DON’T RUSH– IT’S JAM THERE

By

Naveed Ahmed

Muhammad Azeem Qamar

Kamran Riaz

Muhammad SibghatUllah Tariq

Submitted to the Faculty of Computer Software Engineering

National University of Sciences and Technology, Islamabad in partial fulfillment for

the requirements of a B.E degree in Computer Software Engineering

ABSTRACT

DON’T RUSH – IT’S JAM THERE

Traffic jam is a major problem faced by everyone in large cities all over the world. Stucking in a traffic jam costs us time and fuel. This project is to inform people about traffic jam, within specific area for specific location using Android App. Its is to help people to get to their home or office without being stuck into traffic jam.

This system will facilitate user to get information about traffic congestion before they start their journey. The System will have a video feed from roadside camera installed. The process of congestion detection is completed through processing of this video. Process is completed through Haar cascades algorithm that calculates count and speed of vehicle. After the processing of video Server saves the result (whether jam or not) in database. The Server will also be connected to User through an Android App. Server gets location of user through GPS. User will also have the facility to give destination and get congestion information on the route. Any time User may ask for congestion information, the information will be extracted from database and then sent to that user. In case of traffic jam Android App will have the facility to give alternate route to destination of User.

In order to view the real-time processing on Server, there is a desktop application and server side is developed using C# and Visual Studio for GUI. This application is compatible with Windows 7 and latest. The application will show the processing and it will be connected to database. The android based application shall be an android form based java application, which shall provide a graphical user interface for user friendly environment.For communication between Server and Android App, Sockets protocols will be used. Each Android device will be identified by its unique MAC address. The android app uses Google maps to show location or route for user. The target of this project is to inform app users (general public) about traffic jam proivding an alternate path thus developing a cost effective solution for traffic jam in existing system.

CERTIFICATE FOR CORRECTNESS AND APPROVAL

Certified that work contained in the thesis – Don’t rush its jam there carried out by Naveed Ahmed, Azeem Qamar, Muhammad SibghatUllah Tariq and Kamran Riaz under supervision of Dr. Naima Altaf for partial fulfillment of Degree of Bachelor of Software Engineering is correct and approved.

Approved by

Dr. Naima Altaf

CSE DEPARTMENT

MCS

DATED:

_____

DECLARATION

No portion of work presented in the dissertation
has been submitted in support of another award or qualification either at this institution or elsewhere.

DEDICATION

In the name of Allah, the most beneficent and merciful

To our parents and teachers without whose support and cooperation

this project would not have been possible.

ACKNOWLEDGMENTS

We would genuinely like to thank our project supervisor Dr. Naima Altaf for her gaudiness and continuous support. She is an exception supervisor. We couldn’t have come this far without her support and help.

This would not have been completed without the support of Dr. MohsinRiaz and Sir Saleh Usman form COMSATS. He gave tips and full-filled requirements we needed to complete this project. Sir Thank You very much.

TABLE OF CONTENTS

LIST OF FIGURES

1 Chapter 1: Introduction

1.1 Problem Statement

Today the world is seeing an energy crisis. Traffic jam is a major source of energy wastage as cars burn fuel and waste time, money and energy. We need a cost effective solution for traffic jam. Using modern technology and internet we have worked out a working and effective solution for users to save themselves from the jam. This will save general public from wastage of time and money.

1.2 Scope

The project consists of two parts:

1. Android App named as ‘Traffic Sense Android’, will get User’s location using GPS and send it to Server, User will give his destination or route to get information about the clearance of route. Server cantrigger pop-up about congestion situation in near-by area of Users.
2. Desktop App or Server named as ‘Traffic Sense Desktop’ gets video from roadside camera and starts processing it according to way of detection of traffic congestion. Server can also detect fog information from this video and save the result to database.

1.3 Objectives

The objectives of the project include:

1. Develop a cost effective solution for traffic jam in existing system.
2. Get video of roadside and extract information for it.
3. Make extracted information useful in a sense that it gives congestion and fog information.
4. Inform people i.e. Users about traffic status using an Android App.
5. Establish a connection with Users of Android App.

1.4 Deliverables

The outcome of the project will be a fully functional system that will be able to detection traffic jam form a video feed at desktop side and for Users, there will be and Android application which will be responsible for updating user about current traffic situation in near-by location.

2 Chapter 2: Literature Review

2.1 Introduction

Traffic jam is a major problem faced by everyone in large cities all over the world. Stucking in a traffic jam costs us time and fuel. This project is to inform people about traffic jam, within specific area for specific location using Android App. Its is to help people to get to their home or office without being stuck into traffic jam.

This system will facilitate user to get information about traffic congestion before they start their journey. The System will have a video feed from roadside camera installed. The process of congestion detection is completed through processing of this video. Process is completed through Haar cascades algorithm that calculates count and speed of vehicle. After the processing of video Server saves the result (whether jam or not) in database. The Server will also be connected to User through an Android App. Server gets location of user through GPS. User will also have the facility to give destination and get congestion information on the route. Any time User may ask for congestion information, the information will be extracted from database and then sent to that user. In case of traffic jam Android App will have the facility to give alternate route to destination of User.

2.2 Related Work

ClearFlow:

> ClearFlow is software designed by Microsoft. Microsoft says that it gives driver an alternative router information that is more accurate and attuned to current traffic patterns on both freeways and side streets. This system is avialable for 72 cities in United States. It uses Artifical Intelligence techiniques to predict the flow of traffic.

2.3 Proposed Project

Our project will extract informations from the video and decide wthethere there is traffic jam or not based of some parameters like speed and count of vehicles. Within specific area for specific location using Android App. Its is to help people to get to their home or office without being stuck into traffic jam.The target of this project is to inform app users (general public) about

traffic jam proivding an alternate path thus developing a cost effective solution for traffic jam in existing system.The system gets a video of a road from specific location where camera is installed and send this video to server for processing. Server will process the video using image processing techniques and detect traffic jam by calculating some parameters like vechicle detection, vehicle conjestion and movement detection. These parameters will help in deciding either traffic is jam or not. Our project includes:

- Desktop app
- Android app

3 Chapter 2: Software Requirement Specification

This section introduces the Software Requirement Specification for the Don't Rush – Its Jam There (Traffic jam detection based on image processing) to its readers. The aim of this document is to gather, analyze and provide an in-depth insight of the system, by defining the problem statement in detail.

3.1 Overall Description

3.1.1 Product Function

The product is aimed to provide information about traffic flow to the User through an Android App.

Traffic flow is determined from a video of roadside from camera. The process of traffic flow detection is completed through processing of this video. Process is completed through some algorithms that calculate count and speed of vehicle. The Server that processes video will save result and send a pop-up message to Users near-by. Server also extracts fog information from the video.

There is also an Android App for targeted Users. Android App will be connected to Server. Server gets location of Users through GPS. Users can get near-by traffic information by-default or by providing destination and asking about information for that route.

The plan to carry out this project consists of three main tasks:

1. Make an algorithm to get vehicle congestion and fog information from the video.
2. Develop Traffic Sense for Desktop and Android.
3. Interfacing Desktop and Android App.

3.1.2 Product Features

Desktop App:

1. Get Video Feed
2. Frame extraction and enhancement

3. Vehicle detection
4. Congestion Calculation
5. Fog detection
6. Save result in Database

Android App:

7. Establish connection with User
8. Get location through GPS
9. Request and Response
10. Alternate path

3.2 External Interface Requirement

3.2.1 User Interfaces

Firstly, in order to view the real-time processing on Server, there shall be a desktop application based on .NET framework. This application shall be compatible with Windows 7 and latest.The application will show the processing and it will be connected to database.

Secondly, the android based application shall be an android form based java application, which shall provide a graphical user interface for user friendly environment. All the data entry shall be done with keypad and the user can navigate between menus/forms/screens with the help of navigation keys.Finally, the user interface for the android application of the System, shall be compatible to all android devise but for best user experience the following versions are preferable

1. Jelly beans 4.1.2
2. ICS 4.1.1

3.2.2 Software Interfaces

Server side of system shall be installed on a PC with Windows 7 or later.Android App will be installed on android device with android version ICS or later.Server side shall be developed using C# and Visual Studio for GUI.For this system, following API and external libraries will be used:

1. Matlab assemblies
2. Google APIs
3. Android GPS APIs

Android App shall be accessible on Server, connected through internet connection.

3.2.3 Communication Interfaces

For communication between Server and Android App, Sockets protocols will be used. Each Android device will be identified by its unique MAC address.The android application shall use Google Maps to show location or route for User.

3.3 System Features

3.3.1 Get Video Feed

Table 2-1: System Feature: Get video feed

Description	Server will get a video of roadside. A video that will be used for analysis and further processing.
Priority	High
Pre-Conditions	Serer should be ready to get video.
Stimulus/Response	
Post-Conditions	Video is available in MS Directs how supported format.
Risk	Medium

3.3.1.1 Functional Requirements

REQ-1 Server will automatically get video feed.

REQ-2 Video length must be greater than 40 seconds.

REQ-3 Video must be from roadside view.

3.3.2 Frame extraction and enhancement

Table 2-2: System Feature: Frame extraction and enhancement

Description	Frames are extracted from video it receives and then enhanced to remove noise and improve quality of image.
Priority	Medium
Pre-Conditions	Video should be in MS Directshow supported format.
Stimulus/Response	Frames are extracted and saved to directory on server. Noise is removed and smoothing filters are applied using matlab on extracted frames.
Post-Conditions	Frames are available for further processing.
Risk	Low

3.3.2.1 Functional Requirements:

REQ-4 Server shall automatically start extract frames from video.

REQ-5 Every frame extract must be in sequence and have specified time difference.

3.3.3 Vehicle Detection

Table 2-3: System Feature: Vehicle detection

Description	Image processing is applied to extracted frames to detect vehicles and vehicle count. A region of interest is defined for this purpose. This process is applied on every extracted frame of a particular video.
Priority	High

Pre-Conditions	
Stimulus/Response	System will get frames one by one and apply the process on each frame. Information extracted will be saved separately such as vehicle count, vehicle per frame, total number of vehicles, position of every vehicle.
Post-Conditions	Speed of every vehicle and average speed of all vehicles is calculated.
Risk	High

3.3.3.1 Functional Requirements:

REQ-6 System shall automatically start detection process as per frames are ready.

REQ-7 System shall detect vehicles from region of interest.

3.3.4 Congestion Calculation

Table 2-4: System Feature: Congestion calculation

Description	System will calculate congestion parameters and decide whether there is a traffic jam or not and save the result to database.
Priority	High
Pre-Conditions	Count of Vehicles, Speed of Vehicles
Stimulus/Response	Using count of vehicles, total number of vehicles, speed of vehicles and time, system calculates that if there is enough congestion to declare jam or not.
Post-Conditions	Result of this processing is saved to database with current time and date.
Risk	Medium

3.3.4.1 Functional Requirements:

REQ-8 System shall automatically calculate congestion with given formula.

REQ-9 If vehicle count detected is zero, system shall not crash or generate error.

REQ-10 Information will be stored in database.

3.3.5 Establish Connection

Table 2-5: System Feature: Establish connection

Description	Using Android App, User needs to connect to server to get congestion and fog information.
Priority	High
Pre-Condition	Server must be ready to accept connection.
Stimulus/Response	When User starts the application, application will connection to Server on specified IP and Port.
Assumption	User has an active internet connection.
Post-Conditions	App is connection to Server.
Risk	Low

3.3.5.1 Functional Requirements:

REQ-11 Android App shall automatically connect to server on start-up.

3.3.6 Acquire User Location

Table 2-6: System Feature: Acquire user location

Description	Application acquires the location of User to get information ready for near-by areas of User.
Priority	Medium
Pre-Condition	Application is connected to remote Server.

Stimulus/Response	When the App is connected to Server, Server request for its location, the App finds its location using GPS and return to Server.
Post-Conditions	Location of User is available of Map in Latitude and Longitude.
Risk	Low

3.3.6.1 Functional Requirements:

REQ-12 App shall send User location automatically when it is connected to Server.

3.3.7 Request and Response

Table 2-7: System Feature: Request and Response

Description	App User can send a request for any route information, if Server has that route information, it will respond, otherwise it will generate an error.
Priority	Medium
Pre-Condition	User location and destination
Normal Course	User will get near-by information if he is subscribed for updates.
Post-Condition	User will have information on Map.
Alternate Course	User can request for information with his destination.
Post-Condition	User will get have route information on Map, and also alternate route.
Risk	Low

3.3.7.1 Functional Requirements:

REQ-13 User will send his destination using Google Maps.

REQ-14 On response from Server, App will show a pop-up.

3.3.8 Show Alternate Path

Table 2-8: System Feature: Show alternate path

Description	In case of traffic jam and congestion, App will give an alternate path to the User to get to their destination.
Priority	Low
Pre-Condition	App has requested for information to server.
Stimulus/Response	If response from Server shows that there is any congestion or traffic jam, then App will show a Map with all alternate paths available.
Post-Condition	
Risk	Low

3.3.8.1 Functional Requirements:

REQ-15 System shall give shortest available alternate path to the User.

3.4 Other Non-Functional Requirements

3.4.1 Performance Requirements

Response Time

Server shall receive video after very one hour. Response time of Server on request from Android App must not exceed 30 sec. Android App should not take more than 5sec in getting User location and send it to Server.

Throughput

Server must process every video within 5-10 minutes.

Concurrency

A minimum of 5 concurrent Android App Users and 1 active Server as Main User must support about base lined response and throughput.

Reliability

Failure rate of application/transaction must not exceed 5 in 10,000. MTBF (Mean Time Between Failure) for server should be 1 week at minimum within 6 month of deployment. However after 6 months of deployment MTBF must be 1 month. MTRS (Mean Time to Restore Service) must be 2 hours max for critical, 1 hour for major and 30 minutes for minor faults.

3.4.2 Safety Requirements

Integrity of Information

In case of hardware failure (i.e. PC has crashed or stopped working) or a hardware component has stopped responding (for at least 2 minutes) an appropriate error (TBD) should be generated and displayed to main Server Application.

Backup & Recovery

Backups of database shall be maintained automatically every day at least once. I case of Server failure or database or application gets corrupted, it should not take more than 1 hour to install the application again and load the backup database.

3.5 Security Requirements

Data Transfer

The system shall use secure sockets in all transactions that include any confidential user information.

3.5.1 Software Quality Attributes

The application shall be scalable for maximum possible Users. Data of locations older than 5 years shall be discarded for better system performance.

Flexibility

System shall be flexible enough to cater future modifications such as prediction of jam, etc. System shall be modifiable if more features need to be added.

Usability

The system shall provide a uniform look in all application pages. The system shall provide colored GUI for better visualization. The Map displayed to App User shall be a colored one and different colors shall be used to highlight route and location. The android application shall provide colored icons and menu bars for navigation and better visualization and user experience.

Accessibility

Navigation between different menus and screens shall also be possible using keyboard commands and shortcuts. The system shall only provide single language support i.e. English.

Availability

The system shall be available and online 24/7, with no more than 3 hours of downtime per week for maintenance. There should be a contractual agreement with an internet service provider for T3 access with 99.99% availability for the main server.

Reliability

There should be an alternate android device, such that incase of failure, within a time span of 2 hours alternate android device should be up and operational. Backup power source (external battery) shall be provided in order to ensure maximum availability of the system.

4 Chapter 3: Design and Development

4.1 Introduction

This chapter provides clear description of the functionality and design of the project Don't Rush - its Jam There. Detailed architecture is provided in this chapter. The design is further elaborated by using diagrammatical representation of system components, classes, states, and sequence of events, flow of events and their relationships.

4.2 3.2 Overview

This chapter of document has described architectural design of Don't Rush-Its Jam There App. The high level components and their interactions, suitable architectural pattern and design decisions applied to the whole system.

The final chapter of the System design specification is on component and detailed design. It includes design patterns, Sequence diagrams, class diagrams, activity diagram, state transition diagram, use case diagrams and user interface diagram.

4.3 System Over

This system consists of two module, the server module and client module. Details of these modules are given below.

4.3.1 Server Module

The server-side is a desktop application. It is responsible for video processing, extracting the required information from video and then save this information in database. It is also responsible for sending this information to clients on request or automatically.

4.3.2 Client Module

The client-side is an android application which request for information from server and show it to user of the app. It may give an alternate route on request by user.

4.4 3.4 Design Consideration

This section contains all of the assumptions, dependencies, and constraints that were considered during the design of each subsystem and component.

4.4.1 Assumptions and Dependencies

The project is based on following assumptions:

1. Android based mobile will be available.
2. Windows PC will be available.
3. Internet connection will be available for the connection between mobile and computer.
4. User will be familiar with the android application.

4.4.2 Constraints

Our system requires a small bandwidth. So, it can work with any internet connection, slow or fast. There will be parallel processing and clients handling on server-side, so a good PC will be required with at least three cores.

4.5 System Architecture and Design

4.5.1 System Architecture

This section provides a detailed and comprehensive architectural overview of the system. Client server pattern is followed in which Mobile acts as a client and sends connections request on TCP server on PC.

4.6 Design Details

4.6.1 Design Pattern

The design pattern used here is **Facade Pattern**. It provides a unified interface to a set of interfaces in a subsystem and defines a higher-level interface that makes the subsystem easier to use. A facade exposes simplified functions that are mostly called and the implementation conceals the complexity that clients would otherwise have to deal with. In general the implementation uses multiple packages, classes and function there in. Well written facades make direct access of other classes rare.

4.6.2 Flow chart Diagram

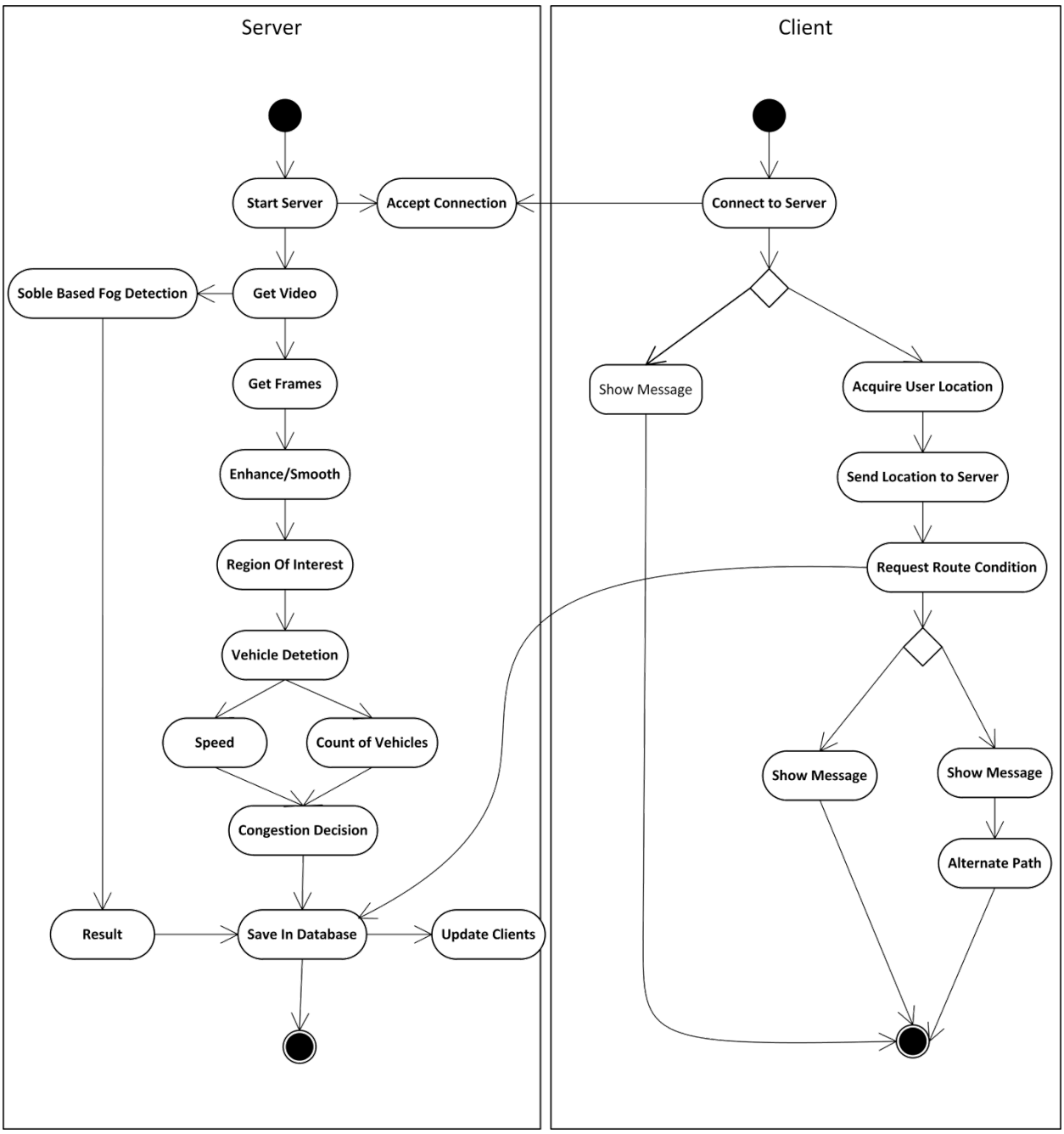

Figure 1Flow Chart Diagram

4.7 Use Case Diagrams

4.7.1 Video Feed Use Case diagram

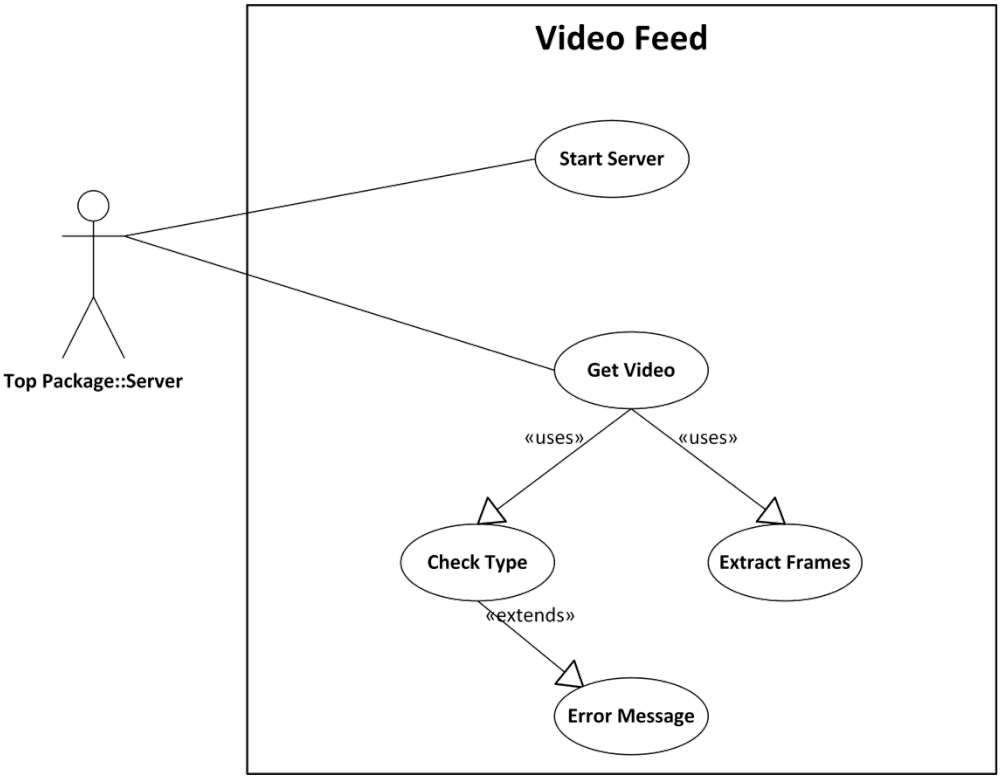

Figure 2Use Case Diagram: Video Feed

4.7.1.1 ***3.7.1.1 Use Case Description***

This use case describes how to the server starts. How the server will get the video feed and then it will check the type of video. This is how it will check whether video is capable of being used for processing or not. If video is of our required quality, frames will be extracted from the video. If the video is not of our required quality then an error message will be displayed.

4.7.1.2 ***Actors***

Server

4.7.1.3 ***Pre-Conditions***

Server should be ready to get video.

4.7.1.4 ***Basic Flow of Events***

1. Server starts
2. Video feed is given to server
3. Server checks the type of video for father processing.
4. Frames are extracted from video.
5.

4.7.1.5 ***Alternate Flow***

If the video is not of required type and quality then an error message is generated.

4.7.1.6 ***Post Condition***

Video is successfully provided to server and frames are extracted from the video for further processing.

4.7.2 **Congestion Calculation Use Case Diagram**

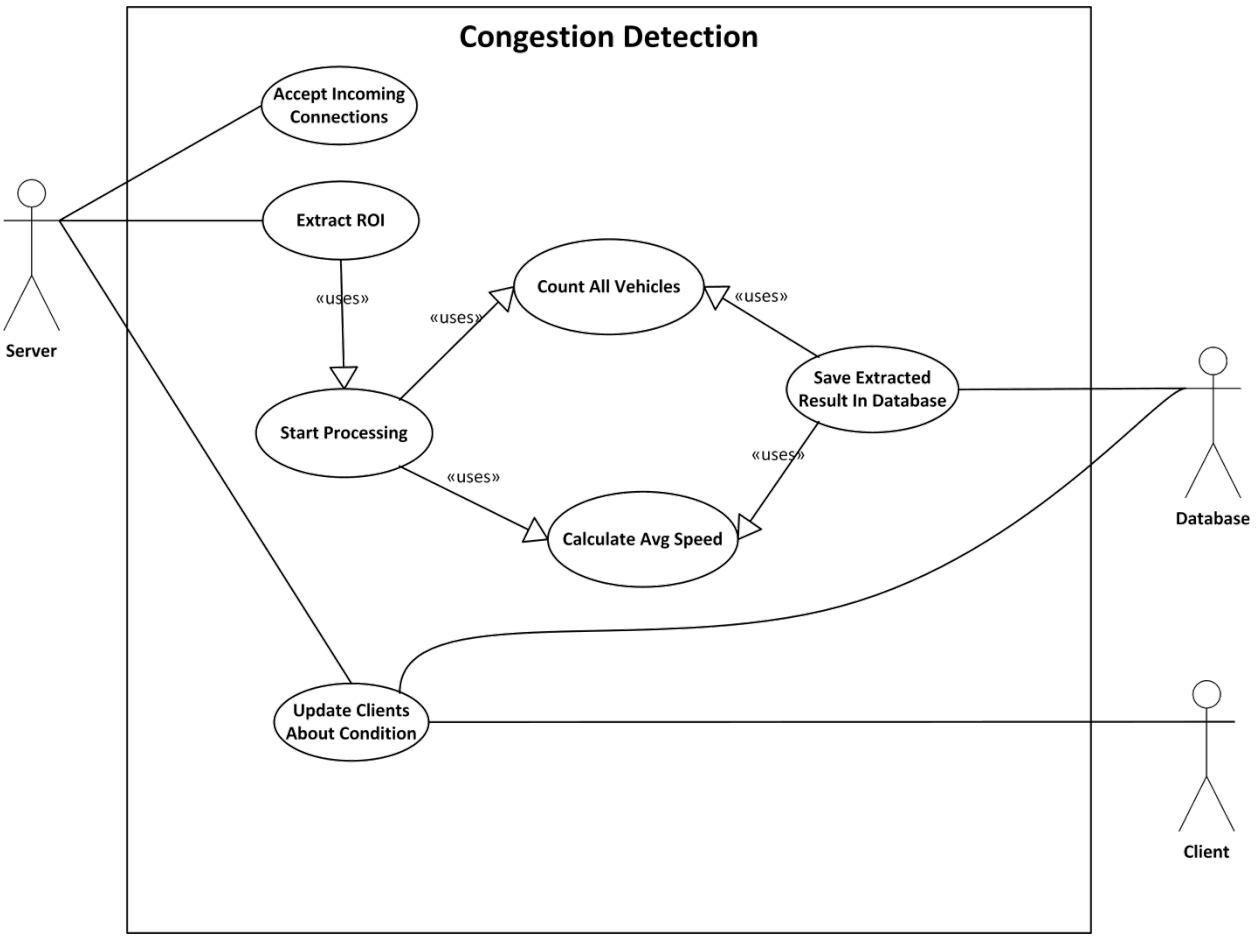

Figure 3Use Case Diagram: Congestion calculation

*4.7.2.1 **Use Case Description***

This use case diagram describes how the clients will connect to the server for getting information about the traffic situation of their desired path. Server will extract region of interest from the extracted frames of video. After extracting region of interest the server will further do processing and calculated the number of vehicle there on the road. It will also calculate the average speed of the vehicles on the road. I this way it will get to a decision point and save the decision information about traffic situation in the database. When user will ask about traffic situation, server will check in database about that particular location, and then server will send information to the user.

4.7.2.2 ***Actors***

1. Server
2. App User
3. Database

4.7.2.3 ***Pre-Conditions***

1. User must have android application installed on his device.
2. Count of vehicles should be available.
3. Speed of every vehicle and average speed of all vehicles should also be available.

4.7.2.4 ***Basic Flow of Event***

1. Start Processing of Video.
2. Detect and count vehicles.
3. Count total number of vehicles in video and save them.
4. Calculate speed of vehicle in video and calculate average speed of all of them.
5. Decide congestion based on speed and number of vehicles.
6. Server sends the information to all clients.

4.7.2.5 ***Alternate Flow***

1. The server is already busy and is in processing. The server will not accept video feed.
2. There is no video feed available.

4.7.2.6 ***Post Conditions***

Based on congestion decision parameters it is decided that whether there is a jam or not.

4.7.3 Establish Connection and Get Location Use Case Diagram

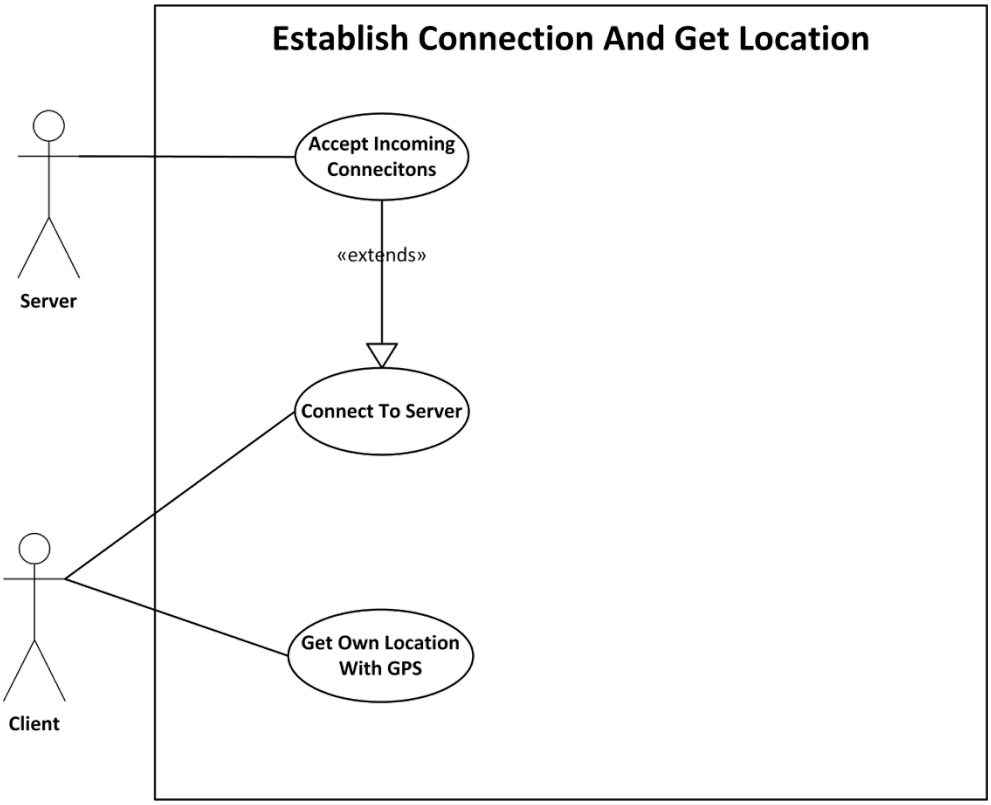

Figure 4Use Case Diagram:Establish Connection and Get Location

4.7.3.1 ***Use Case Description***

The client has to connect to server using Android app to obtain the information regarding traffic and fog. When user starts the app it will connect to server automatically on specified IP and port. When app is connected to server, server requests for its location and the app acquires the location of the user using GPS and sends the server its location.

4.7.3.2 ***Actors***

1. Server
2. Client (App user)

4.7.3.3 Pre-Conditions

1. Server must be ready to accept connection.
2. Application is connected to server to get location.

4.7.3.4 Basic Flow of Event

1. User starts the Android app.
2. App makes the connection with server.
3. App finds its location using GPS.
4. App sends its location to server.

4.7.3.5 Alternate Flow

Error message is displayed on connection failure.

4.7.3.6 Post Conditions

1. Connection is made with server
2. Location of user is sent to server.
3. Server starts gathering information regarding that location.

4.7.4 Request and Response Use Case Diagram

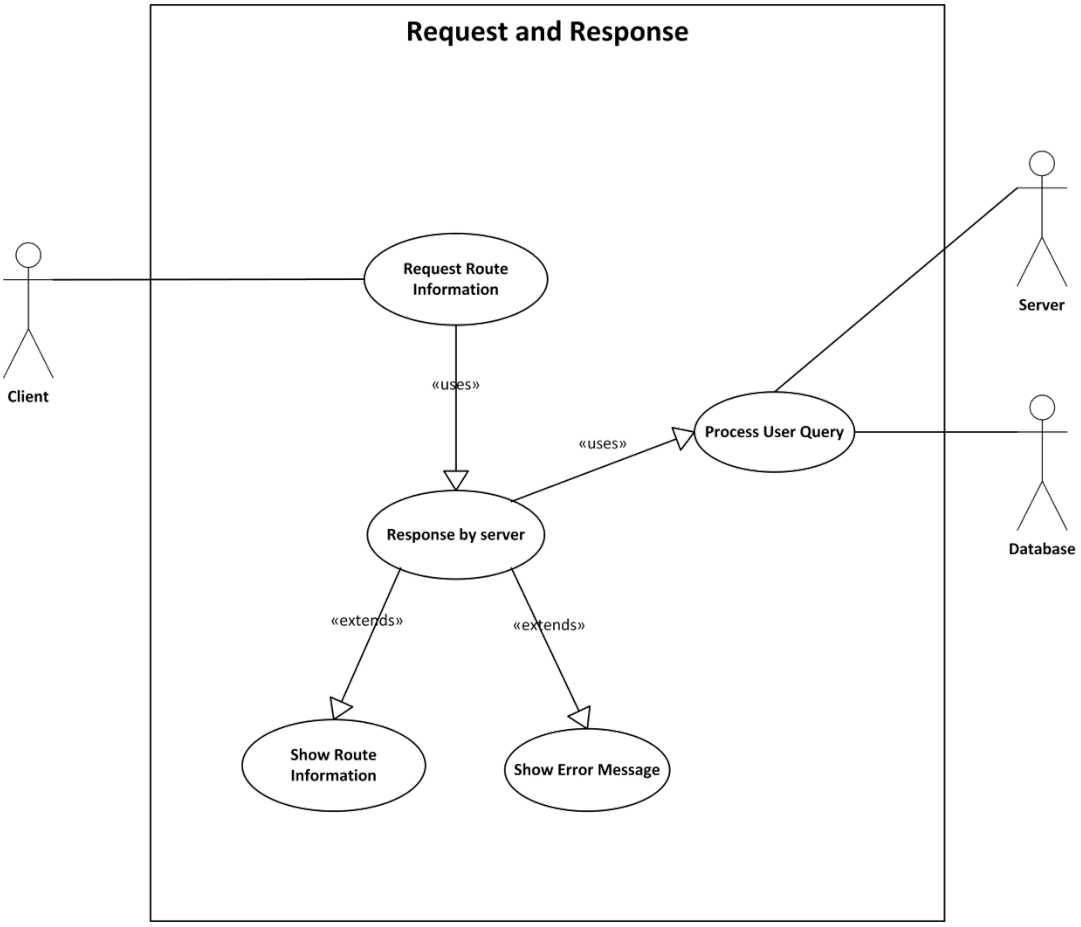

Figure 5Use Case Diagram: Request and Response

4.7.4.1 Use Case Description

App user can send a request for any route information; if server has information regarding that route then it will respond with that otherwise it will generate an error message.

4.7.4.2 Actors

1. Server
2. Database
3. Client

4.7.4.3 ***Pre-Conditions***

Server has the information of the location and destination of the user.

4.7.4.4 ***Basic Flow of Events***

1. User sets its source and destination.
2. App request information to server.
3. Server sends response.
4. App acts accordingly to the response.

4.7.4.5 ***Alternate Flow***

1. Server cannot get the route information and generates an error message.
2. Connection between app and server is lost.

4.7.4.6 Post Conditions

User will have information about the route.

4.7.5 Alternate Route Use Case Diagram

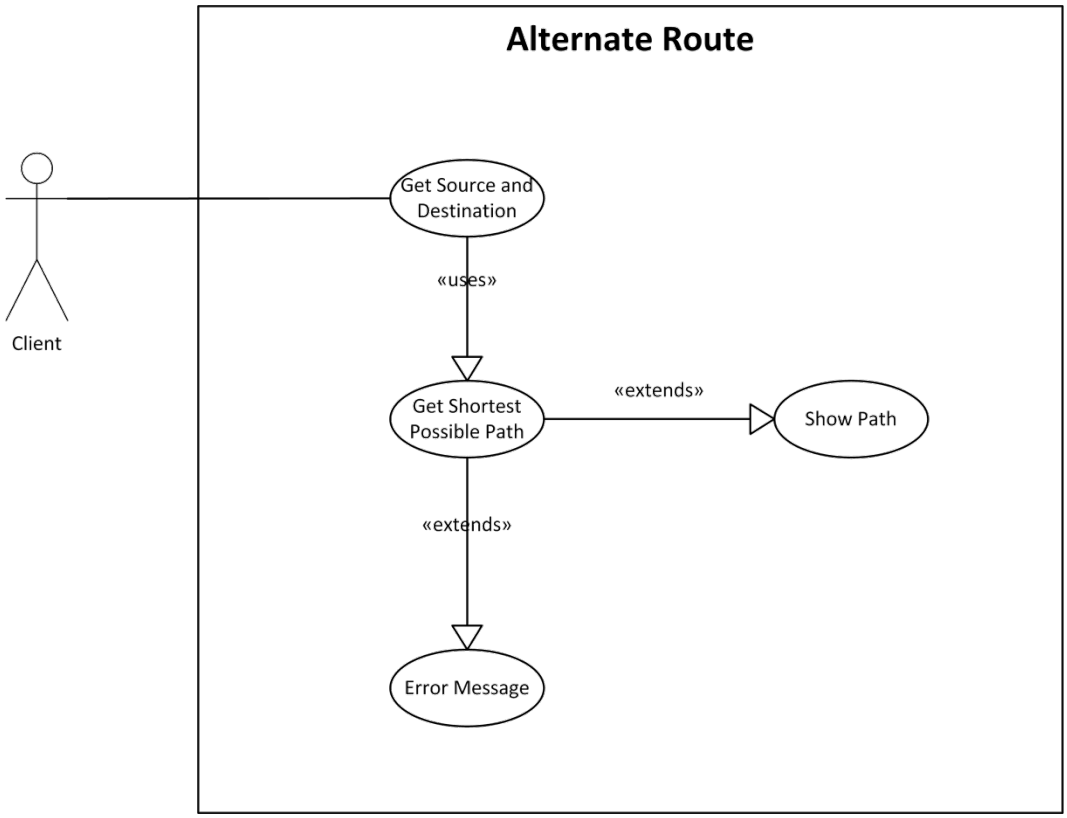

Figure 6Use Case Diagram: Alternate Route

4.7.5.1 ***Use Case Description***

In case of traffic jam and congestion app user will give an alternate path to the user to get to their destination. If there is no alternate route available the app will generate an error message.

4.7.5.2 Actors

Client

4.7.5.3 Pre-Conditions

The Android app has requested for the alternate route to server.

4.7.5.4 Basic Flow of Events

1. User sets its source and destination.
2. App request for information of route.
3. App shows an alternate route in-case of heavy congestion or fog.
4. If alternate route is not available the app shows an error message.

4.7.5.5 Alternate Flow

1. Alternate route is not available, app generates error message
2. The connection between app and server is lost.

4.7.5.6 Post Conditions

The app user has the information about the alternate route to destination.

4.8 Sequence Diagrams

4.8.1 Video Feed Sequence Diagram

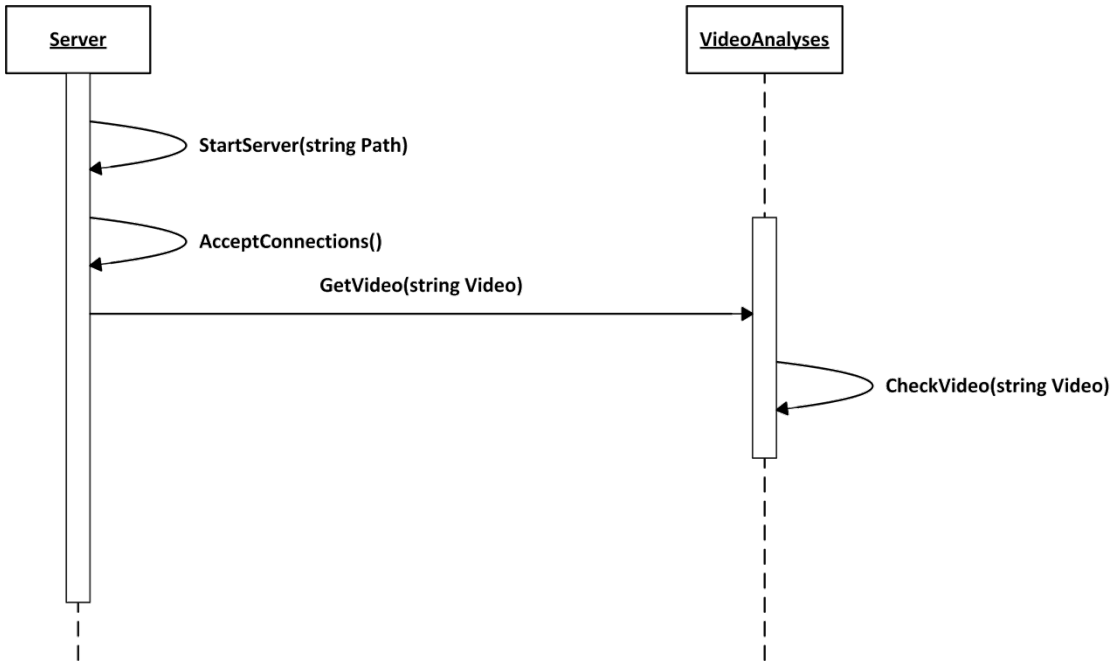

Figure 7Sequence Diagram: Video Feed

4.8.2 Congestion Calculation Sequence Diagram

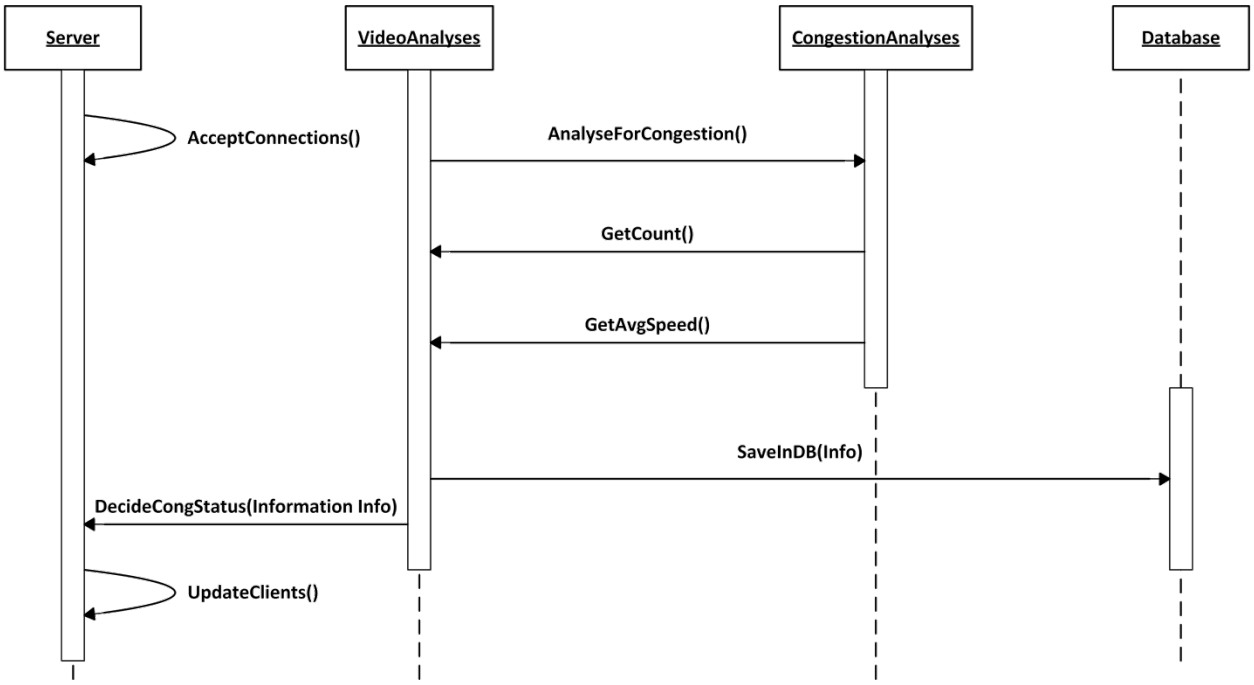

Figure 8Sequence Diagram: Congestion Calculation

4.8.3 Establish Connection and Get Location Sequence Diagram

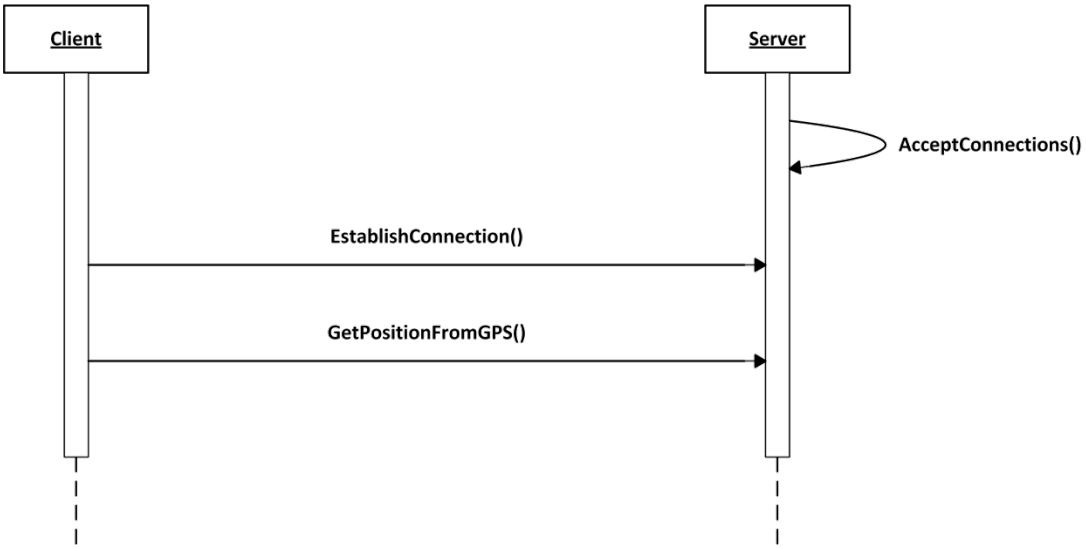

Figure 9Sequence Diagram:Establish Connection and Get Location

4.8.4 Request and Response Sequence Diagram

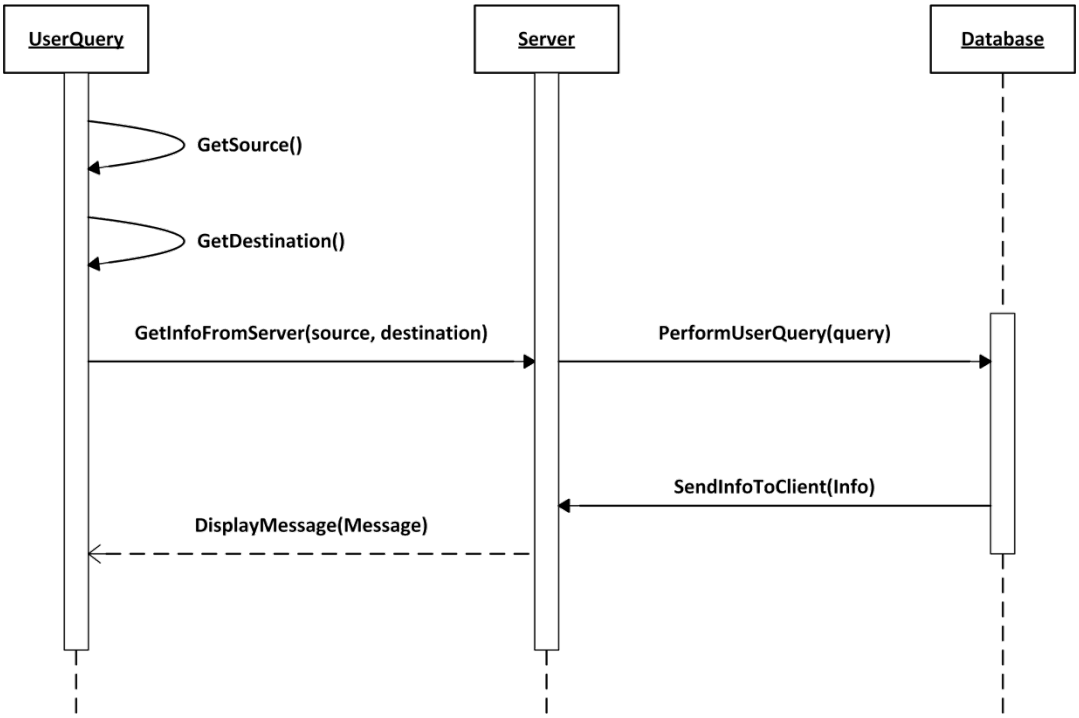

Figure 10Sequence Diagram: Request and Response

3.8.3 Alternate Route Sequence Diagram

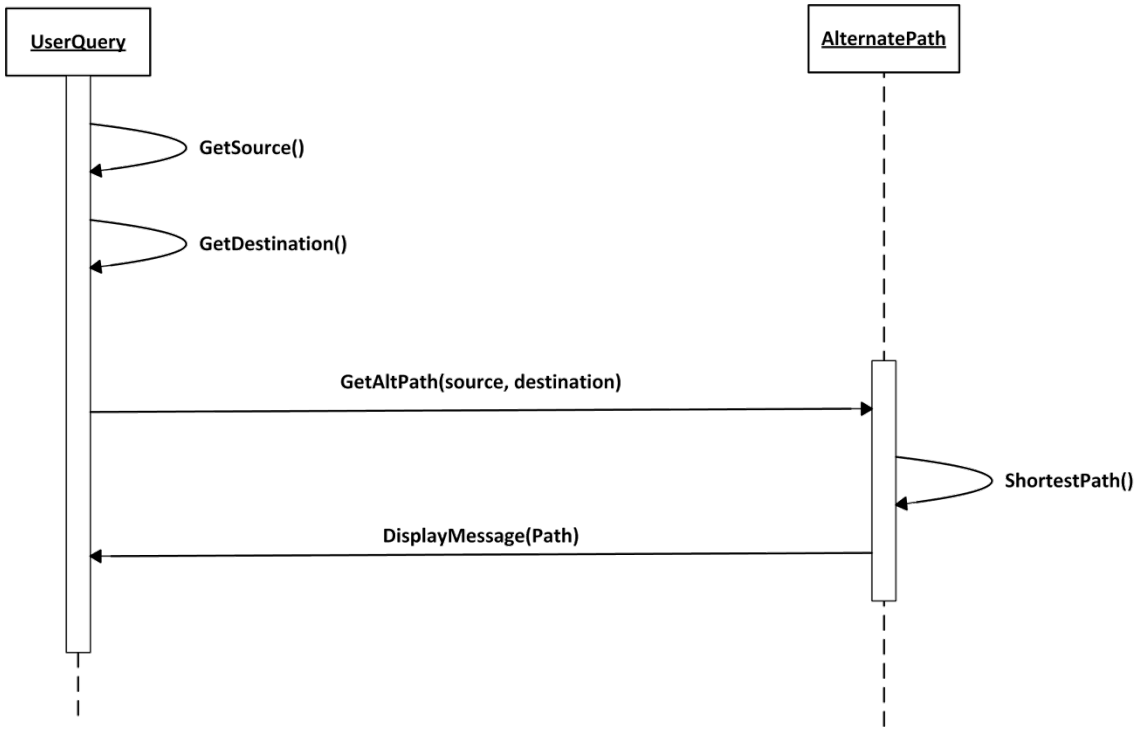

Figure 11Sequence Diagram: Alternate route

4.9 Implementation View

4.9.1 System Class Diagram

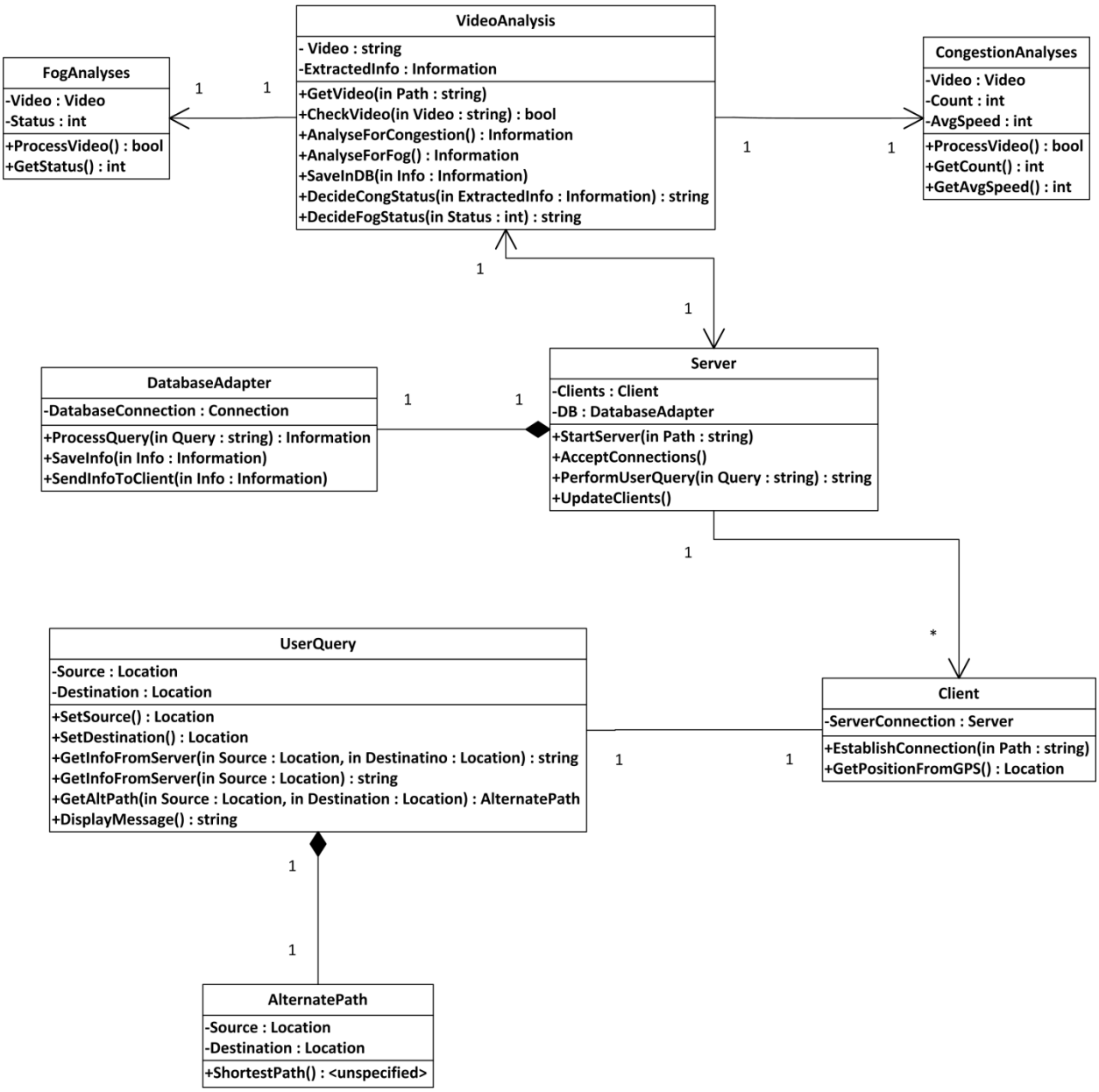

Figure 12System Class Diagram

4.10 System Classes Description

4.10.1 Server Class

Table 3-1: Server class

Name	Server
Description	This class holds the server side. It communicates with clients and processes their query. This class is responsible for starting server and connecting with Client.
Functions	**StartServer(Path)**: It start the server on given path or address. **AcceptConnections()**: It start a thread which accepts the incoming connection request by client. **PerfromUserQuery(Query)**: It process the any query sent from client. **UpdateClients()**: It updates near-by clients about the condition on road.

4.10.2 VideoAnalyses Class

Table 3-2: VidepAnalyses class

Name	VideoAnalyses
Description	This class is responsible for processing a video which is given in input. It saves the results from video in a database. This class decides the condition on road by getting the result from video analyses.
Functions	**GetVideo(Video)**: This functions gets a video input. **CheckVideo(Video)**: This function checks whether the video format if known to program for processing purpose. **AnalyseForCongestion()**: It analyze for congestion information. **AnalyseForFog()**: It analyze for fog information.

	SaveInDB(Info): It save extracted information from video in database. **DecideCongSatus(Info)**: It decides the congestion status by using the extracted information. **DecideFogStatus(Status)**: It decides the fog status by using the extracted information.

4.10.3 CongestionAnalyses Class

Table 3-3: CongestionAnalyses class

Name	CongestionAnalyses
Description	This class performs congestion analyses on video. It extracts information from video such as number of vehicles in the video and average speed of vehicles.
Functions	**ProcessVideo()**: This function is responsible for processing of video. It extracts information from the video. **GetCount()**: This function returns the number of vehicles discovered in the video. **GetSpeed()**: This function returns the average speed of vehicles.

4.11 DatabaseAdapater Class

Table 3-4: DatabaseAdapter class

Name	DatabaseAdapter
Description	This class holds the database. It performs query on the database and return their results. It saves the results in database and extracts when needed.
Functions	**ProcessQuery(Query)**: It processes the query and returns the result. **SaveInfo(Info)**: It saves information in database. **SendInfoToClient(Info)**: It send information to client when requested.

4.11.1 Client Class

Table 3-5: Client class

Name	Client
Description	This class holds the client-side and is responsible for establishing connection with server. On connection it gets location using GPS and send it to server.
Functions	**EstablishConnection(Path)**: It established the connection with server on given path or address and if fails it generate an error message. **GetPositionFromGPS()**: It gets location of client/Android Device using GPS and sends it to server.

4.11.2 UserQuery Class

Table 3-6: UserQuery class

Name	UserQuery
Description	This class request server for any query by user of Andorid App. It holds the source location and destination location of user and may request the server for condition of the route.
Functions	**SetSource()**: This function sets the source location of User. **SetDestination()**: This function sets destination location of User. **GetInfoFromServer(Source,Destination)**: This function gets information about route condition from server between source and destination. **GetInfoFromServer(Srouce)**: This function get information from server about route condition only in source area. **GetAltPath(Source,Destination)**: It gets alternate path between route and destination. **DisplayMessage(Message)**: It display message return from three above functions.

4.11.3 AlternatePath Class

Table 3-7: AlternatePath class

Name	AlternatePath
Description	This class gives path between two points. It holds the source and destination location and returns the shortest possible path.
Functions	**ShortestPath()**: This function return the shortest possible path between source and desitnation.

4.12 User Interface Design

User Interface consists of two applications, one desktop application and other android application.

4.12.1 Main Window:

This window shows the interface of Desktop Application of the Software. It plays the video to be processed in it. 'Process' Button asks for a video input that needs to be processed.

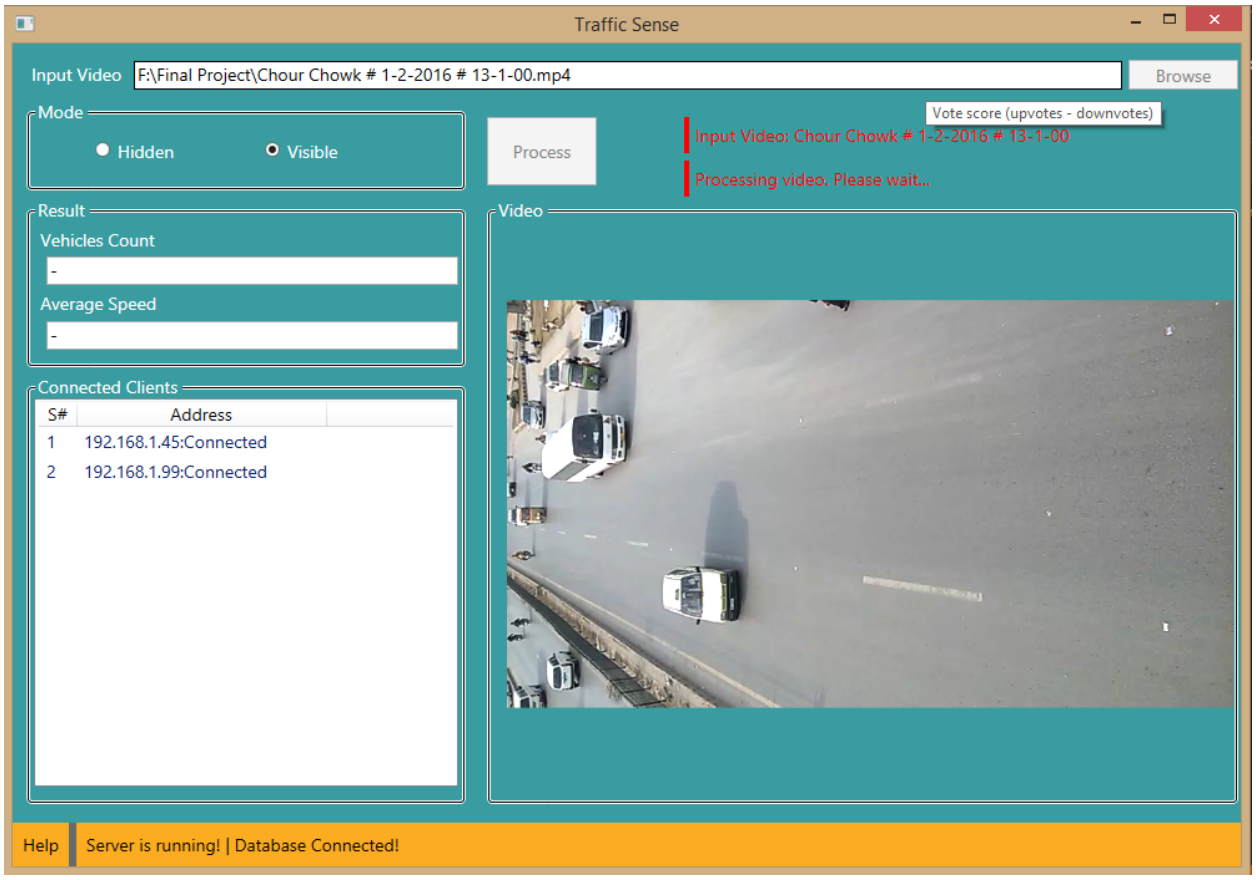

Figure 13Main Desktop Interface

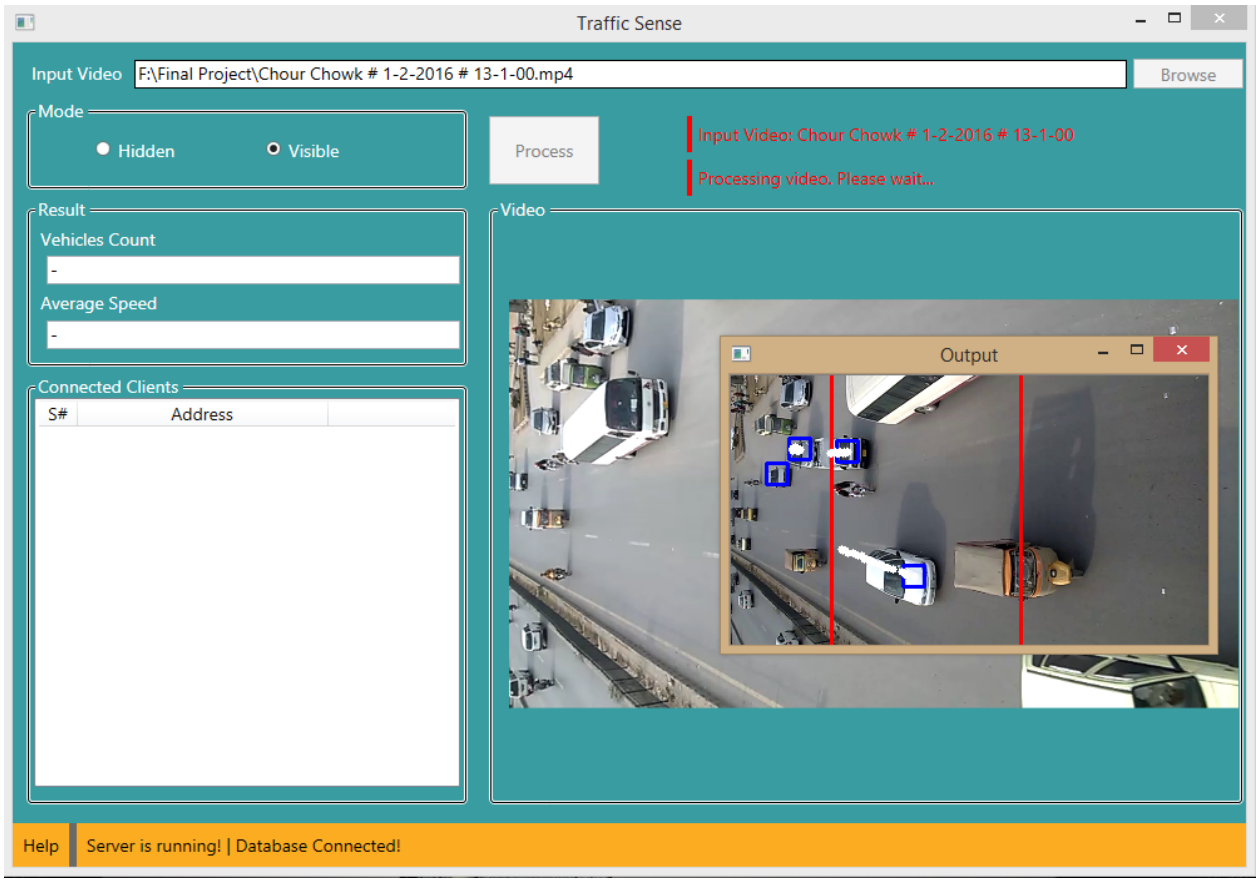

Figure 14Video being processed

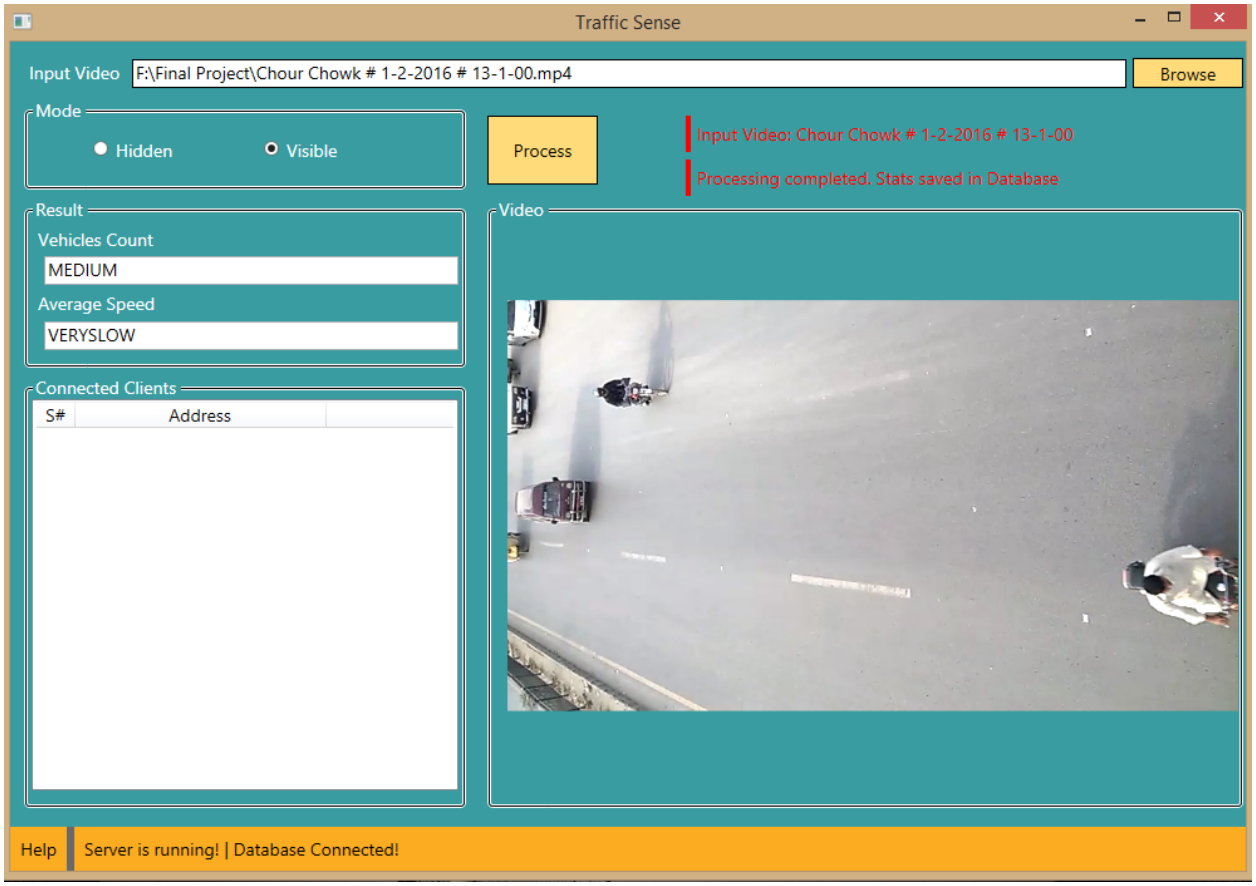

Figure 15Showing result after Processing

4.12.2 Android Main Activity:

This is the interface for Android User. It contains two buttons. ‘Get traffic Info’ is used to request information from server. ‘Alternate Route’ Button shows the possible alternate path between source and destination.

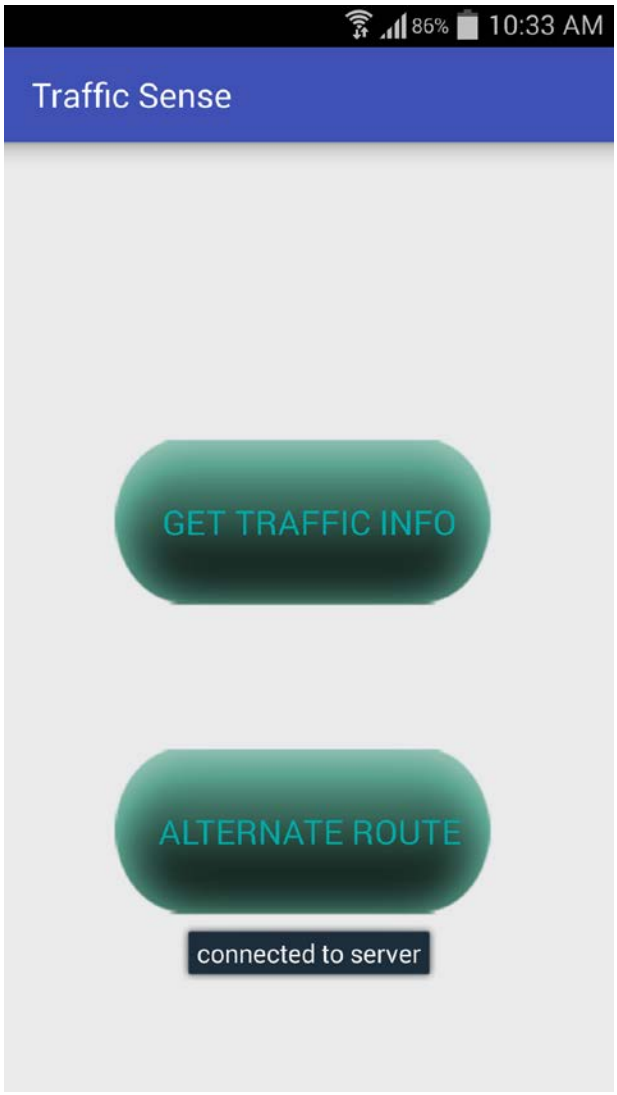

Figure 16Android App Interface

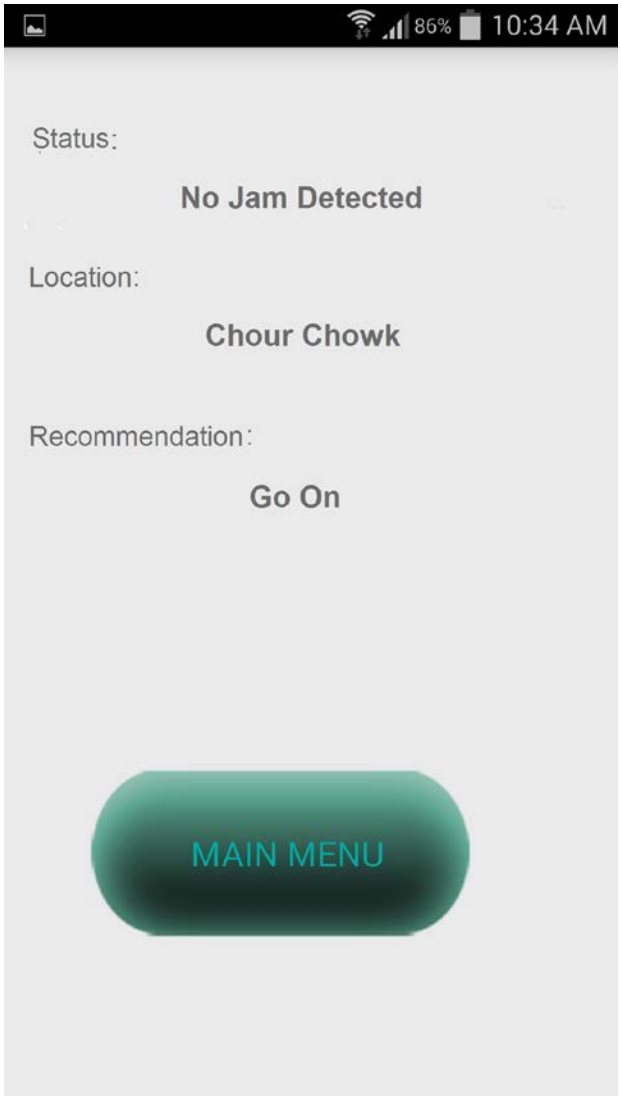

Figure 17Interface showing result to user

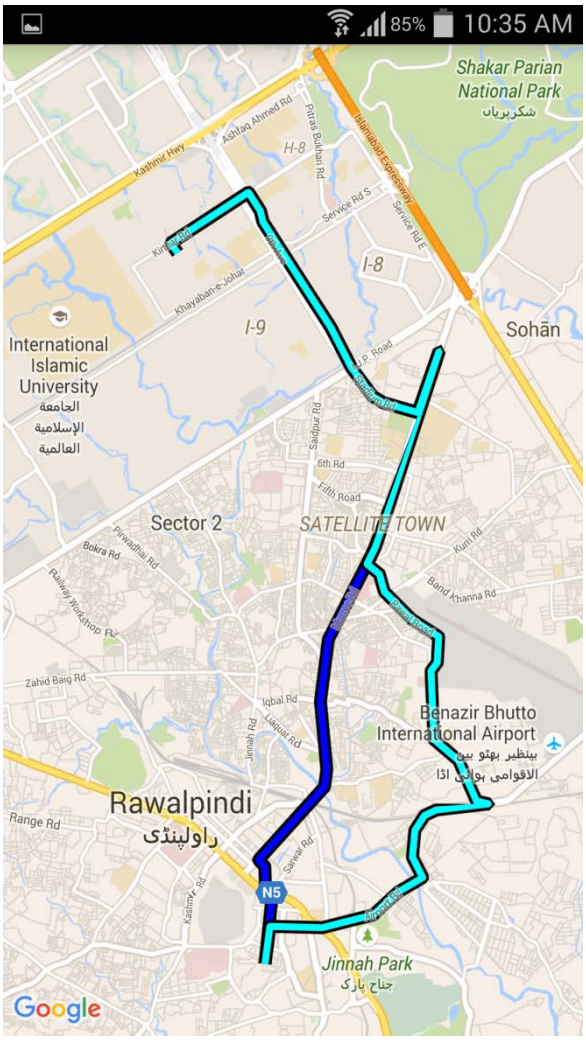

Figure 18Alternate path being shown

4.13 Pseudo code

4.13.1 Server

```
START SERVER

RUN THREAD

WHILE(1)

        ACCEPTINCOMINGCONNECTIONS()

END

IF VIDEOPROCESSED == TRUE

        UPDATECLIENTS()

END
```

4.13.2 VideoAnalyses

```
GETVIDEO

IF TYPE(VIDEO) == PROCESSABLE

        EXTRACTEDINFO = PROCESSFORCOGESTION(VIDEO)

        DECIDECONGESTIONSTATUS(EXTRACTEDINFO)

        EXTRACTEDINFO = PROCESSFORFOG(VIDEO)

        DECIDEFOGSTATUS(EXTRACTEDINFO)

        SAVEINDB (EXTRACTEDINFO)

END
```

4.13.3 Client

```
CONNECTTOSERVER

IF CONNECTION == SUCCESSFUL

        LOCATION = GETLOCATION

        SENDTOSERVER(LOCATION)

        WAITFORRESPONSE

ELSE

        ERROR MESSAGE

END
```

5 Chapter 4: Analysis and Evaluation

5.1 Introduction

The purpose of this chapter is to describe the scope, approach, resources, and schedule of the testing activities for our Don't Rush-Its Jam There application. Testing is done to ensure that the system is working in the manner in which it was intended. Our project is in modules so we did the testing phase by testing each module separately i.e. unit testing and then step by step integrating modules to test them with each other i.e. integration testing and then the complete application is tested as a whole in system testing.

During the testing stage special attention was paid to all of the implemented functions and how they behave on different participants. First of all the focus was set on the interface conceptual model evaluation and then on system evaluation.The purpose of the interface evaluation was to come up with a good user interface before the basic features are implemented. The purpose of the system evaluation was to evaluate the usability and performance of the application. During testing we focused on testing the system processing a video and sending the correct information as planed to the client when requested.

5.2 Test Items:

Performance, Interface, Vehicle Detection, Integration, Client-Server connection, Request and Response, Alternative Routes and Database, Fog Detection.

5.3 Feature to be tested:

Performance, Interface , Vehicle Detection, Integration, Client-Server connection, Request and Response, Alternative Routes and Database.

5.4 Approach:

This is a master test plan in which overall testing strategy shall be beta testing. Our project has client-server architecture so we will start the testing phase by testing server-side application and its features implemented in server, and then client-side application.

5.5 Item Pass/Fail Criteria:

Test Items will pass the test If

a. Pre-condition are met.
b. Inputs are carried out as specified.
c. Test case produce the desired output as mentioned with each test case separately.

5.6 Suspension Criteria and Resumption Criteria:

Testing will be suspended when a defect is introduced/found that cannot allow any further testing. Testing will be resumed after defect removal.

Testing will be suspended when a defect or flaw is found, the defect or flaw will be removed and further testing will be resumed.

5.7 Test Deliverables:

Following are the Deliverables as per this Plan:

a. Test cases
b. Output from tools

5.8 Testing Tasks:

a. Develop Test Cases.
b. Execute tests on the basis of the test cases developed
c. Report defects during tests if any.
d. Complete the test report.
e. Manage the changes made after testing.

•

5.9 Environmental Needs:

5.9.1 Hardware

a. Windows PC with 32bit processor.
b. Android Phone with API level 19 or higher.

5.9.2 Software

a. Visual Studio 2012/2013
b. OpenCV 2.4.9
c. Android Sutdio v1 or higher

5.9.3 Responsibilities

All developers of the project are responsible for the completion of all components testing and integration testing tasks.

5.9.4 Staffing and Training needs

Basics knowledge of testing strategies and techniques is needed for the testing of the project.

All the developers will be testing each other's work and will be actively participating in the development and testing of the project simultaneously.

5.9.5 Schedule:

All the test cases are executed at least once By March 1, 2016. If a defect was found, the developers will need to fix the problem. The testers should then re-test the failed test case until it is functioning correctly.

The performance testing is being done in parallel with the development. The tester need to conduct regression testing by April 1-5, 2016 to make sure the developers did not accidentally break parts of the software while fixing another part. This can occur on test cases that were previously functioning properly.

The system acceptance testing was done when the project is completed that is by April 7-9, 2016.

5.9.6 Risks and Contingencies:

5.9.7 Budget Risk:

The budget will be compensated by using less costly alternatives to fit the budget requirements.

5.9.8 Operational Risks:

Operational risks will be eliminated by Scheduling daily meetings and regular deadlines to meet the goals of the project as well as provide proper communication within the group.

5.9.9 Technical risks:

Technical risks will be eliminated by keeping the once defined requirements constant.

5.9.10 Programmatic Risks:

In case of a programmatic risk the scope of the project will be limited in order to stay inside the constraints of the project.

5.10 Test Cases

Test Case Name	Video Feed
Test Case No	1
Description	Browse, select and open any valid video.
Testing Technique Used	White Box Testing
Precondition	Server must be running.
Input Values	MP4 or AVI video of traffic
Valid Inputs	Video that uses Microsoft DirectShow Codecs. i.e. MP4, AVI, WMV
Steps	1. Start Server 2. Browse to folder 3. Select Video 4. Click Open
Expected Output	If selected file is a valid video file, it's loaded and shown in Media Element of GUI otherwise it generates error.
Actual Output	Video is loaded
Status	Pass

Test Case Name	Congestion Calculation
Test Case No	2
Description	Process video and show the output after processing
Testing Technique Used	White Box Testing
Precondition	Video file should be a valid traffic video from specified angle and height. This must be insured from user.
Input Values	Video file selected already
Valid Inputs	A video file that must exists
Steps	1. Video is selected and loaded

	2. Select mode between Hidden and Visible, applied to processing of video 3. Click Process button 4. Program will start processing of video and wait for it to finish 5. Result is saved in result.txt file 6. Program reads that input file
Expected Output	If video is processed successfully, it writes output to result.txt file, shows the result in two textboxes on GUI i.e. Vehicle Count and Average Speed. And result is saved in database.
Actual Output	Video is processed successfully and result of video is shown as expected and saved in database.
Status	Pass

Test Case Name	Establish Connection to Server
Test Case No	3
Description	Android app establishes connection with server through sockets.
Testing Technique Used	White Box Testing
Precondition	Server must be running.
Input Values	-
Valid Inputs	-
Steps	1. Start Android App 2. Start thread that connects with server
Expected Output	If server is running, then connects with server and show message in toast about connection with server.
Actual Output	Connected with server and toast show “connected with server”
Status	Pass

Test Case Name	Request and Response

Test Case No	4
Description	Request information from server and get response
Testing Technique Used	White Box Testing
Precondition	Server must be running and client should be connected to server.
Input Values	Source Location: Saddar Destination Location: Pirwadhai
Valid Inputs	Source location and destination location from auto-fill of Google API places.
Steps	1. Click “Get Traffic Info” button 2. Enter Source Location 3. Enter Destination Location 4. Click “OK” button 5. Next activity shows response from server.
Expected Output	If server has information about this entered route, server returns information about traffic condition on this route else it shows “no information found”.
Actual Output	It shows response from server about traffic condition.
Status	Pass

Test Case Name	Alternate Route
Test Case No	5
Description	Show alternate route in Google Maps
Testing Technique Used	White Box Testing
Precondition	Internet must be available
Input Values	Source location and destination location
Valid Inputs	Source and destination locations
Steps	1. Click “Alternate Route” button 2. Enter source location

	3. Enter destination location 4. Click “OK” button 5. Next activity show map of alternate route
Expected Output	If internet is working, it shows map of alternate route from Google APIs.
Actual Output	Map shows alternate route.
Status	Pass

6 Chapter 6: Future Work

The system developed can be improved in way of detection of jam through video in real-time. The extended scope of this project can be improved to fog and many other detection system. The project can be applied in real-life for traffic signaling, traffic police help and more accurate jam detection in real-time.

This system can be applied for traffic police help, rescue, VIP movements, choosing be ambulance route for ambulance etc.

7 Chapter 5: Conclusion

We have able to processing video to calculate the detection parameters using image processing algorithms and calculated the speed of vehicles. We have calculated the average speed of vehicles. On the basis of vehicle average speed and vehicle count we have categorized the traffic congestion status. If the vehicle count is high and the average speed of the vehicles is below a threshold speed then we have categorized this situation as the high congestion traffic situation. If the vehicle count is normal and the average speed of vehicles is also in the normal speed bracket then this situation is categorized as the normal traffic situation. If the vehicle count is low but the average vehicle speed is below threshold then this situation is categorized as congestion situation. After the calculation of results from the input video we have saved the result of that video and that location name in a database. On the other hand we have developed an android app in which user is asked to enter his current location and his desired location. Both the source and destination of the user is sent to server for result fetching. On the server, the source and destination location of the user is checked in database. If the server has results for the particular path, then server sends back a result as traffic status on the path and the time when the video was captured. If there is no result for that path in database then an error message is sent to the user.

Bibliography

References

1. [Online]. Available: https://www.fhwa.dot.gov/policyinformation/pubs/vdstits2007/vdstits2007.pdf.

2. [Online]. Available: http://www.siemens.co.uk/traffic/pool/documents/brochure/wimag-flyer.pdf.

3. Y. Wang, Y. Zou, H. Shi, and H. Zhao, "Video image vehicle detection system for signaled traffic intersection," vol. 1, pp. 222–227, Aug. 2014. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5254300&tag=1.

4. N. Chintalacheruvu, V. Muthukumar, and S. R. Publishing, "Video based vehicle detection and its application in intelligent transportation systems," *Journal of Transportation Technologies*, vol. 02, p. 305, Oct. 2012. [Online]. Available: http://www.scirp.org/journal/PaperInformation.aspx?paperID=23832.

5. "Behance,". [Online]. Available: https://www.behance.net/gallery/Vehicle-Detection-Tracking-and-Counting/4057777.

6. "Object detection using Haar - cascade Classifier,". [Online]. Available: http://ds.cs.ut.ee/Members/artjom85/2014dss-course-media/Object%20detection%20using%20Haar-final.pdf.

7. "Google maps Android API | Google developers," Google Developers. [Online]. Available: https://developers.google.com/maps/documentation/android-api/. Accessed: May 13, 2016.