CENG3004: Software Engineering

*SMART TRANSPORTATION SYSTEM*

**Design Document**

**22/05/2018**

ÇAĞATAY ŞENTURCA

**DISCLAIMER**

This document is produced based on Software Requirements Specification (SRS) that was delivered to and agreed by the customer. For the success of the project, it needs to be developed carefully tracing back to requirements as applicable, set design goals and address design goals thoroughly.

This page can be removed once read.

Contents

1. Overview 3
2. Design Goals 3
3. System Models 3
   1. Class Diagrams 3
   2. Sequence Diagrams 3
   3. Activity Diagrams 3
   4. Statechart Diagrams 4
4. Subsystem Decomposition 4
5. Hardware / Software mapping 4
6. Other Design Concerns (use relevant subsections) 5
   1. Concurrency 5
   2. Data Management 5
   3. Global Resource Handling 5
   4. Boundary Conditions 6
7. Glossary 7
8. References 7
9. Appendix 7
10. Overview

**WHO IS THE CUSTOMER?**

Our system customer could be anyone who wants to use public transportation.

**WHAT IS THE SYSTEM , WHAT DOES IT DO, WHY DO YOU DO IT?**

This system helps the people in transport from one point to another by using public transportation. More specificly, someone wants to go to a place from his location. At this point , our project helps this person at finding the best option like ; How many buses he needs to change? What time is he going to arrive the location where he wants to go? What time he needs to leave his home to go to bus stop?

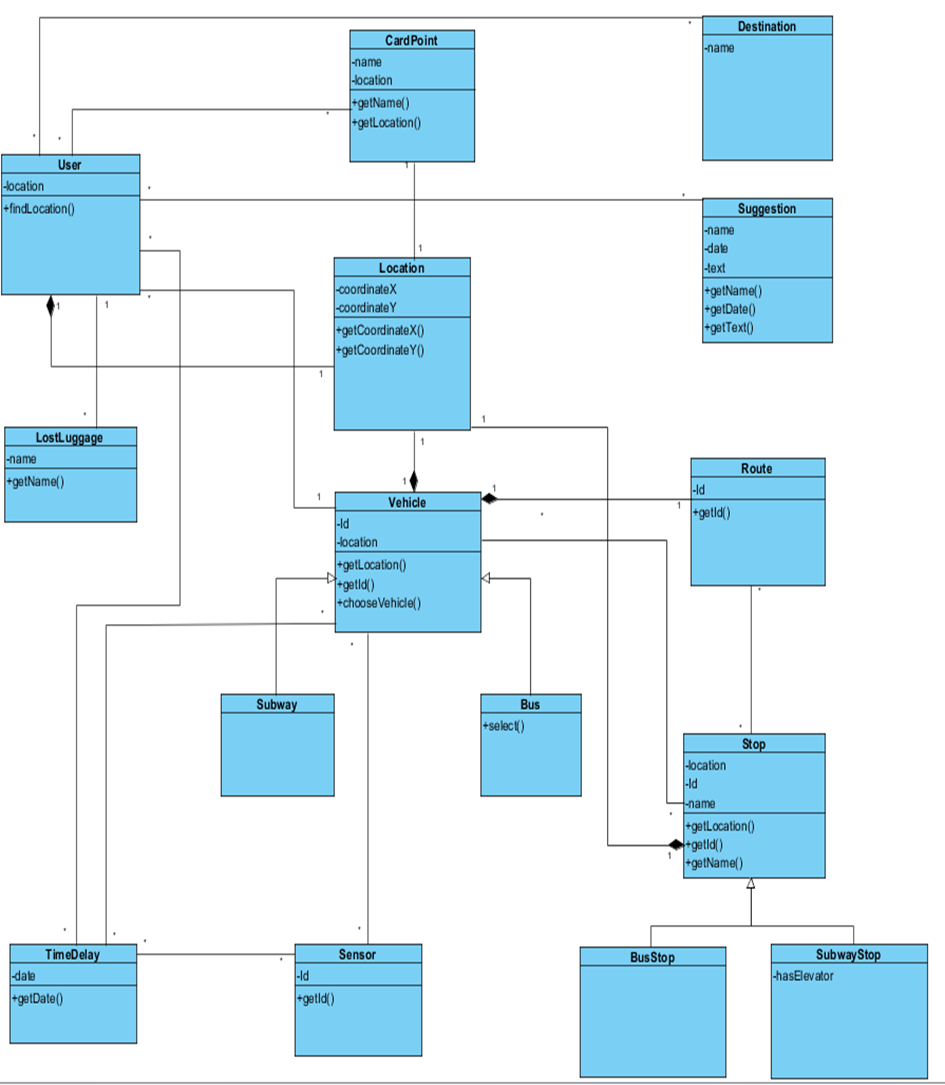
In this project ,we are willing to make people save their time by using transport and if you are stranger at the city this project is the best for you. We are offering them to save time, comfortable transport.

1. Design Goals

|  |  |  |
| --- | --- | --- |
| **Goal’s Concern** | **Related Requirement Identifier** | **Description** |
| Traceability of requirements |  | Users can give feedback about application. |
| Well-defined interfaces |  | Users should use the app easily. |
| Backward-compatibility | NR-1 | The application should be Backward-compatibility. |
| Reliability | NR-1 | The application should work correctly. |
| Modifiability | NR-3 | The application should be modifiable to use different map services. |

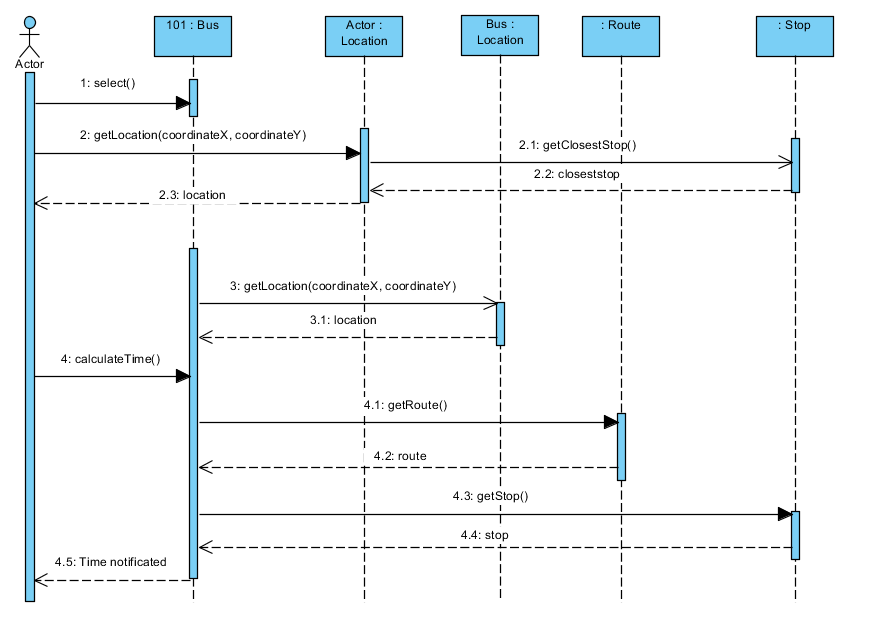
3.System Models

## 3.1 Class Diagram

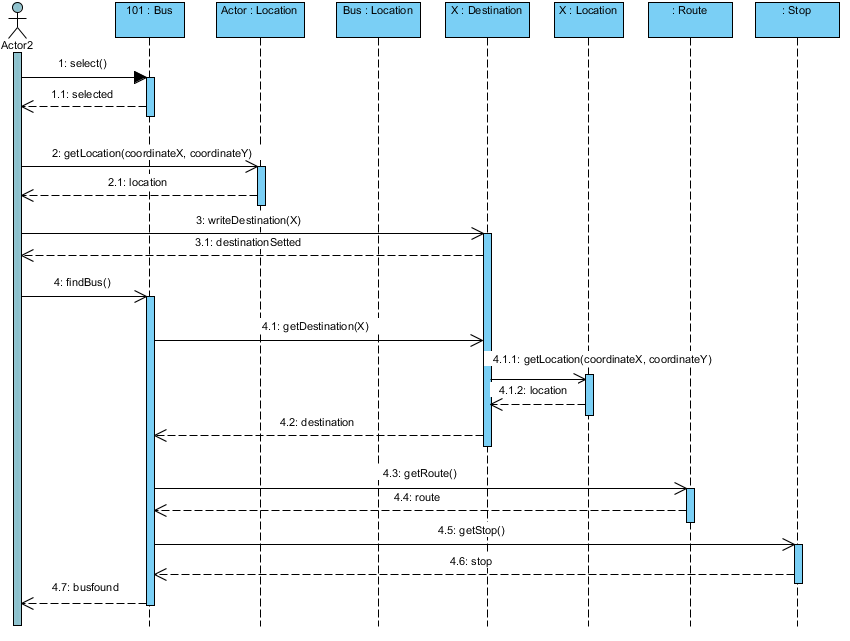


## Sequence Diagrams

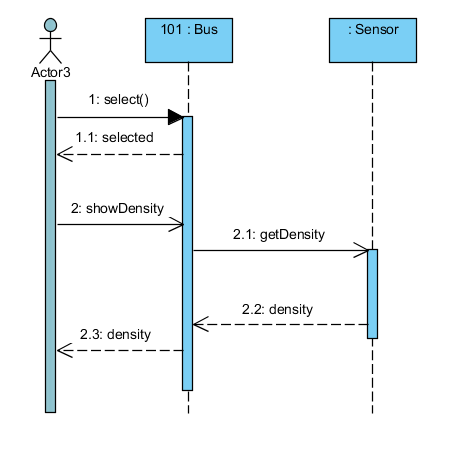
1-CALCULATE TIME



2-PLAN TRIP

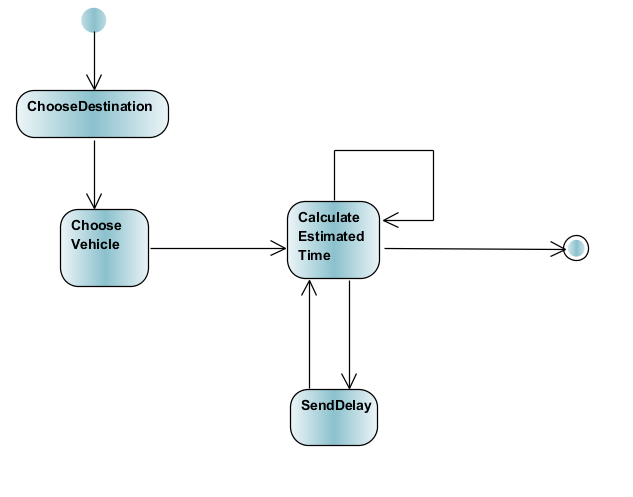


3-SHOW DENSITY

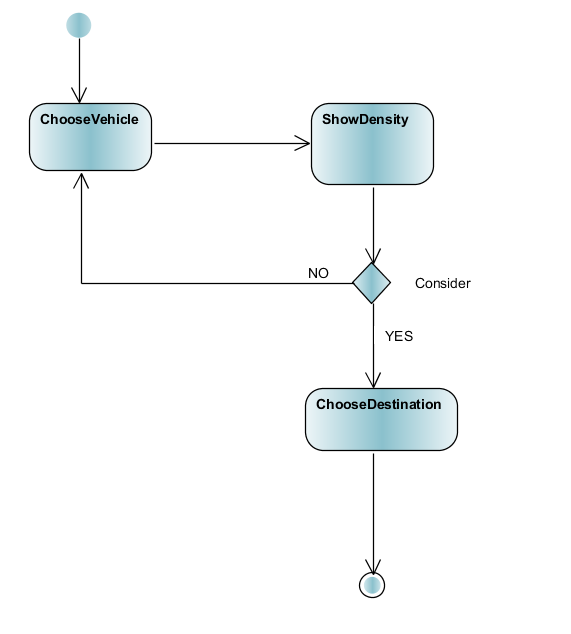


## Activity Diagrams

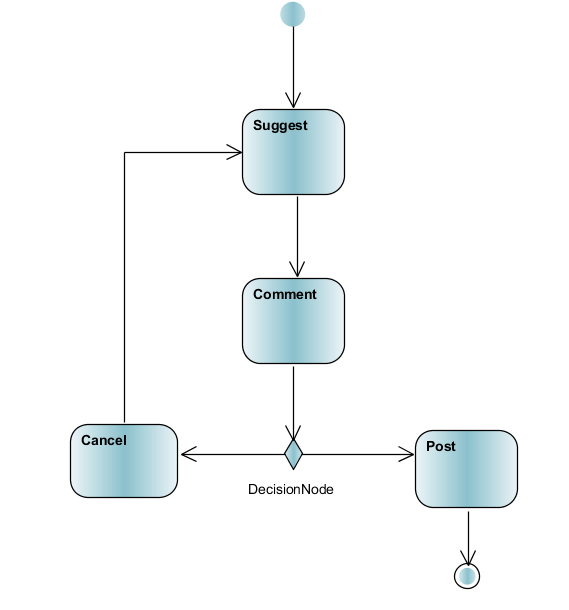
1-



2-

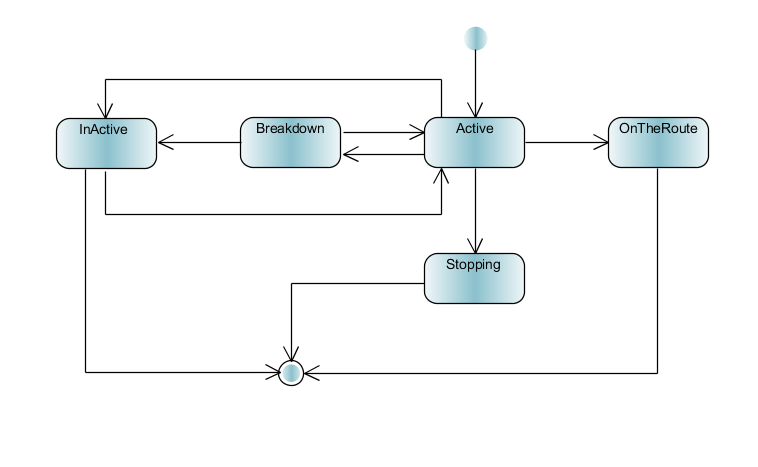


3-

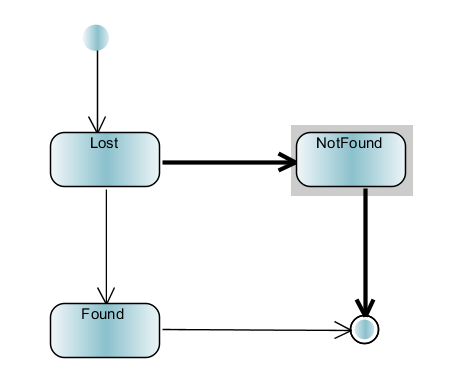


## Statechart Diagrams

1-BUS

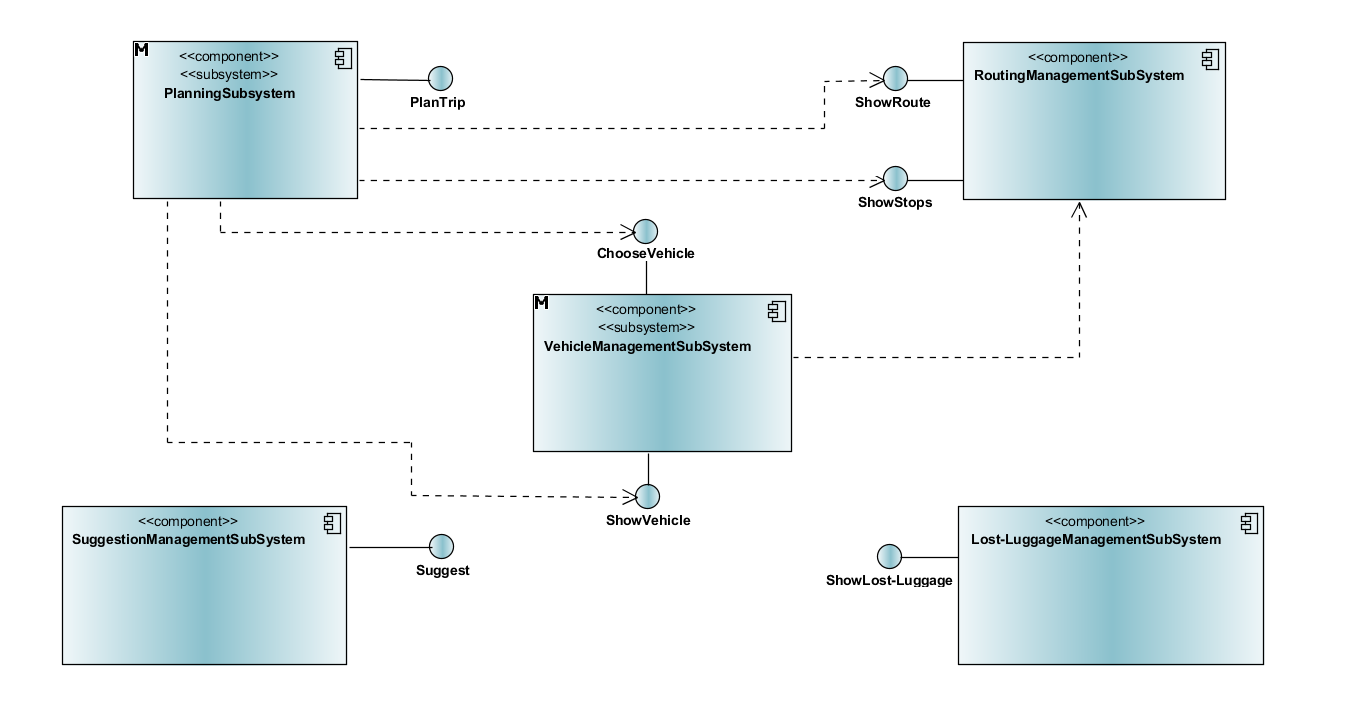


2-LOST LUGGAGE



1. Subsystem Decomposition

COMPONENT DIAGRAM



**PlanningSubsystem** ; It is responsible for constructing a Trip connection a sequence of destinations.It is also responsible for responding to replan requests from RoutingManagementSubsystem. (Classes ; Destination, Route, Stop, Location, Vehicle)

**RoutingManagementSubsystem** ;It is responsible for downloading a trip from the planning service and it is also responsible changing routes of vehicles.(Classes ; RouteProxy, StopProxy, Vehicle)

**Lost-LuggageManagementSubsystem** ;It is responsible for lost items. It adds and deletes lost items. Also it follow lost items situations ( Classes ; Lost-luggages )

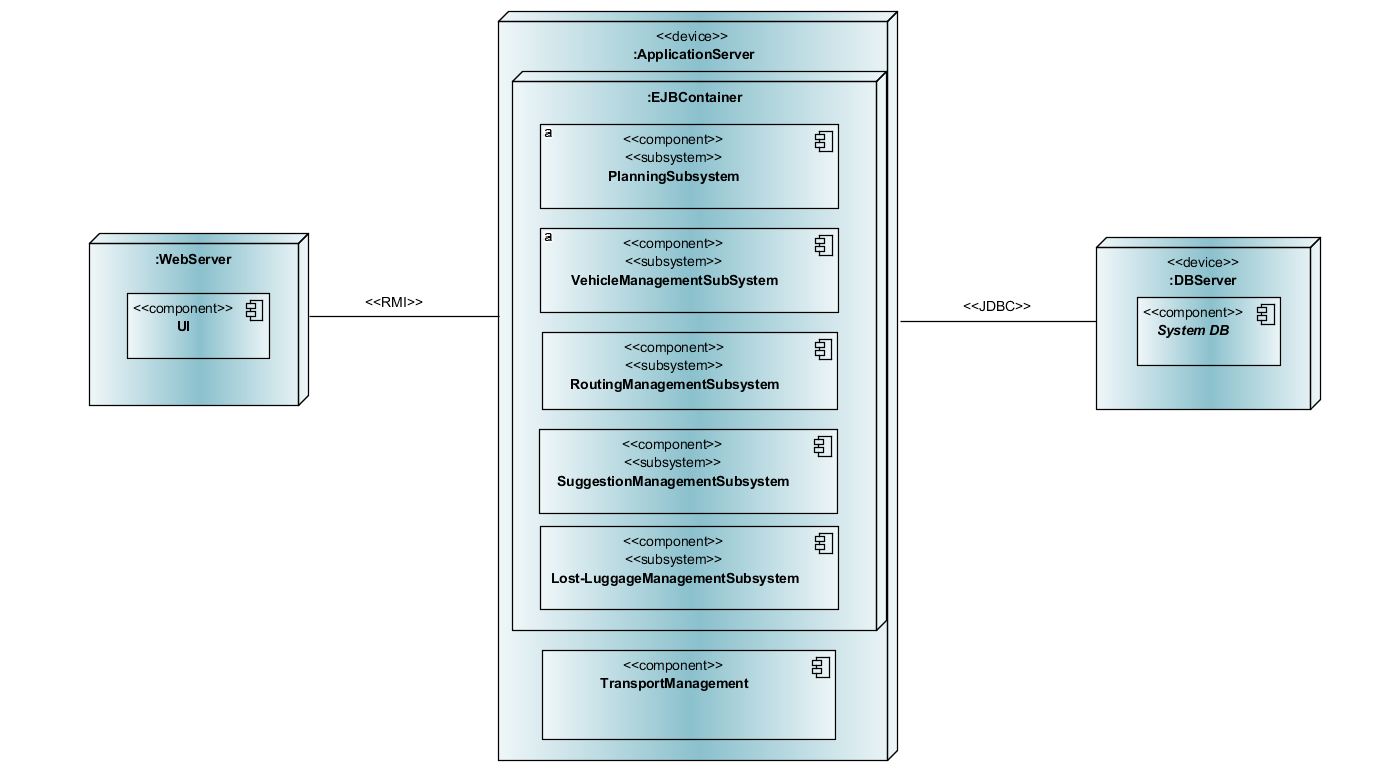
**VehicleManagementSubsystem** ; ;It is responsible to add and delete new vehicles.

(Classes ; LocationProxy, VehicleProxy, RouteProxy, Sensor, StopProxy, TimeDelay)

**SuggestionManagementSubsystem** ; It controls all suggests.(Classes ; Suggest)

5.Hardware / Software mapping

DEPLOYMENT DIAGRAM



6.Other Design Concerns (use relevant subsections)

## 6.1 Concurrency

The System provides access to multiple users.

Example:

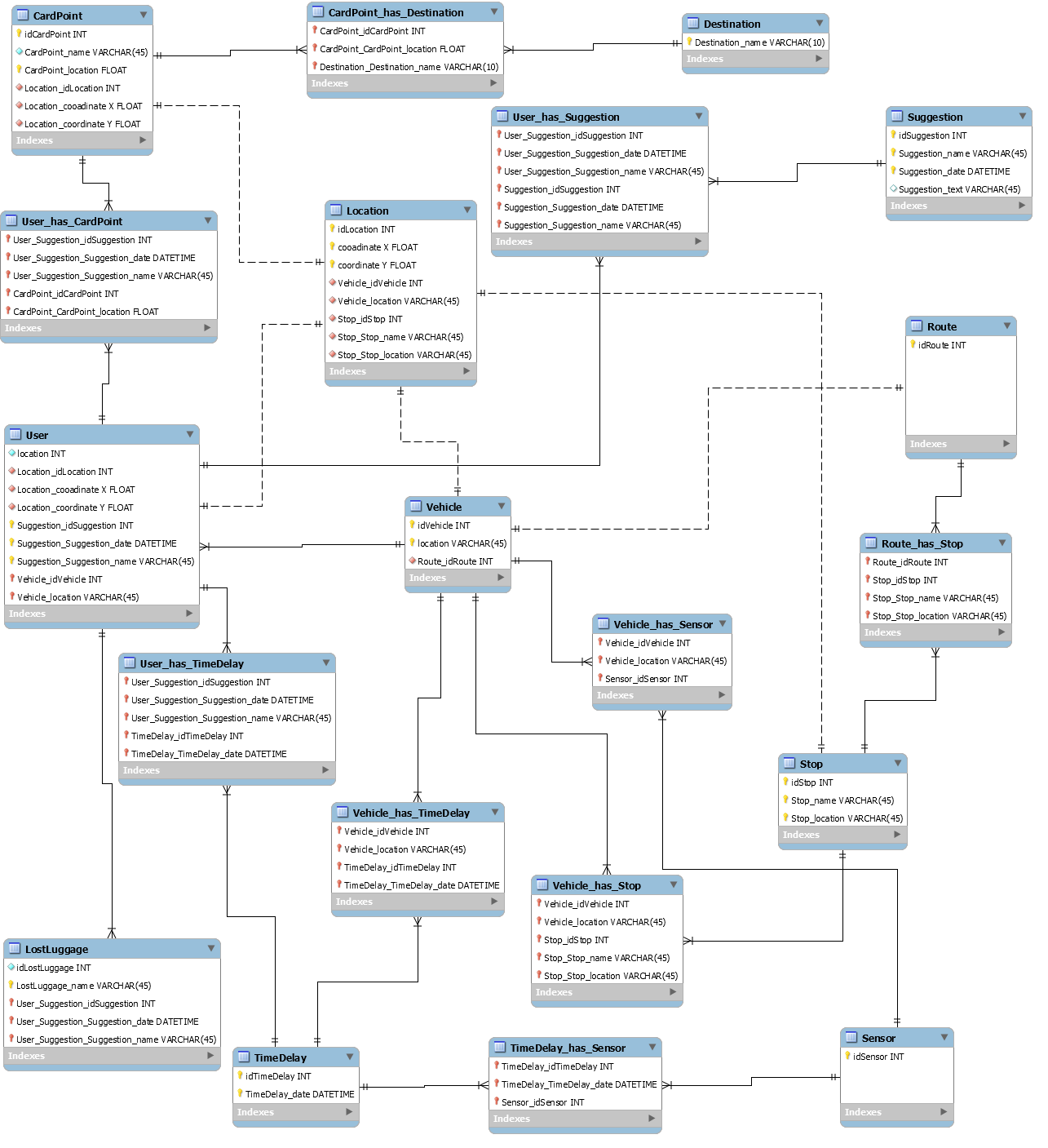
Multiple users can get the location of a specific bus or subway concurrently from the same service.

Multiple users can see their own location from the same service.

Multiple users can access the same bus stop concurrently and make calculations.

## Data Management

ER DIAGRAM



## Global Resource Handling

Discuss authentication/authoritization mechanisms. Provide an “Access Matrix” using f ctors should correspond to actors in uses cases from Analysis Document..

|  |  |
| --- | --- |
| **Classes** | User/Actor |
| Destination | setDestination()  getDestination() |
| CardPoint | getLocation() |
| Suggestion | addSuggest()  deleteSuggest() |
| Location | getLocation() |
| Lost-Luggage | findLostluggage()  reportLostluggage() |
| Vehicle | showVehicleonmap()  getLocation()  getDensity() |
| TimeDelay | reportDelay() |

## Boundary Conditions

Discuss boundary conditions initialization, termination and failure. See below for some questions relevant:

Initialization

* What data need to be accessed at startup time?

The location of the user.

* What services have to be registered?

No need any registration.

* What does the user interface do at start up time?

Main page of app directly comes up on the screen.

Termination

* Are single subsystem is allowed to terminate?

Single subsystem is allowed to terminate

* Are subsystems notified if a single subsystem terminates?

If a single subsystem terminates, subsystems notifies to each.

* How are updates communicated to the database?

Every subsystem can update DB when any changes becomes in them.

Failure

* How does the system behave when a node or communication link fails?

System sends a failure message to users.

* How does the system recover from failure?

It depends on what kind of error system gets. We report error to person of interest.

1. Glossary
2. References
3. Appendix