Software Design Document -

NSW Traffic Penalties Data Analysis Tool

**Jack McLeod**

s5135484

Wednesday 10:00am – 11:45am

**Lachlan White**

s5182678

Online

**Meesum Kazmi**

s5176618

Online

**Group 100**

Data set

Australia NSW traffic penalty data 2011-2017

Table of Contents

[1.0 System Vision 3](#_Toc81738989)

[1.1 Problem Background 3](#_Toc81738990)

[1.2 System Overview 3](#_Toc81738991)

[1.3 Potential Benefits 3](#_Toc81738992)

[2.0 Requirements 4](#_Toc81738993)

[2.1 User Requirements 4](#_Toc81738994)

[2.2 Software Requirements 4](#_Toc81738995)

[2.3 Use Cases 4](#_Toc81738996)

[3.0 Software Design and System Components 7](#_Toc81738997)

[3.1 Software Design 7](#_Toc81738998)

[3.2 System Components 8](#_Toc81738999)

[3.2.1 Functions 8](#_Toc81739000)

[3.2.2 Data Structures 9](#_Toc81739001)

[3.2.3 Detailed Design 10](#_Toc81739002)

[4.0 User Interface Design 12](#_Toc81739003)

[4.1 Structural Design 12](#_Toc81739004)

[4.2 Visual Design 13](#_Toc81739005)

# System Vision Document

## Problem Background

The New South Wales (NSW) government records hundreds of thousands of traffic violations on state roads every year. Due to this, NSW Traffic wishes to develop a tool that can enable uses to quickly analyse large penalty / offence datasets enabling them to create a safer traffic environment in local communities. A predefined cumulative data set has been provided by the NSW government that features various violations relating to primary traffic offenses. This cumulative data set, titled “NSW Traffic Penalty”, contains a vast range of data types in relation to traffic penalties that are to be interpreted by the proposed software application. This tool will be developed to be used by administration officers through specific set of parameters. The goal of this will be to generate statistical analysis which will in turn provide more advanced insight.

## System Overview

The completed system will encompass a lookup and collation tool for the internal staff to determine trends and generate statistical information about traffic penalties. Utilizing a simple graphical user interface (GUI) it will encompass a wide range of functions to return formatted strings, graphs and tables of requested information. The types and number of tools will be pre-determined and defined ahead in this document.

The tools basic specifications are as follow:

* An independent functioning software (offline)
* Connecting to the existing database to import CSV files
* Display all requested information from user prompt
* Returning a graphical representation that reflect a user’s prompt if applicable
* Allowing comparative data based on specified criterion (date range, type of offence, offence code)
* Successfully completes the functional requirements as outlined by NSW Traffic

## Potential Benefits

The potential benefits of the data analysis tool are only limited to the users' parameters. It will allow one to efficiently analyse large data sets quickly, reducing labour cost and worktime. It will allow NSW Traffic stakeholders the ability to make confident statistically reinforced decisions for NSW roads, hopefully improving the commute of your everyday citizen. The secondary use for this tool would be the aid in identifying risks to the community in the form repeat offenders, potential high-risk areas, and trends.

Some potential benefits have been identified as follows:

* Improve the ability of users to access filtered information.
* Reduction / removal in labour costs for simple queries promoting efficiency.
* Potential for a better understanding of trends from ongoing occurrences.
* Provide high level insight to the selected data for further action.
* Allow for pattern recognition due to the visualization of data.
* Reinforce decision making

# Software Requirements

## User Requirements

The user will need a few pieces of information to find their affected item.

* Users will be able to select with an input from five default or a combination of functions
* Users will have the ability to filter outputs via functions.
  + List available functions.
  + Retrieve information based on “Keyword search”.
  + Retrieve information based on a Specified *Date range.*
  + Retrieve information based on the F*requency of occurrence.*
  + Retrieve information based on *various orderings* (most common – least, etc).
* The software will be able to handle invalid or erroneous inputs by prompting for a new input rather than terminating the application.
* All output will be non-mutable (read-only).
* Users will be able to clear search via the form

## Software Requirements

The several minimal requirements as requested by NSW Traffic include:

* The software will provide an error message with appropriate assessment of the issue (incorrect filename / path).
* The application will handle all inputs and ignore erroneous errors.
* The application will prompt the user for another/alternate input if input is not applicable.
* All information from the file will be read only (non-mutable).
* The software will be capable of handling multiple specifications.
* The software will be capable of displaying a visual representation of specific functions only.
* In the event of an erroneous error in the data set the application will continue without terminating.

## Use Cases

Depicted in *Figure 2.3* below shows several use cases that a potential user might input into the data analysis tool. The use case features the selected input, then a description of a potential output from the tool. The use case diagrams can be found below the table with each case labelled at the top of each diagram.

|  |  |
| --- | --- |
| Use Case | Description |
| Various functions | Successfully select a function to manipulate data |
| Organize outputs via by specified date range | Input a date-range to produce information for all occurrences during specified period. |
| Filter via offence code | Display all results by commonality from offence code occurrence. |
| Filter results by a keyword | Input a date-range and a keyword to retrieve all results that match the query in the DCA\_CODE column. |
| Filter results to analyse the type of offence against the value | Display a cumulative list of offences and their total values identifying the highest value offences. |
| Filter results by traffic penalties vs other offences | Prompted from the user a result will display the number of traffic penalties and when vs other offence types and when they occur |
| Clear results via button | Clear all results returning display to default state. |

***Figure 2.3 Use cases***

Diagram

Description automatically generatedDiagram, schematic

Description automatically generated

Diagram

Description automatically generatedDiagram

Description automatically generated

Diagram

Description automatically generated

Diagram

Description automatically generated

# Software Design and System Components

## Software Design

Depicted below in *Figure 3.1* is the software design flowchart, this documents to engineers how the application should work within reason . Beginning the flowchart can be seen a *Startfile()* function, this will attempt to call the file and read the contents of that file based on the user input. The user will then select (via input) the function that they are seeking to enact. The input will be validated, for both *Startfile()* and the *search()* functions respectively. If an invalid input is entered the program instead of closing will simply re-prompt the user for an alternate input

Diagram

Description automatically generated Figure 3.1 – Software design

## System Components

### Functions

**StartFile** – This function is to be called when the program is started. This will load the file as a readable object and assign it a variable name. This function will feature error handling via prompting for an alternate input if the input is invalid or the data file does not exist. This function serves to only load the file as a named object. This function does not produce any side effects.

**Select** – This function is to be used as the main handling option for users within the data analysis tool. It is to be called to define the function from a pre-determined list of strings as the parameter. This function does not return a value however what will be returned will be used as a reference to the function that is required. This function will call for an input from the user based on the function selected.

The five functions available to be selected are; each function will have an associated input defined with its function information.

* All (default)
* analysis
* Search
* insight

This function does not produce any side effects.

**all** - This function is to be called for to display all information related to violations that occurred during a specific date range. It will accept two formatted strings as parameters via an input from the user. Once called the input data will be validated and occurrences between the specified dates will be outputted as a list. This function will utilize error handling along with all other functions in the form of re prompting users for inputs when needed. This function does not produce any side effects.

**analysis -** This function is to be called to display statistical charts of offenses between specified parameters inputed from the user. Combined with other selection criteria’s the user can limit the number of ouputs to ensure the intended result.

* OFFENCE\_CODE
* LOCATION\_DETAILS
* TOTAL\_VALUE

This function will have a returned result that is categorized by the above listed filters. This function does not produce any side effects.

**Search** – This function is to be called for the purpose of displaying all information related to violations that occurred during a specific date range that are relevant to a code from (OFFENCE\_CODE). It will accept two formatted strings as parameters via an input from the user. It will also accept a third *keyword* that is referenced against the OFFENCE\_CODE column. Once called and the input data validated, any occurrences between the specified dates with the keyword will be returned as a list. This function will see basic error handling as with all functions in the method of re-prompting for a valid input. This function does not produce any side effects.

**Insight** – This function will analyse the frequency of violations occurring over time with respect to a range of specified parameters. This function takes three parameters; the first two parameters accept dates formatted as strings inputted by the user, while the other is a choice between two pre-determined strings: ‘camera type’ and ‘value’. The function filters both violations and their worth by the selected user period, as well as categorising results by the pre-determined string. This will return the desired filter in a nested dictionary (dictionary of dictionaries). This function will be able to take several other functionality camera type and value are merely an example. Error handling will be provided if either parameter is in the wrong format, or non-existent. This function does not produce any side effects.

### Data Structures

**File Object:**  This data structure will only be utilized once at the beginning of the application. This is referring the .csv file in read only. This will be stored as the data member named dataSetFile. After the initial loading of the object, it is likely for ease of access this will become a nested list. This is the only occurrence of a file object.

**String:** Strings are featured commonly throughout the program. This includes all input parameters and items for selection (predefined functions/strings). Strings will be featured in all functions.

**List:** This data structure will be the most commonly occurring structure in this application. There are several resources that will use this data structure being as follows; .csv file, this be translated into a list, The main output will be a list named *result* (depending on the function called)*.* When All() and Search() are called their results will be a nested list. Finally when Analysis() is called this will output a dictionary of lists.

**Dictionary:**  This data structure is also featured often. All of dictionary occurrences will be featured within the functions; Analysis() and Insight(). Analysis’s dictionary output will be dependant on the filter input by the user. Insight will utilize a nested dictionary.

### Detailed Design







****





# User Interface Design

The initial interface design will be brainstormed and then roughly plotted on pen and paper in small informal diagrams to be used as reference for things such as themes, organisation, and user interaction. These sketches will be displayed in the appendix*.*  The primary focus of the interface will be usability and ease of access. Given the software is only to be used for its specific purpose, there will be no need for the design to features any other aspects. Below a structural design has been created to provide a basic design to build further visual elements form. After the initial whiteboarding of the design all mock-ups will be completed via the use of *“draw.io”* an online framework / SketchUp tool used to create GUI and flowcharts.

## Structural Design

The highest priority of this tool is usability and readability. Therefore, there will be strategic use of symbols and labels to identify interaction from the user. A high importance placed upon the returned information allowing for a simplistic “no frills” design elements. The design will consist of two sections being the search, reset and logo operating in the top 25% of the application. The rest of the application will contain returned search results. The data will be presented in different formats being, tables charts or strings.

**Structural sample diagram**

Diagram

Description automatically generated

## Visual Design

Depicted in the figures below is a simple mock-up and basic colour schema that will subject to change based on the project need. The GUI has specifically chosen clashing colours to emphasis on the content and not the application. There is a sample logo of a magnifying glass that can be replaced by the requesting companies branding.

**Sample graphical output figure:**

Graphical user interface

Description automatically generated

3

**Simple table output figure:**

Graphical user interface

Description automatically generated