Software Design Document

Accident Analysis Software

Nicolas Donaldson – S5256284

Juniper Lethbridge – S2884940

Toshimitsu Ota – S5251464

Table of Contents

[1.0 System Vision 3](#_Toc112943594)

[1.1 Problem Background 3](#_Toc112943595)

[1.2 System Overview 3](#_Toc112943596)

[1.3 Potential Benefits 4](#_Toc112943597)

[2.0 Requirements 5](#_Toc112943598)

[2.1 User Requirements 5](#_Toc112943599)

[2.2 Software Requirements 6](#_Toc112943600)

[2.3 Use Cases & Use Case Diagrams 7](#_Toc112943601)

[3.0 Software Design and System Components 11](#_Toc112943602)

[3.1 Software Design 11](#_Toc112943603)

[3.2 System Components 12](#_Toc112943604)

[3.2.1 Functions 12](#_Toc112943605)

[3.2.2 Data Structures / Data Sources 15](#_Toc112943606)

[3.2.3 Detailed Design 16](#_Toc112943607)

[4.0 User Interface Design 18](#_Toc112943608)

[4.1 Structural Design 18](#_Toc112943609)

[Main window 18](#_Toc112943610)

[Chart window 19](#_Toc112943611)

[Dataset window 19](#_Toc112943612)

[4.2 Visual Design 19](#_Toc112943613)

[Layout 19](#_Toc112943614)

[Visual Elements 20](#_Toc112943615)

[Colour 21](#_Toc112943616)

[Font 22](#_Toc112943617)

[Mock-up Screens 24](#_Toc112943618)

# System Vision

## Problem Background

A dataset containing data on Victoria State Accidents presents opportunities to discover underlying causes and other statistics surrounding road accidents that occur in Victoria. Currently, the client, the department of transport for Victoria does not have software to easily access and analyse the data, and present the data to external stakeholders. In addition, there is an increasing need to train new employees on data analysis to understand what is happening in the field through data. With the new software, the client hopes to draw conclusions and present findings, so that they may overcome the challenges associated with data analysis and human error. As data becomes too large, it becomes difficult to analyse the data without human errors, and misunderstandings. The intention for this software is to resolve these issues and also enable the client to utilise the dataset for awareness campaigns and projects in the future.

## System Overview

This system will provide a GUI for the user to visualise the data. The system will accept CSV files and save the data to a database. The system will be able to produce results based on the input of the user. It will also provide various visualisation tools with charts. Different views will include the analysis of traffic accidents related to alcohol, along with a location-based analysis. The system will include features that:

* Displays the information of all accidents that happened in the period for a user-selected period.
* Produces a chart to show the number of accidents in each hour of the day for a user-selected period.
* Retrieves all accidents caused by an accident type that contains a user- entered keyword.
* Provides analysis of alcohol related accidents by comparison between alcohol related and non-related accidents.
* Allow the user to analyse results and filter the result by LGA and/or Region to compare data between different areas with the assistance of charts.

## Potential Benefits

The implementation of this software will alleviates the pressure on humans to avoid producing errors in data analysis. By streamlining common tasks into software functions, the client will benefit increased employee productivities and faster data processing without spending wasteful financial or human resources. Software can discover patterns and statistics that are otherwise difficult for humans to observe from a big dataset. With the quality analysis from the software, the client can save time and financial resources, moreover, the software will improve the quality of trainings for new employees to help make better decisions to serve people by having better understanding of the real world that they will work in.

# Requirements

## User Requirements

As the client is the Department of Transport for Victoria, employees of the Department are the end users of the program. The end user must first begin with inserting the dataset. Once entered, the user may toggle between variations of the user interface. Users can then decide appropriate search parameters for their required query. Once decided, and entered, the results of the user’s query shall be viewable via the output.

In a sequential list of needs, the above would be translated to:

1. Open program
2. Insert Dataset (if first time using application)
3. Insert Query Parameters
4. View Results

Each function will be available from the main view tree in the user interface, with the required function being run dependant on the user input. A field will be available for users to input keywords, which will limit the data output to accident data only containing those keywords.

Options to limit the timeframe will be displayed, if a user decides to use this function the output will be limited to data from within this function. If no time is specified, then all relevant data from the dataset will be displayed.

All data queries will be returned in table format.

Once the data has been displayed, a button to display a chart for the average number of accidents in each hour of the user specified period will appear below the returned data. If this button is selected, a chart will be generated and will be displayed in a new window.

There will also be a menu which will allow the user to specify that they wish to analyse alcohol related incidents. If this menu is selected, a chart will be displayed showing both alcohol and non-alcohol related incidents over the user-specified period in a new window.

In a separate menu, users will be able to view accident data overlayed onto a map of the relevant local government area (LGA). Users will be able to see accidents being displayed as a coloured dot. The colour of each dot is determined by their accident type, a legend will be displayed to the user explaining the colour correlation. Users will be able to filter the data displayed on the map by region/LGA/accident type etc.

In addition to this, the end user should be aware that their query is being processed. To provide this context, for each query a loading prompt (i.e. loading bar, circle etc.) will be displayed to make the end user aware that their query has been accepted and the system is working on displaying their query.

## Software Requirements

1. The program shall have a GUI implementation.
   1. The program shall allow for user input through GUI.
   2. The program shall accept a csv dataset file from user input.
   3. The program will limit returned data by user input.
   4. The program will limit returned data by time range.
   5. The program shall graphically display data through charts & tables.
   6. The program shall output a summary of results.
2. The program shall have a database via sqlite3 library.
   1. The program shall allow for users to upload a .csv dataset file to the database
   2. The program shall convert .csv files into a database
   3. The program shall perform SQLite queries
3. The program shall have maths & statistical modules.
4. The program shall have a datetime module.
5. The program shall have a PyPlot module.

## Use Cases & Use Case Diagrams

|  |  |
| --- | --- |
| Use Case | Display all accident information |
| Primary Actor | User |
| Secondary Actor | Database |
| Description | The user specifies a time-period. The system then retrieves all accident information from this period and displays it in a tabular format |
| Flow of events | User specifies a start date and an end date, and after pressing the ‘search’ button, all accident data within the user-specified period is retrieved from the database and presented in a tabular format. |
| Trigger | The client requires to view accident data within Victoria during a specific time-period. |
| Pre-condition | The client knows the period they wish to view and data from within this period is within the database. |
| Post-Condition | Data from the database has been queried and displayed by the program. |
| Success Scenario | Accident information from within the user-specified period is successfully displayed within the program. |
| Abort Scenario | Data from within the user-specified period is not in the database. |

|  |  |
| --- | --- |
| Use Case | Display average hourly accident rate |
| Primary Actor | User |
| Secondary Actor | System |
| Description | From data retrieved within the user-specified time-period the average hourly accident rate is calculated and presented graphically as a chart. |
| Flow of events | The user has already retrieved data from within a time-period they have specified. The user then clicks a button which calculates the average hourly accident rate within the period and generates a chart to display the data. |
| Trigger | The user requires to analyse the hourly accident rates within a specified time-period. |
| Pre-condition | A set of data has been successfully retrieved from the database and is currently being displayed within the program. |
| Post-Condition | The hourly accident rates have been calculated and the system has accurately displayed these rates as a chart |
| Success Scenario | The hourly accident rate from within the user-specified time-period has been displayed as a chart. |
| Abort Scenario | There is not enough data within the timeframe to display an average accident rate for a 24hr period. |

|  |  |
| --- | --- |
| Use Case | Display accident data containing user-specified keywords |
| Primary Actor | User |
| Secondary Actor | System |
| Description | The user enters a keyword which limits the retrieved data to only records containing that keyword within the user-specified period. |
| Flow of events | User inputs a keyword into the relevant text field. The database is queried with this keyword along with the user-specified timeframe and all matching records are displayed in a tabular format. |
| Trigger | The user requires to view accident data with a relevant keyword and within a specific timeframe. |
| Pre-condition | Data is available from within this timeframe and containing the relevant keyword(s). |
| Post-Condition | The data has been successfully queried and all relevant records have been displayed. |
| Success Scenario | Records within the user defined parameters are displayed |
| Abort Scenario | No records are found with the user defined parameters |

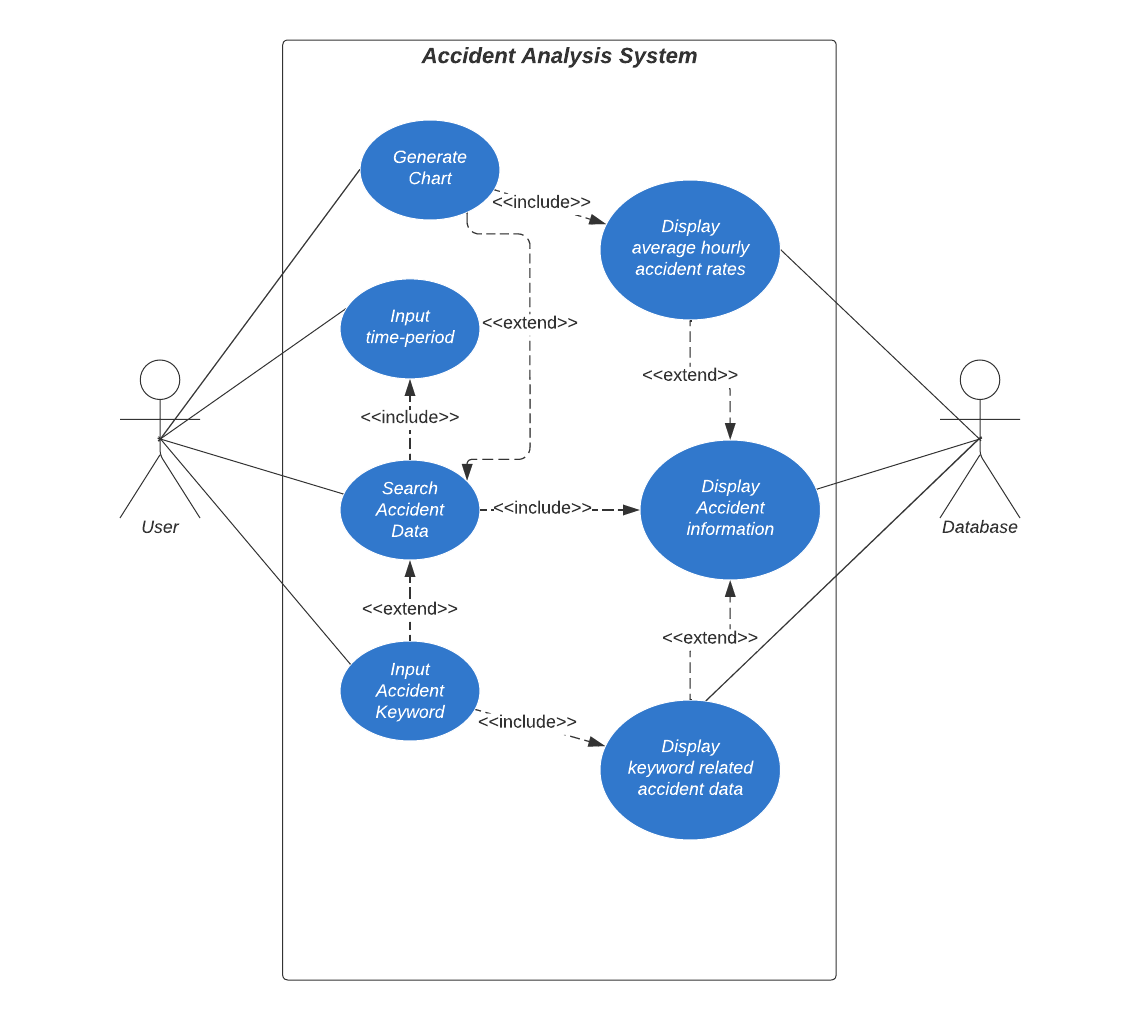


Diagram 1: Use Case Diagram for Functions 1,2,3

|  |  |
| --- | --- |
| Use Case | Display all alcohol related incidents within a user-specified time |
| Primary Actor | User |
| Secondary Actor | System |
| Description | All alcohol related accident data within a user-specified period is displayed. |
| Flow of events | The user selects the alcohol related accident tab and specifies a time-period from which data will be retrieved from. This data is then retrieved from the database and displayed in a tabular format. |
| Trigger | The user requires to view all alcohol related incidents within a specified time-period. |
| Pre-condition | The user requires to view all alcohol related incidents within a specified time-period. |
| Post-Condition | Nil |
| Success Scenario | All alcohol related records within the user-selected period are displayed |
| Abort Scenario | No records are found within the user defined parameters |

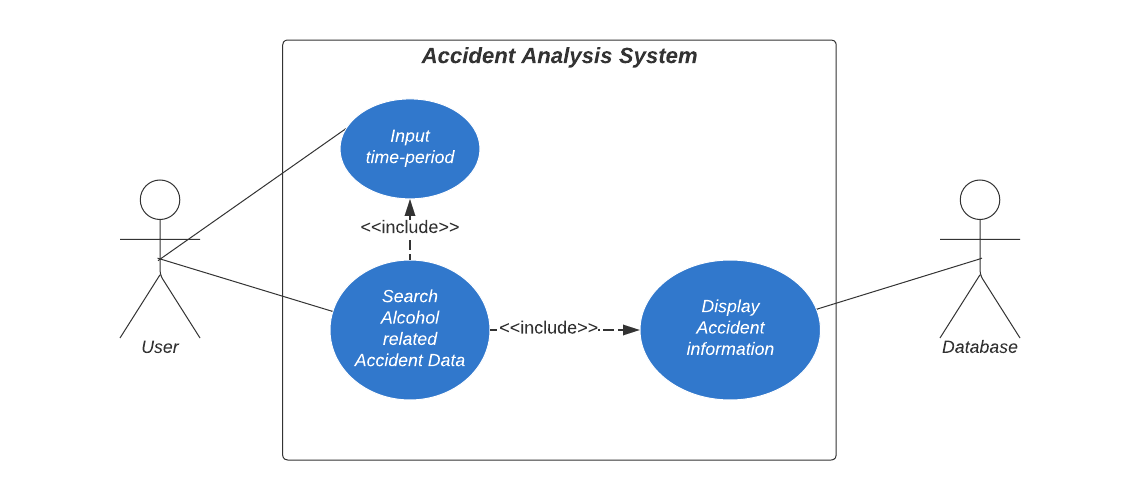


Diagram 2: Use Case Diagram for Function 4

|  |  |
| --- | --- |
| Use Case | Visually overlay geographical accident data |
| Primary Actor | User |
| Secondary Actor | System |
| Description | Accident data is overlayed onto a map of the local government area (LGA). Users can filter results by LGA, region or accident type. |
| Flow of events | Accident data is retrieved from the database and visually overlayed onto a region map. This map can be further filtered by the user by accident type, LGA and/or region. |
| Trigger | The user requires to view the accident data visually by geographical location. |
| Pre-condition | Records exist within the database containing geographical location. |
| Post-Condition | Accident data has been retrieved from the database and overlayed onto a region map of the relevant areas. |
| Success Scenario | An overlay of records is displayed on a map of the relevant area. |
| Abort Scenario | No records exist within the database. |

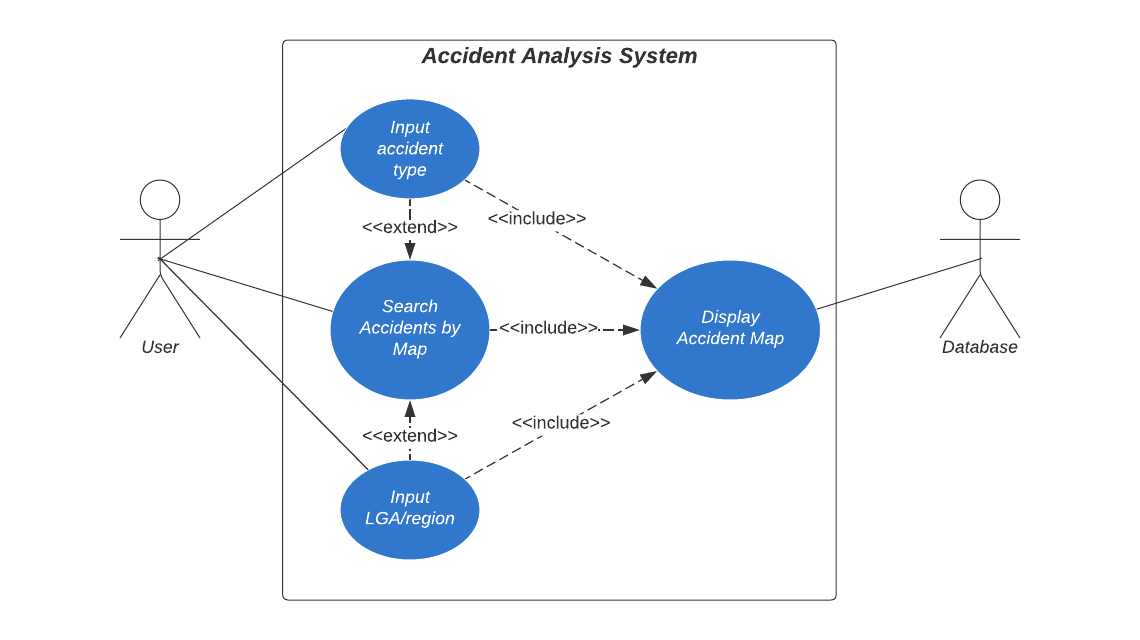


Diagram 3: Use Case Diagram for Function 5

# Software Design and System Components

## Software Design

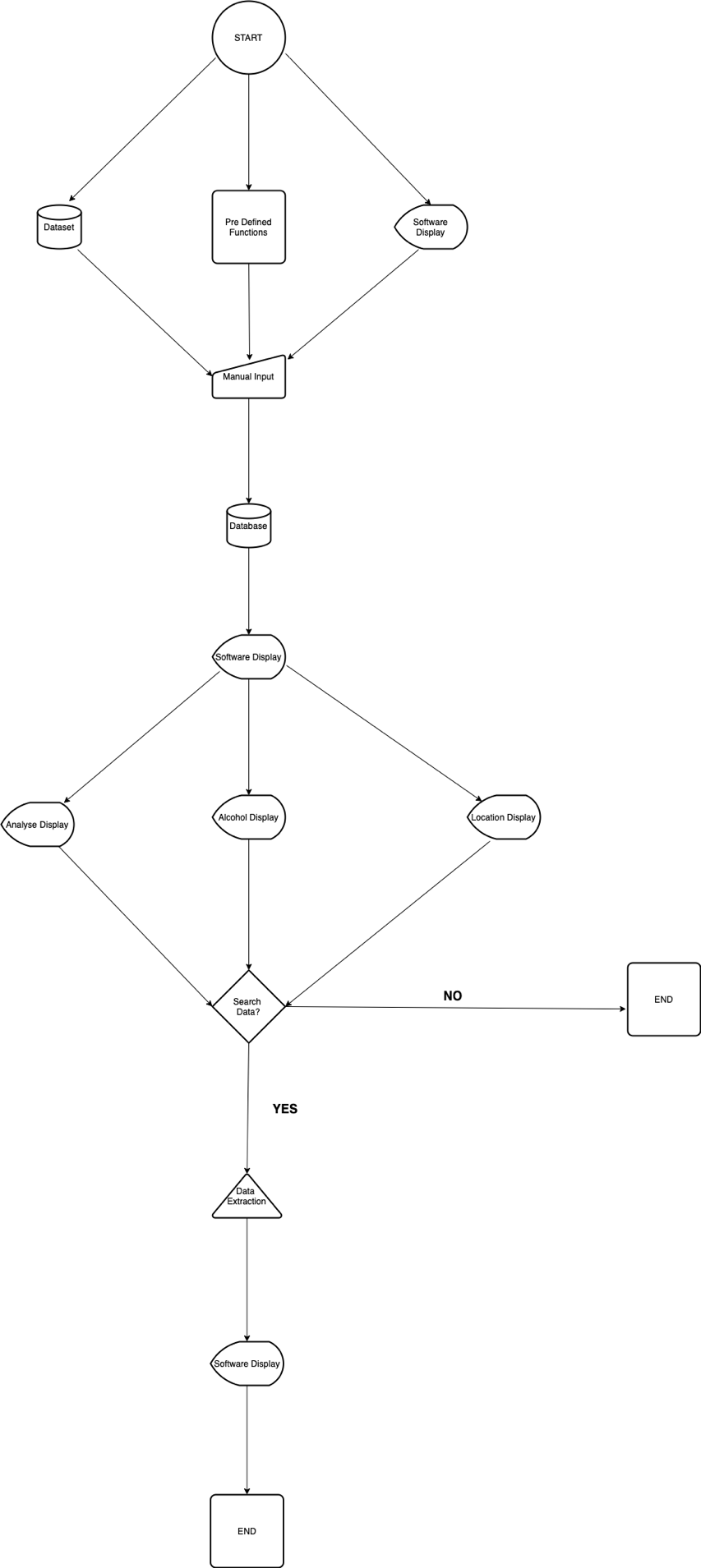


Diagram 2: System Design Flowchart

## System Components

### Functions

#### Import dataset

Reads a selected CSV dataset file and validates to see if it contains required data fields. Return the data in tuple format.

**Input Parameter:**  
File location (String) to determine path to the file.  
File name (String) to determine the file to read.

**Return value:** Accidents data (List of Tuple).

**Side effect:** None.

#### Create database

Connects to SQLite and creates a database and tables.

**Input Parameter:** None.

**Return value:** None.

**Side effect:** Creates a global database object.

#### Save to database

Saves the accidents data to the SQLite database.

**Input Parameter:** Accidents data (List of Tuple).

**Return value:** None.

**Side effect:** Modify database data.

#### Retrieve from database

Reads data from database and return accident data for the defined criteria.

**Input Parameter:**   
From date (String) to set the date range to search accidents after the date.  
To date (String) to set the date range to search accidents before the date.  
Accident type keyword (String) to search accidents with accident type which include the keyword.  
Accident type (String) to search accidents with the accident type.  
LGA (String) to search accidents only in the LGA.  
Region (String) to search accidents only in the region.  
Output type (List of Int) to determine which field to show.

**Return value:** Accident data (List of Tuple).

**Side effect:** None.

#### Calculate hourly average

Calculates the average number of accidents in each hour of the day.

**Input Parameter:** Accident data (List of Tuple) to use for the calculation.

**Return value:** Average for each hour (List).

**Side effect:** None.

#### Calculate accident type

Calculate the number of accidents in each accident type.

**Input Parameter:** Accident data (List of Tuple) to use for the calculation.

**Return value:** Total number of accidents for each accident type (List).

**Side effect:** None.

#### Calculate by month

Calculates the number of accidents in each month.

**Input Parameter:** Accident data (List of Tuple) to use for the calculation.

**Return value:** Total number of accidents for each month (List).

**Side effect:** None.

#### Calculate by day

Calculates the number of accidents in each day.

**Input Parameter:** Accident data (List of Tuple) to use for the calculation.

**Return value:** Total number of accidents for each month (List).

**Side effect:** None.

#### Calculate LGA

Calculates the number of accidents in each LGA.

**Input Parameter:** Accident data (List of Tuple) to use for the calculation.

**Return value:** Total number of accidents in each LGA (List).

**Side effect:** None.

#### Calculate region

Calculates the number of accidents in each region.

**Input Parameter:** Accident data (List of Tuple) to use for the calculation.

**Return value:** Total number of accidents in each region (List).

**Side effect:** None.

#### Generate bar chart

Generates a bar chart with legend.

**Input Parameter:** Labels and values (dictionary) to place in the chart.

**Return value:** File path of the generated image (String).

**Side effect:** None.

#### Generate map chart

Generates a map chart to show the locations of accidents on a Victorian map colour coded by fatal or injury accident with legend.

**Input Parameter:** List of accident kind and geo coordinate (List of Tuple) to place dots in the chart.

**Return value:** File path of the generated image (String).

**Side effect:** None.

#### Display main

Creates a main window to display for users to perform data search.

**Input Parameter:**  
Analysis type (String) to determine which component to place in the window.

**Return value:** None.

**Side effect:** Generate GUI components.

#### Display chart

Creates a window to display a chart image with a title.

**Input Parameter:**  
Title (String) to place in the window.  
File name of chart image (String) to place the image in the window.

**Return value:** None.

**Side effect:** Generate a GUI component.

### Data Structures / Data Sources

|  |  |  |
| --- | --- | --- |
| Data | Structure | Description |
| CSV dataset file | CSV Table | This file contains accident data from Department of Transport and saved as CSV file. This will be imported to the software. The dataset file should contain columns of ACCIDENT\_NO, ACCIDENT\_DATE, ACCIDENT\_TIME, ACCIDENT\_TYPE, DAY\_OF\_WEEK, SEVERITY, LONGITUDE, LATITUDE, LGA\_NAME, REGION\_NAME, FATALITY, SERIOUSINJURY, ALCOHOL\_RELATED.  Related function: Import dataset |
| Database | Table | This file will be saved when the software imports the CSV dataset file. Analysis in the software will use this data. The attribute will include ACCIDENT\_NO, ACCIDENT\_DATE, ACCIDENT\_TIME, ACCIDENT\_TYPE, DAY\_OF\_WEEK, SEVERITY, LONGITUDE, LATITUDE, LGA\_NAME, REGION\_NAME, FATALITY, SERIOUSINJURY, ALCOHOL\_RELATED.  Related function: Create database, Save to database, Retrieve from database |
| Menu | Array | This is to determine which content to display and set different type of analysis to perform. The members will be dataset, analyse, alcohol, and location.  Related function: Display main |
| Analysis type | Array | This is a list to contain types of analysis which includes general, alcohol, and location. This data will be used to determine attributes for search and which type of chart to generate. The members will be general (default), alcohol, and location.  Related function: Retrieve from database, Display main |
| Search attribute | Class | This is to determine which search criteria will be available for the analysis functions to search the dataset. Available attributes will be different depending on which analysis type is chosen. From data, to data, accident type keyword, accident type list, and output type will be the members for the general analysis function.  Related function: Retrieve from database, Display main |
| Accident type | List | This data will be generated from JSON dataset file referencing the values in the ACCIDENT\_TYPE field. This list will be used by the search attribute to perfume an analyse search. Related function: Retrieve from database, Display main |
| Output attribute | key-value dictionary | This data to store the preference of which fields (attributes) to show in the output of analyse search in the table view. True/False Boolean values will be stored for each output type.  Related function: Retrieve from database, Display main |
| Summary values | key-value dictionary | This data contains total values of records, fatality, and injury which are generated by the search dataset function. This will be used to display search result summary.  Related function: Display main |

### Detailed Design

The following pseudocodes demonstrate how each non-standard function in the software will perform:

#### Import dataset

Import csv and sqlite3 python packages.  
Input file name and location.  
If the file exists, examine the file format and file content of the CSV dataset file.  
Else error: File does not exist.  
If file format is not CSV and it does not contain required fields, error: File type is not supported.  
Connect to SQLite and create a database and tables.  
Insert data into database tables.  
Commit and close database connection.

#### Search dataset

Import sqlite3 python package.  
Connect to SQLite database.  
If date range is set in the search attributes, find records within the date range.  
If accident type is set in the search attributes, find records with the accident type.  
If the found records is more than zero, output values with only output attributes that are defined True.  
Else display a message: No record found.  
Calculate summary values (total records, fatality total, injury total) to display.  
Close database connection.

#### Generate bar chart

Import library for chart creation (PyPlot library).  
Check which type of chart (chart type) selected.  
Read the search result records generated from the search dataset function.  
Calculate total value for each attribute.  
Allocate colours for attributes and set labels.  
Generate a bar chart.  
Display the chart with legend.

#### Generate map chart

Import library for chart creation (PyPlot library).  
Read the search result records generated from the search dataset function.  
Allocate colours for attributes and set labels.  
Convert LONGITUDE, LATITUDE values in the records to X, Y values to place in the chart.  
Draw circles with the X, Y values with setting colours depending on the accident type.  
Generate a bar chart with a background Victorian map image.  
Display the chart with legend.

# User Interface Design

This section illustrates how the software interact with users through GUI components and graphics with justifications of design decisions made. The structure of user interface components is also defined. Adobe XD is utilised to design the mock-up screens to present the user interface design for the software.

## Structural Design

The software has the main window as the top hierarchical level. The structure of the main window is designed to indicate steps that users follow by placing sections from top to bottom. The estate of main window is limited; thus, charts will be displayed in a secondary window to create more display surface.

### Main window

The main window of the software contains the following sections. To provide the best useability, they are grouped by action which are navigate, search, result (view), summary (view), and chart (view).

**Navigation tool bar**

This section is located at the top of the main window and contains buttons to perform the main functionalities. They are Dataset, Analyse, Alcohol, and Geographical.

**Data search box**

This section is located under the navigation tool bar and contains input fields for users to define search criteria such as dates, and accident type along with a search button to initiate data search.

**Data table view**

This section displays output data in a table view. It is located under the data search box and users can see search result.

**Chart selection bar**

This section is located at the bottom of the main window under the data table view. It shows available chart types which users can view by clicking. This section will not be available when there is no available option.

**Summary view**

This is located at the right side next to the data search box to show summary of result from a search. It can include number of records found, total number of target records, and total number of injury/fatality.

### Chart window

A secondary window will open when users select one of options in the chart selection bar. It displays chars with selected conditions. This window is independent from the main window and can be placed anywhere in users’ screen. Users can close this window without affecting the main operation of the software.

## Visual Design

### This section demonstrates how components will be placed and what types of graphic elements will be used in the software to support user interface. The purpose of the use of colours and graphics is to create the modern friendly look and feel for the use in new staff trainings.

### Layout

The following depicts the layout of the structural components in the main window.

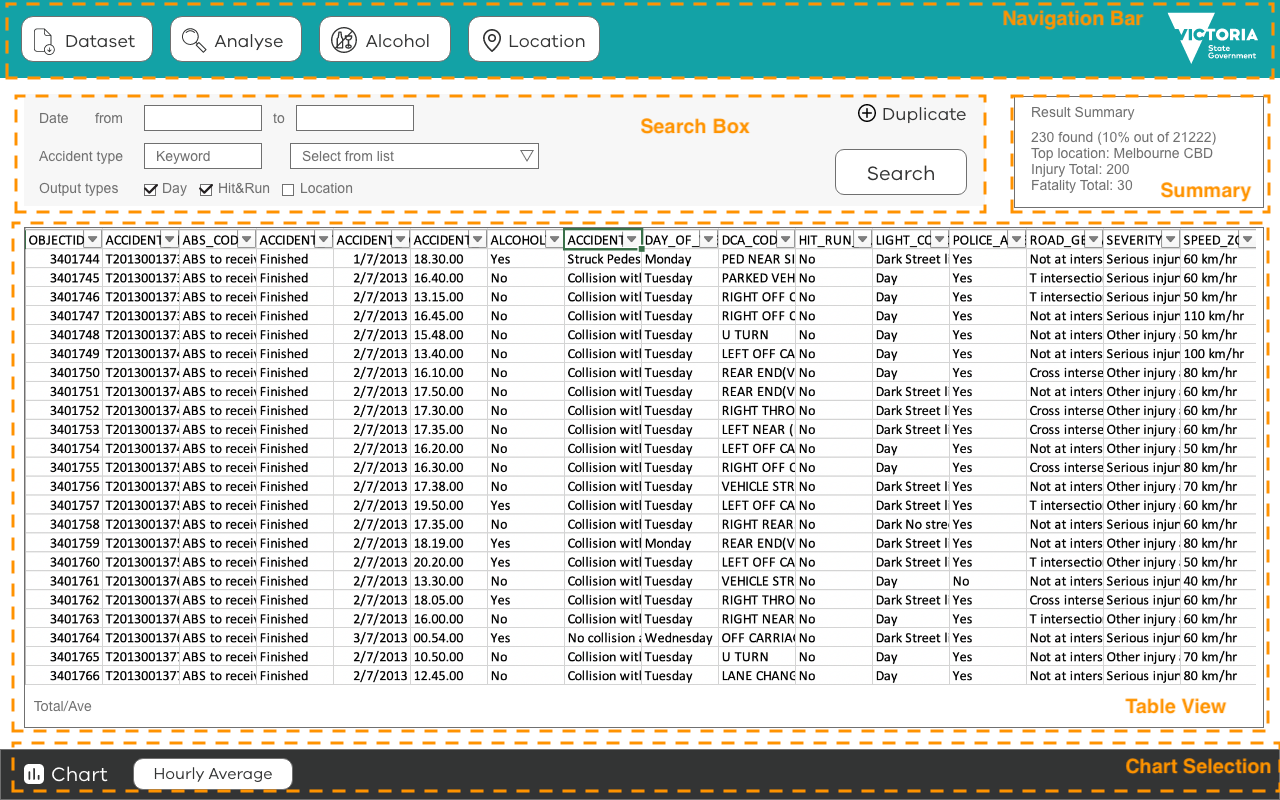


Figure 1 Main Window Layout

### Visual Elements

#### Logo

The client’s logo, Victoria State Government logo, will be placed in the navigation bar sourced from Brand Victoria website (<https://www.vic.gov.au/brand-victoria-using-our-logos>). The colour of the logo will be adjusted to white to have a contrast against the background colour.



Figure 2 Victoria State Government logo

#### Icons and Buttons

|  |  |  |
| --- | --- | --- |
| Graphic | Name | Description |
|  | Dataset Icon | Used in the button for the Dataset menu to indicate importing a file. Open a dialog to select a file to import to use as a dataset. |
|  | Analyse Icon | Used in the button for the Analyse menu to indicate to set search conditions. Displays the standard search box to analyse data. |
|  | Alcohol Icon | Used in the button for the Alcohol menu to indicate the data analysis of alcohol related accidents. Displays the extended search box to analyse alcohol related accidents data. Slashed version is selected to be work environment friendly instead of non-slashed designs. |
|  | Location Icon | Used in the button for the Location menu to indicate the geographical data related analysis using maker which is used in map apps often. Displays the extended search box to analyse accidents data by area or location. |
|  | Chart Icon | Used to supplement the Chart title in the chart selection bar to indicate the section is about charts. |
|  | Search Button | Buttons have round corners to indicate the actions are assigned to them as well as create the soften the impression of the software. |
|  | Navigation menu button | Buttons have round corners and a label with an icon to indicate that they are main menus in the software. They are placed in the navigation bar at the top. The size is bigger than other elements to stand out. |

### Colour

The most key colours for the software are chosen as same as the Department of Transport Victoria website to be cohesive and adhere the client’s corporate ID (see figures below).

|  |  |  |
| --- | --- | --- |
|  | Colour Name (Code) | Description |
|  | Teal (#13a2a6) | The feature colour that is used for the Department of Transport Victoria website. This colour is used for the background of the navigation bar in the software. |
|  | Dark grey (#333434) | The colour is used for the background of the chart selection bar at the bottom of the software. It is the same colour that is used for the client’s website as the footer background colour. It is also used as the text colour of menu buttons and table view body content. |
|  | Grey (#707070) | This colour is used for the secondary text labels such as labels in the search box and the result summary box. Its shade is lighter than the dark grey (#333434) while it holds the decent readability. |
|  | Light grey (#F7F7F7) | This colour is used for the background of the search box to indicate the distinct area and separate from adjacent sections. The shade of this colour is adjusted to keep enough contrast with text elements in the section for maximum readability. |
|  | White (#ffffff) | This colour is used for the background of the majority space in the software to deliver the clean look and feel. It is the same colour which has been used for the background of the client’s website. It is also used for the colour of the buttons in the navigation bar to make a contrast with the navigation bar background colour. |

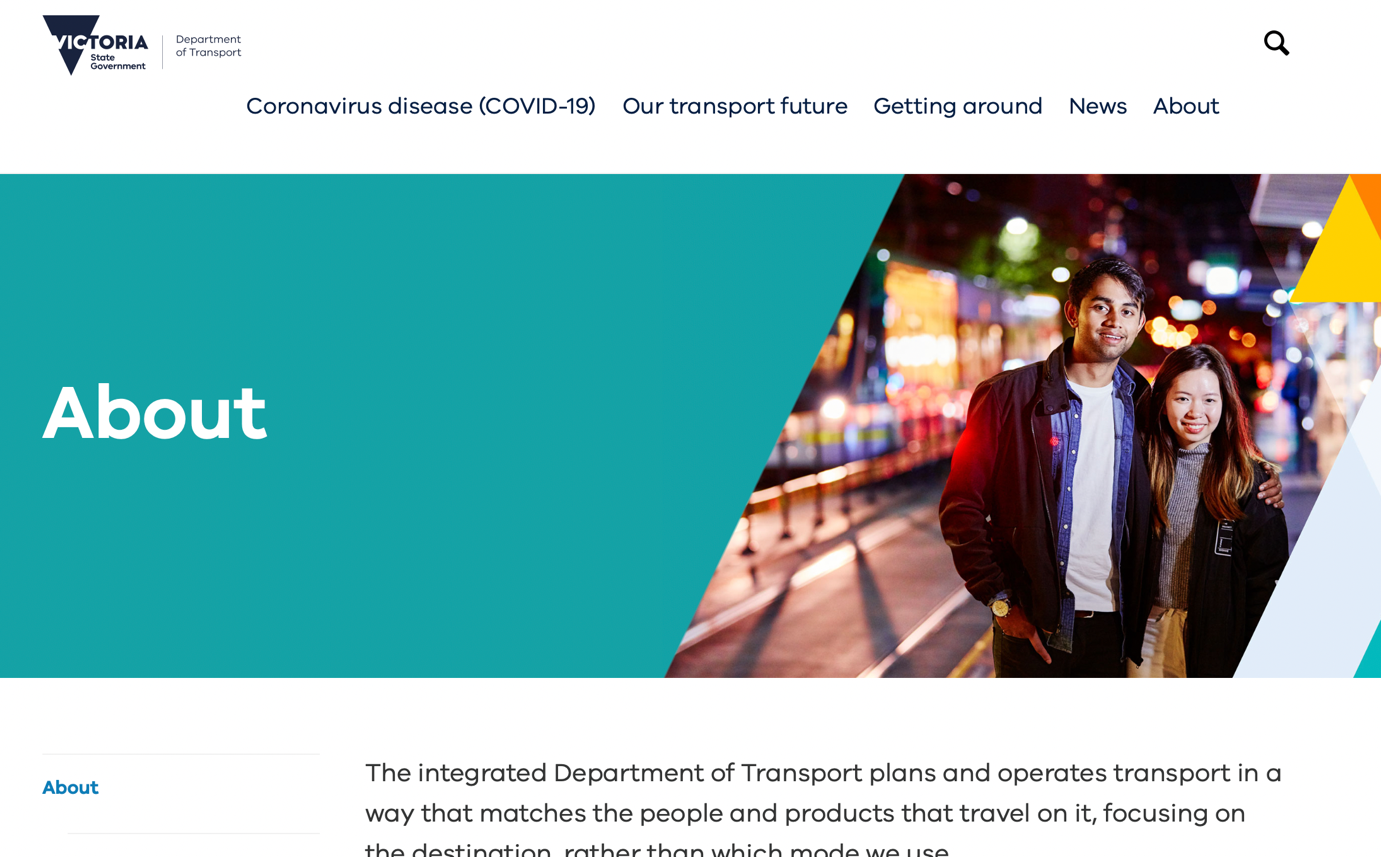


Figure 3 Department of Transport Website

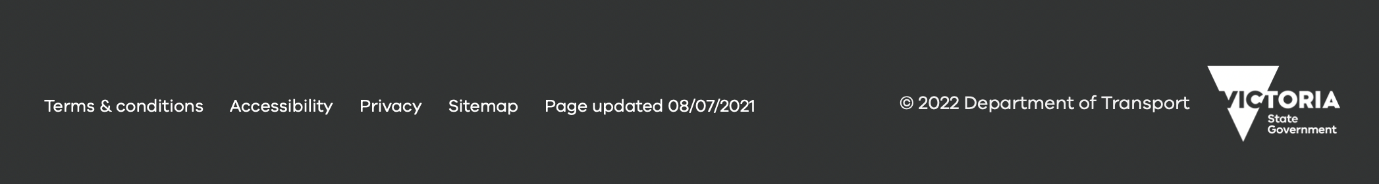


Figure 4 Department of Transport Website Footer

### Font

#### Feature Font

The VIC font will be used for headings and labels of buttons in the software. This is to adhere the design guidelines defined by the client, Victorian Government. The font files are available on vic.gov.au website (<https://www.vic.gov.au/brand-victoria-fonts>) for download to use.

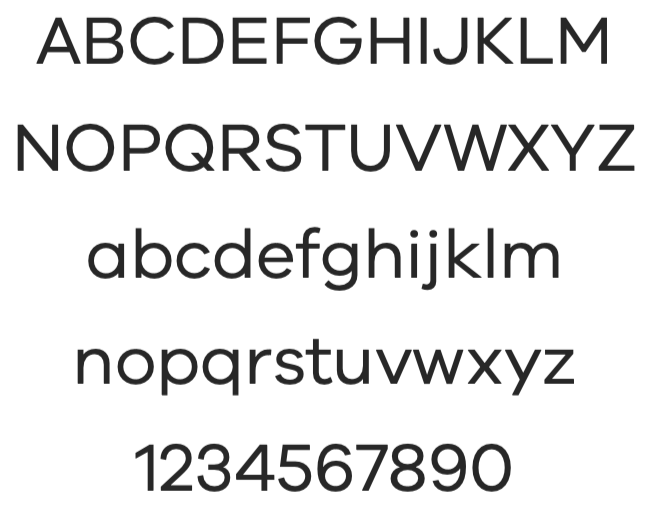


Figure 5 VIC font

#### Body font

Arial font will be used for body text elements as instructed in the font usage guide on the client’s website.

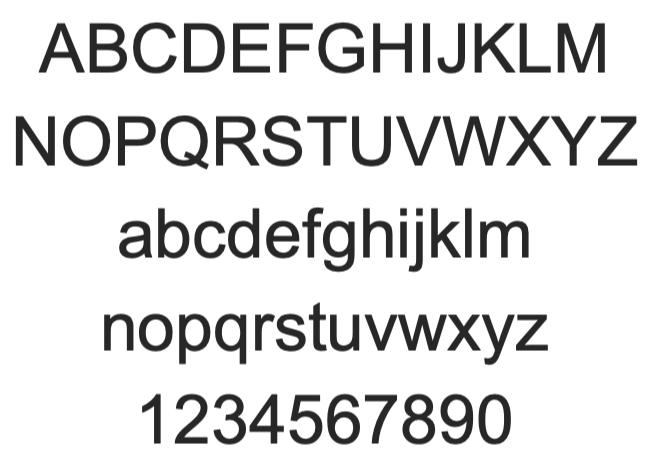
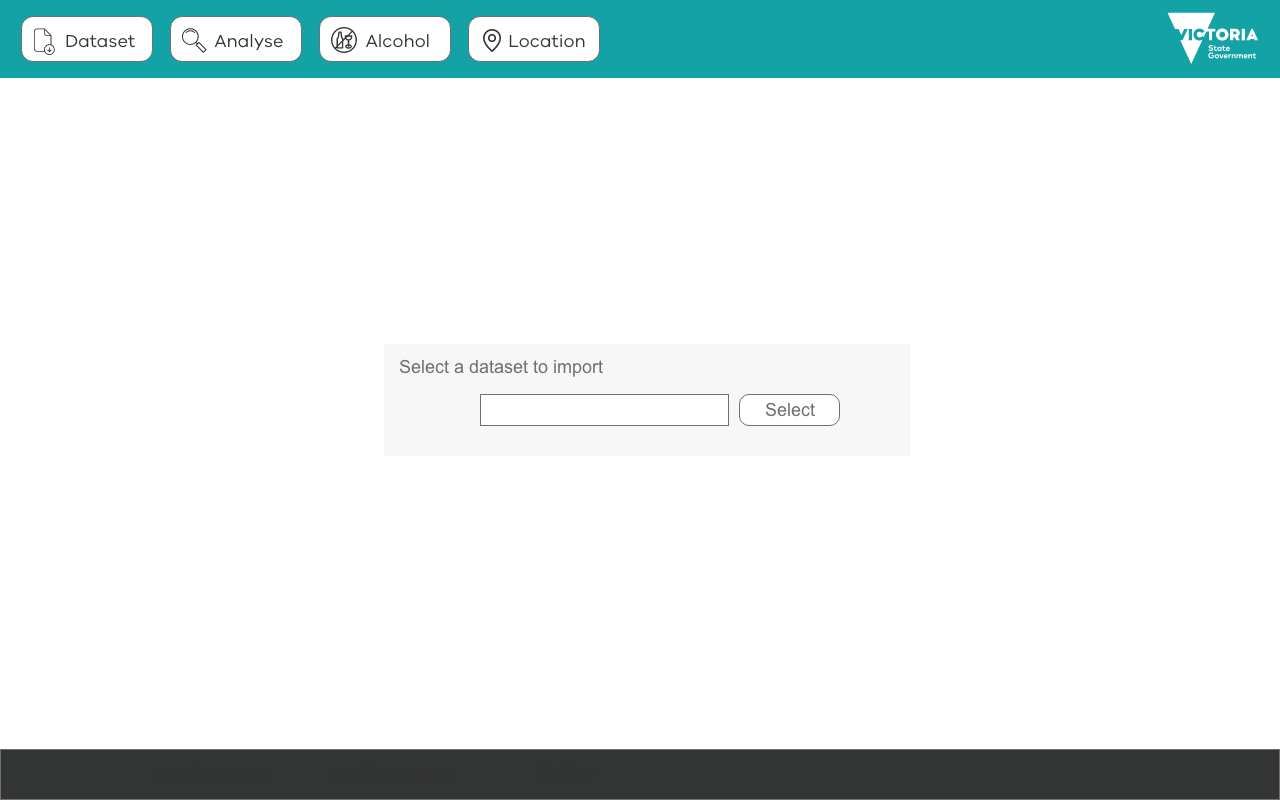


Figure 6 Arial font

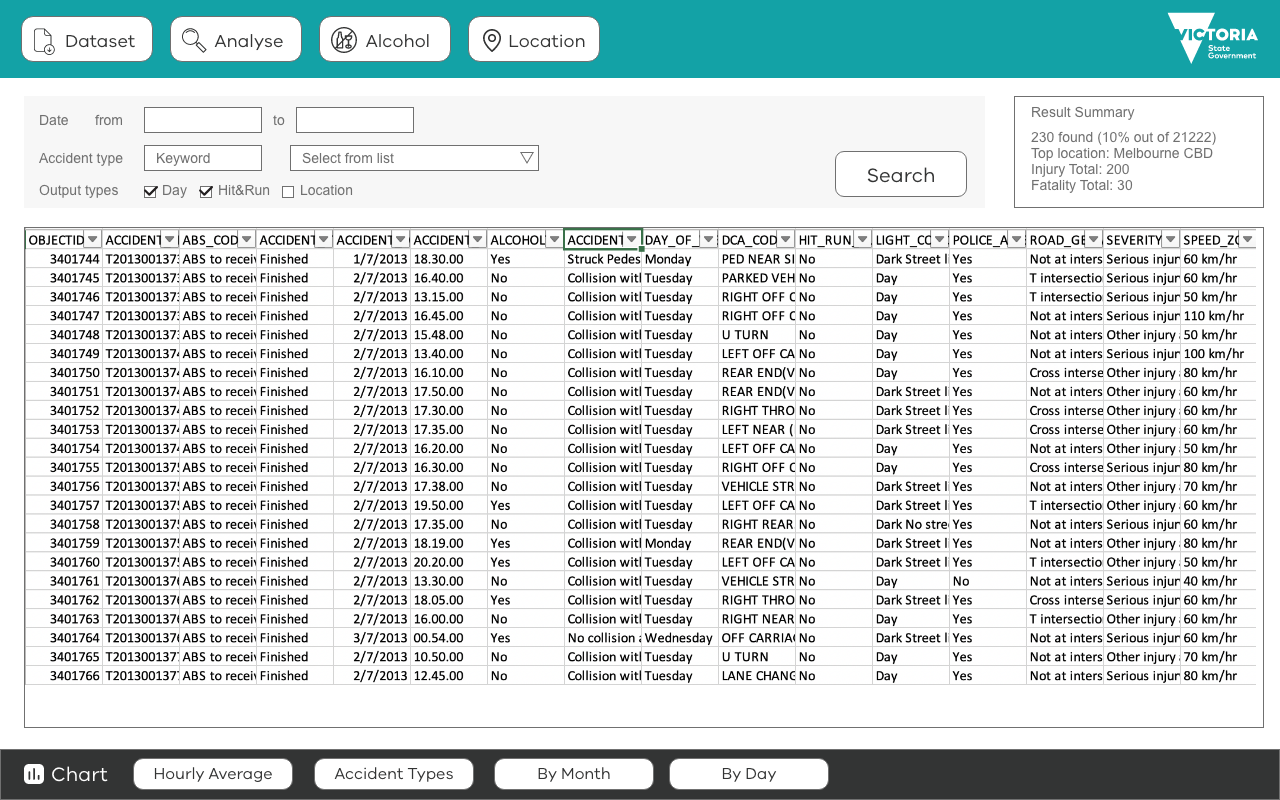
### Mock-up Screens



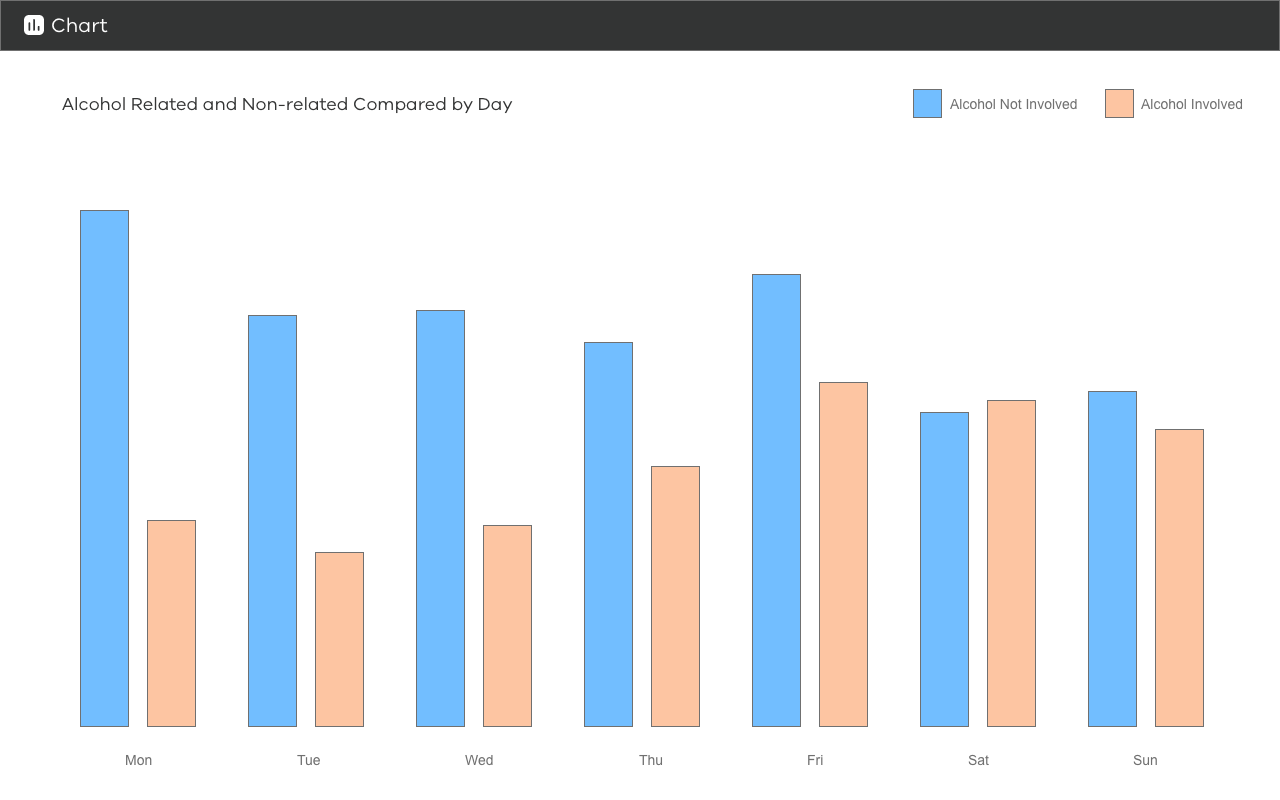
Screen 1 Dataset Selection



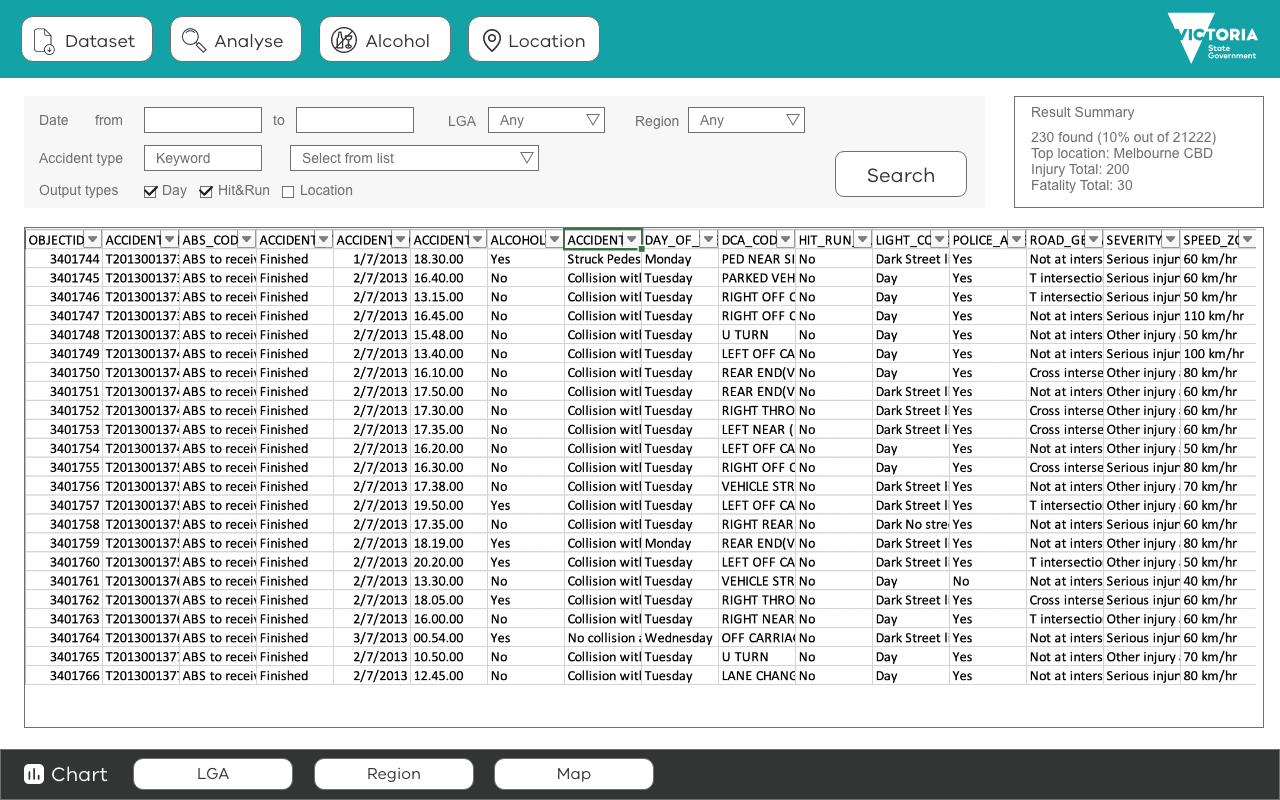
Screen 2 Analyse



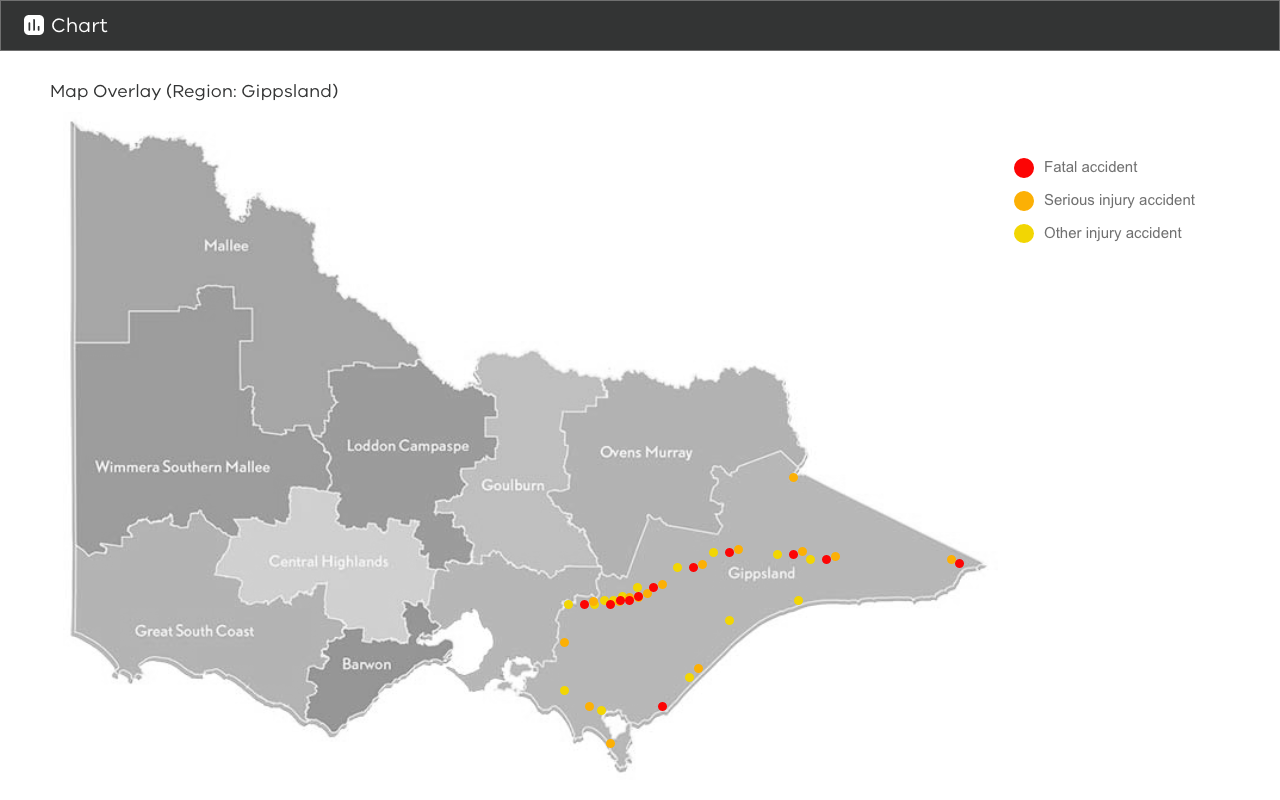
Screen 3 Alcohol Section



Screen 4 Chart Alcohol Related



Screen 5 Location



Screen 6 Chart Map