.

Prediction Model

Prediction Strategies:

These are machine learning algorithms we use to predict the future value, when a set of past events is given.

Curve fitting is the basis of prediction. It is a process of drawing a curve that best fits with a given data set. Given a particular set of points, we draw a graph that passes through all the points or closest to all the points and passing through most of them. Thus making it possible for predicting further values. In this process it is important to carefully select the order, there is a chance that the curve may pass too far from the points if the order is too low and if the order is too high there is a chance that there are unnecessary oscillations and the prediction may not be right.

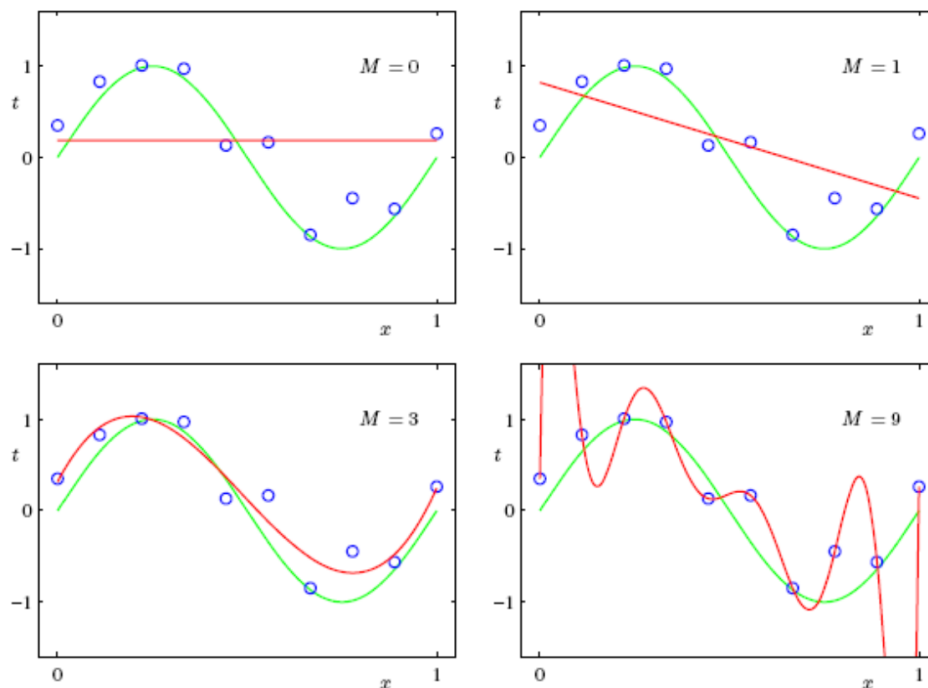


Fig: It shown the curve for different values of order M .

There are many types and strategies used to implement curve fitting. We will be using a couple of them in the project to achieve our results with the most probability of it being right. Some of the strategies used have been described below.

Bayesian Curve Fitting:

The Bayesian Curve fitting uses bayesian probability, that is we try to predict the distribution of data. We consider the previous data as training data X and t and another test point x to predict another t . So the distribution we get is $p(t|x, X, t)$.

$$p(t|x, x, t) = p(t|x, w)p(w|x, t) dw$$

Use of weighted average method in the prediction strategy, giving more weight to inputs that are more relevant to output than the ones less relevant. For example in this project taking the case of predicting the traffic at 4P.M. We give more weightage to the traffic data from previous days than that of the past hour.

In this project, we are taking two datasets in particular a) Historic data collected from a particular time and b) Live data collected in the last 1 hour. In this case when you take the case of a particular time, for example, 5P.M. is the time when people start coming home and traffic suddenly increases. If we use the live data to predict it then it may not show the traffic congestion with more precision than that of the case if we consider historical data. If we consider the historical data of 5P.M. on every working day. Then we will be able to estimate/predict precisely.

Since the historical data is more important for prediction we will be giving it more weight compared to live data. And when we are predicting for a different date since it is not possible to get the live data we just consider the historical data.

Traffic congestion	Weight
High	5
Moderate high	4
Moderate	3
Moderate low	2
Low	1

So in the equation:

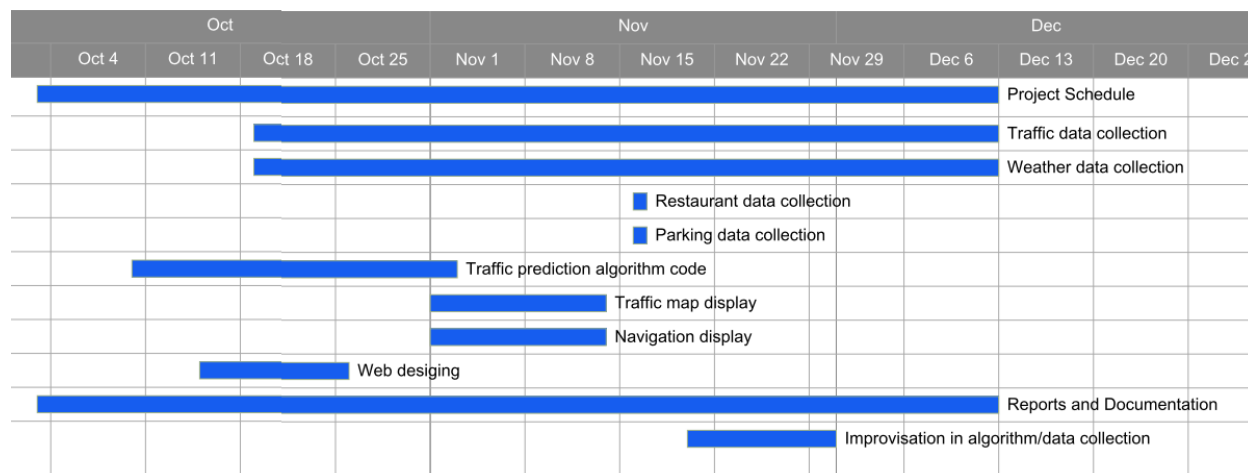
$$p(t|x, x, t) = p(t|x, w)p(w|x, t) dw$$

We take t value as the Weight of congestion and x is date/time so we predict the value of t depending on the test set, that is the future date/time. So that the value we obtain is the congestion weight at the particular time.

Plan of Work

	Task Name	Start Date	End Date	Assigned To	Duration
1	Project Schedule	10/03/15	12/12/15		71d
2	Traffic data collection	10/19/15	12/12/15	Kush, Tejas	55d
3	Weather data collection	10/19/15	12/12/15	Kush, Tejas	55d
4	Restaurant data collection	11/16/15	11/16/15	Kush, Aaditya, Tejas, Jeffrey	1d
5	Parking data collection	11/16/15	11/16/15	Kush, Aaditya, Tejas, Jeffrey	1d
6	Traffic prediction algorithm code	10/10/15	11/02/15	Everyone	24d
7	Traffic map display	11/01/15	11/13/15	Shreyas, K.Vinayak	13d
8	Navigation display	11/01/15	11/13/15	Shreyas, K.Vinayak	13d
9	Web designing	10/15/15	10/25/15	Shreyas, K.Vinayak	11d
10	Reports and Documentation	10/03/15	12/12/15	Everyone	71d
11	Improvisation in algorithm/data collection	11/20/15	11/30/15	Not assigned currently	11d

Gantt chart continued below..



Individual Responsibility Breakdown:

1) Kush and Tejas: Kush and Tejas will start working on traffic and weather data collection shortly. They will help Aaditya and Jeffrey to collect restaurant and parking information data.

2) Kisholoy and Shreyas: As mentioned earlier Kisholoy and Shreyas will be working on web development. Also they will be work on and Navigation and Traffic map display. Web designing is 50% complete.

3) Jeff and Aaditya: Jeff and Aaditya have started working on traffic prediction algorithm. Rest of the groups will be helping them in this task considering it is the most important task of the project.

Apart from this, everyone is participating in making reports. 10 days are reserved for improvisation in algorithm/data collection and additional features (if any).

References

- 1) Chen, H., Roseman, R. and Rule, A. (2015). *Curve Fitting & Multisensory Integration*. [online] <http://www.cogsci.ucsd.edu/>. Available at: http://www.cogsci.ucsd.edu/~ajyu/Teaching/Cogs202_sp14/Slides/lect3.pdf [Accessed 15 Oct. 2015].
- 2) Creately.com, (2015). *Online Diagram Software to draw Flowcharts, UML & more | Creately*. [online] Available at: <http://creately.com> [Accessed 15 Oct. 2015].
- 3) Dev.virtualearth.net, (n.d.). [online] Available at: http://dev.virtualearth.net/REST/v1/Locations?postalCode=07657&key=AvITzUaX5nqGg0JKU43_QSP30BB_ovX37MIND9WahtRbBQpeV4ugoB3HQ60LRxQL8&o=xml [Accessed Oct. 2015].
- 4) GenMyModel, (2015). *Software Modeling in the Cloud*. [online] Available at: <http://genmymodel.com> [Accessed 16 Oct. 2015].
- 5) <https://developers.google.com/maps/?hl=en>, (n.d.). *Google Maps API*. [online] Available at: <http://google.com> [Accessed Oct. 2015].
- 6) Marsic, I. (2012). *Software Engineering*.
- 7) Msdn.microsoft.com, (n.d.). *Find a Location by Address*. [online] Available at: <https://msdn.microsoft.com/en-us/library/ff701714.aspx> [Accessed Oct. 2015].
- 8) Msdn.microsoft.com, (n.d.). *Get Traffic Incidents*. [online] Available at: <https://msdn.microsoft.com/en-us/library/hh441726.aspx> [Accessed Oct. 2015].
- 9) Msdn.microsoft.com, (n.d.). *Traffic Incident Data*. [online] Available at: <https://msdn.microsoft.com/en-us/library/hh441730.aspx> [Accessed Oct. 2015].
- 10) Openweathermap.org, (n.d.). *OpenWeatherMap current weather and forecast*. [online] Available at: <http://openweathermap.org> [Accessed 4 Oct. 2015].
- 11) Project Report Spring 2011, (2011). *Project: Traffic Monitoring*. [online] Available at: <http://www.ece.rutgers.edu/~marsic/books/SE/projects/Traffic/2011-g7-report3.pdf> [Accessed 2015].

- 12) Project Report Spring 2013, (2013). *Traffic Monitoring Service*. [online] Available at: <http://www.ece.rutgers.edu/~marsic/books/SE/projects/Traffic/2013-g7-report3.pdf> [Accessed 2015].
- 13) Smartsheet, (n.d.). *Online Project Management Software | Smartsheet*. [online] Available at: <http://smartsheet.com> [Accessed 15 Oct. 2015].
- 14) Stackoverflow.com, (n.d.). *Stack Overflow*. [online] Available at: <http://stackoverflow.com/questions/33069298/bing-traffic-api-not-returning-any-data-for-accidents-delays> [Accessed 2 Oct. 2015].
- 15) W3schools.com, (2015). *W3Schools Online Web Tutorials*. [online] Available at: <http://w3schools.com> [Accessed 10 Oct. 2015].
- 16) Websequencediagrams.com, (n.d.). *WebSequenceDiagrams.com - Draw and Edit Sequence Diagrams in seconds*. [online] Available at: <http://websequencediagrams.com> [Accessed 8 Oct. 2015].
- 17) Wikipedia, (2015). *Curve fitting*. [online] Available at: https://en.wikipedia.org/wiki/Curve_fitting [Accessed 15 Oct. 2015].
- 18) Youtube.com, (n.d.). *YouTube*. [online] Available at: <http://youtube.com> [Accessed 7 Oct. 2015].