Software Testing Report

UI implementation for Victoria state accident database

Manish Saily s5236772

Edward Winston s5133176

Table of Contents

[1.0 Unit Tests 3](#_Toc49779837)

[2.0 Coverage Report 4](#_Toc49779838)

[3.0 Requirements Acceptance Testing 5](#_Toc49779839)

# Unit Tests

| **No** | **Test Case** | **Expected Results** | **Actual Results** |
| --- | --- | --- | --- |
| **1.0** | **Correct file (CSV) upload** | **File is read and generate message (“CSV file uploaded successfully!”)** | **File is read and generate message (“CSV file uploaded successfully!”)** |
| 1.1 | Test incorrect file format | Error message and not accept file | Error message and not accept file |
| 1.2 | Changing file to another | **File is read and generate message (“CSV file uploaded successfully!”)** | **File is read and generate message (“CSV file uploaded successfully!”)** |
| **2.0** | **accepts Accident type input** | shows the data in table format | shows the data in table format |
| 2.1 | No input entered for accident type | Prompt user to input accident type | Prompt user to input accident type “Please enter an accident type” |
|  | No input entered for year | Use default 2013 | Uses default 2013 |
| 2.2 | No matched input for accident type | Show empty table | Show empty table |
| **3.0** | Display data for selected year | Display table of information | Display table of information |
| 3.1 | Select any year from drop down menu | User inputed value will be inputed into the variable called selected\_year | User inputed value will be inputed into the variable called selected\_year |
| 3.2 | All headings change according to the selected year | **Correct display of charts from the chosen year** | **Displayed graph** |
| **4.0** | **Display graph for speed zones** | **Graph generated** | **Graph generated** |
| 4.1 | Display correct chart for speed zone accidents | **Once the speed zone button is clicked it should show a speed zone chart** | **Correct chart shown** |
| **5.0** | **Display graph for accidents per hour** | **Generate a graph for accidents per hour** | **Correct graph generated** |
| **6.0** | **Display chart for alcohol impacts** | **Generate a graph for alcohol impacts** | **Correct graph generated** |

# Coverage Report

We have created the code as each button of the user interface is a function. This helps to better keep track of all the components of the code and make easy adjustments without breaking other parts. Every time a button or unput is entered the matching function is called.in total five functions were used in the development of the code and were tested in manually. The main function that requires a true or a false is the read csv file function. As the UI will not begin until the user has uploaded a csv file condition had to be met for the file error, wrong file type, and accepting the correct csv file. Streamlit has a built-in function where it is not possible to upload a file that is not a csv. Using st.file\_uploader() and specifying type=[’csv’] the UI will now to only accept csv files. Once the fie is uploaded the user is prompted with a successful file upload prompt and the UI is ready to use. Another function that requires a true or a false is the accident type input. When the user types the correct input for accident types, they are shown chart containing all the accident of that type. If the show accident type button is pressed without an input the user is prompted to enter the input. The rest of the tests were conducted after the file was uploaded and buttons were clicked manually. Using streamlit’s if st.button() when the button on the screen is pressed it will call the relevant function to display all the information..

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

# Requirements Acceptance Testing

| **Software  Requirement No** | **Test** | **Implemented (Full /Partial/ None)** | **Test Results (Pass/ Fail)** | **Comments (for partial implementation or failed test results)** |
| --- | --- | --- | --- | --- |
| 1 | The user interface needs to be able to access the data set. | Full | Pass |  |
| 2 | The user interface needs to be able to read the data using pandas .read\_csv(). | Full | Pass |  |
| 3 | The user interface needs to be able to filter the data based on the year. | Full | Pass |  |
| 4 | The user interface needs to have a drop-down menu for the user to select the time period. | Full | Pass |  |
| 5 | The interface needs to be able to read user inputs for keywords. | Full | Pass |  |
| 6 | The interface needs to be able to produce charts for the users. | Full | Pass |  |
| 7 | The interface needs to be able to produce relevant tables. | Full | Pass |  |
| 8 | The interface needs to be able to calculate the percentage of alcohol involvement per type of accident. | None |  | We decided to provide a pie chart for better visualisation of percentage of alcohol impacted accidents. |
| 9 | The interface needs to allow the users to return to the starting page. | Full | Pass |  |
| 10 | The interface needs to be able to produce a table to show the data the user searched for using the search engine. | Full | Pass |  |
| 11 | The program will use streamlit to create the user interface. | Full | Pass |  |
| 12 | Streamlit will create the UI through a local host meaning that installation of streamlit through miniconda is required. | Full | Pass |  |
| 13 | Streamlit will also be used to produce all the relevant charts and table for the user. | Full | Pass |  |