**CMSC 335**

**Herman Mann**

**Project - 3**

**User Guide, Test Plan, UML and Lesson Learned**

**05/08/2022**

**Contents**

[**Introduction** 3](#_Toc77133169)

[**Overview** 4](#_Toc77133170)

[**Design** 5](#_Toc77133171)

[**Test Cases** 6](#_Toc77133172)

[**Test Case #1: Build and Run the Project** 6](#_Toc77133173)

[**Test Case #2: Click Start button – To start vehicle traffic simulation** 7](#_Toc77133174)

[**Test Case #3: Click Pause button – To pause vehicle traffic simulation** 8](#_Toc77133175)

[**Test Case #4: Click Continue button – To Continue vehicle traffic simulation** 10](#_Toc77133176)

[**Test Case #5: Click Stop button – To Stop vehicle traffic simulation** 11](#_Toc77133177)

[**Test Case #6: Intersection Color – Display Green/Yellow/Red** 12](#_Toc77133178)

[**Setup and Run Application** 13](#_Toc77133179)

[**Lesson Learned** 14](#_Toc77133180)

# **Introduction**

This is the project of the Class 335. In this project, I have constructed a Java Swing GUI that uses event handlers, listeners and incorporates Java’s concurrency and multithreading functionality and the use of threads. I designed and implemented the GUI for the project using concepts of Object Orientated Programming.

In this project, I used the Java Swing GUI that displays time, traffic signals and other information for traffic analysts. The final GUI design is up to you but should include viewing ports/panels to display the following components of the simulation for Current time stamps in 1 second intervals, Real-time Traffic light display for three major intersections and X, Y positions and speed of up to 3 cars as they traverse each of the 3 intersection.

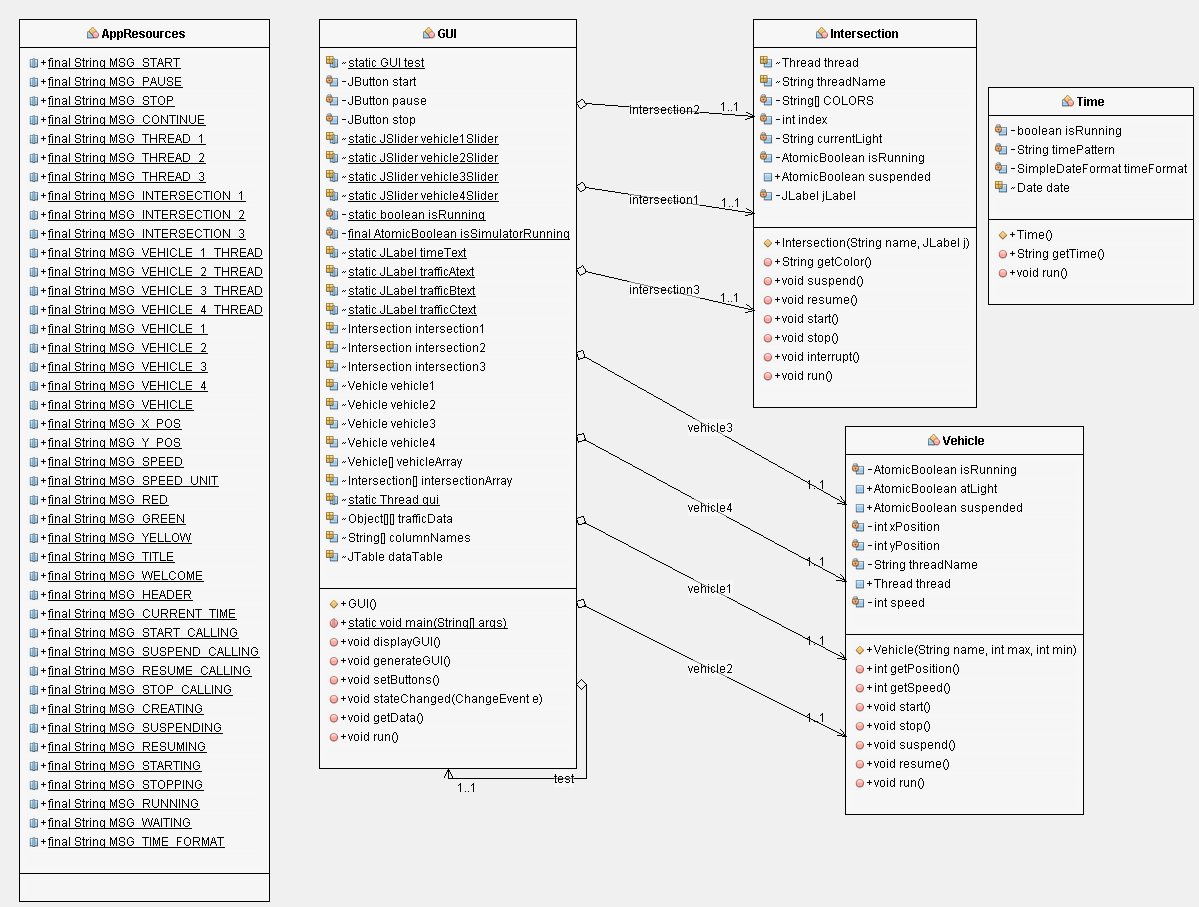
This simulation project are including the following guidelines such as the components listed above should run in separate threads, Loop through the simulation with button(s) providing the ability to start, pause, stop and continue the simulation, uses of distance formula with units and assume a straight distance between each traffic light of 1000 meters as well as vehicle is traveling a straight line, by assuming Y = 0 for your X, Y positions.

# **Overview**

This project is implemented using the Java programming language. In this project, I created the following classes using the mutltithreading and cocurrency concepts of advanced Java.

1. Project2GUI – Class to control handle and draw GUI for the application and implements the runnable with change listener events
2. AppResources - Class to handle application resources
3. Intersection - Class to handle intersection to display traffic light colors
4. Time - Class to handle the application date time and implements the runnable
5. Vehicle - Class to handle the vehicle related methods and implements the runnable

# **Design**



# **Test Cases**

|  |  |
| --- | --- |
| **Test Case #1: Build and Run the Project** | |
| **Description** | To build and run the project successfully |
| **Input** | Application code |
| **Expected Result** | Project is successfully build and run |
| **Actual Result** | Project is successfully build and run without error |
| **Pass** | **Yes** |
| **Screenshot** |  |

|  |  |
| --- | --- |
| **Test Case #2: Click Start button – To start vehicle traffic simulation** | |
| **Description** | To Start vehicle traffic simulation |
| **Input** | Click start button |
| **Expected Result** | Vehicle traffic simulation should be started successfully |
| **Actual Result** | Vehicle traffic simulation started successfully |
| **Pass** | **Yes** |
| **Screenshot** |  |

|  |  |
| --- | --- |
| **Test Case #3: Click Pause button – To pause vehicle traffic simulation** | |
| **Description** | To Pause vehicle traffic simulation |
| **Input** | Click pause button |
| **Expected Result** | Vehicle traffic simulation should be paused successfully |
| **Actual Result** | Vehicle traffic simulation paused successfully |
| **Pass** | **Yes** |
| **Screenshot** |  |

|  |  |
| --- | --- |
| **Test Case #4: Click Continue button – To Continue vehicle traffic simulation** | |
| **Description** | To Continue vehicle traffic simulation |
| **Input** | Click continue button |
| **Expected Result** | Vehicle traffic simulation should be continue successfully |
| **Actual Result** | Vehicle traffic simulation continue successfully |
| **Pass** | **Yes** |
| **Screenshot** |  |

|  |  |
| --- | --- |
| **Test Case #5: Click Stop button – To Stop vehicle traffic simulation** | |
| **Description** | To Stop vehicle traffic simulation |
| **Input** | Click Stop button |
| **Expected Result** | Vehicle traffic simulation should be stopped successfully |
| **Actual Result** | Vehicle traffic simulation stopped successfully |
| **Pass** | **Yes** |
| **Screenshot** |  |

|  |  |
| --- | --- |
| **Test Case #6: Intersection Color – Display Green/Yellow/Red** | |
| **Description** | To display intersection color green/yellow/red vehicle traffic simulation |
| **Input** | Click Start button |
| **Expected Result** | Vehicle traffic simulation should display intersection color green/yellow/red successfully |
| **Actual Result** | Vehicle traffic simulation display intersection color green/yellow/red successfully |
| **Pass** | **Yes** |
| **Screenshot** |  |

# **Setup and Run Application**

1. Download the project zip file
2. Open the NetBeans IDE
3. Under File menu option, Select Import Project from ZIP
4. Choose the project zip file location from file chooser dialog box and click import button
5. Project will be imported in NetBeans
6. Compile the project
7. Run the project

# **Lesson Learned**

During the design and implementaion of this project, I have learned a lots of new things about the GUI and Object Oriented Programming Languages. Here are few implmetation done in this projects

* Coded all the classes in Java with advanced concepts of OOPs and mutlithreading application
* Java Swing library is used for the GUI implementation
* Learned s Swing AWT components including Layout Managers, Event Handlers, Listener Interfaces, Adapter Classes, Inner Classes, Buttons and other widgets GUI components, mouse events, button events
* Application resources are maintained in as share class
* Inheritence and polymorphism concepts are used with the help of extends, super and override keywords
* Project is written in resuseable code manner
* A modular approach of coding is used for the implementation of the classes
* Error and Exception handling is properly implemented