

**School of Engineering and**

**Information Technology**

**ICT365 MAJOR ASSIGNMENT ONE**

**CHECK LIST**

**Surname (Family Name):** \_\_Panicciari\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Given Names:** \_\_Mitchell\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Student Number:** \_\_\_\_32674186\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Tutor’s Name:** \_\_\_Giles Oatley\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Assignment Due Date: \_\_01/10/2017**\_\_\_\_\_\_\_\_\_\_ **Date Submitted: \_\_01/10/2017**\_\_\_\_\_\_\_\_\_

**Your assignment should meet the following requirements. Please confirm this (by ticking boxes) before submitting your assignment.**

✔All details above are complete.

✔I acknowledge and agree that the assessor of this assignment may, for the purpose of assessing this assignment, reproduce this assignment and provide a copy to another academic staff member.

✔I am aware of Murdoch University’s assessment policy 8.1.8 that states that “Unit Coordinators have the right to submit any assignment to the University’s plagiarism detection software if they suspect plagiarism.”

✔I have read and understood the requirements for submission of assignments specified in the Unit Information and Learning Guide.

✔I have read and understood the requirements for documenting and submitting this assignment that are specified in the assignment question sheet of this assignment.

✔You understand that the archive file should be submitted to ICT365 Unit LMS.

✔You have kept a copy of this assignment, including the archive file.

# Detailed Description of the Problem

**Task 01:** The first task is to consider the events shown in the example SOAP xml file, from there add several more events following the same formatting and layout. Adding in at least 20 events and ensuring that at least two of them are a track log.

**Task 02:** Create a new windows form which will be the main GUI for the application, a key part of this form must be that it is a canvas for an area or map to be displayed from which your events would be shown from the SOAP file. Having a windows form with the background set to an image of a map would allow for the events to be plotted and make it look realistic.

**Task 03:** The main aim of task 3 is to use LINQ to XML in order to load in the events in the SOAP file and then to store them in an appropriate collection of objects. Once the XML file has been read in and all of the elements needed are selected the objects can be stored and organised using a C# dictionary which allows for a key and an object, the key is the reference to the object allowing you to access the object at a later time. An appropriate class or classes need to be developed in order to handle the creation of the events which will then be stored in the object. This task is important to ensure that each instance of an event is stored and easily accessible.

**Task 04:** Once the events have been stored within their appropriate objects it is now time for them to be displayed with appropriate icons or text on the canvas. Each event object needs to be read in and then determined what kind of event it is in order for the correct icons and text to be plotted onto the map.

**Task 05:** Task 5 is broken up into a couple of different tasks being the option to add new events or retrieve events which are already stored in the xml file. If the user chooses to add a new event a sub form will be shown which allows the user to enter the appropriate information for the specific event, when they have finished and press add the new event is stored into the SOAP file. If the user decides they wish to retrieve the information of an event they have the ability of clicking on the events icon and from there information pertaining to that event will be displayed, a method needs to be developed which differentiates each of the events from one another. When an event is chosen the information related to that event must be displayed for example the contents of the Facebook status update, also a radius is to be drawn around the event to see if it interlinks with any of events or comes close.

# Self-Diagnosis, Evaluation and Declaration

**Features fully implemented:**

* **Task 1:** My SOAP xml file contains at least 20 different events and includes two different track logs. Each event follows a specific structure and is all formatted correctly.
* **Task 2:** My GUI is a picture of some suburbs in Perth which is serves as the canvas for the application.
* **Task 3:** I use LINQ to XML in order to load in the XML document from their specific elements are accessed and stored into their appropriate objects, these objects are obviously each different type of event. Each of the objects that are created is then stored using a C# dictionary in order to format and makes them easily accessible. An appropriate events class exists which then has 5 subclasses for each of the different events.
* **Task 4:** Every single event which is stored in the SOAP XML file is displayed on the canvas with suitable icons, each event has their own icon and the track log is drawn using lines to plot the points of the log.
* **Task 5:** By clicking on the canvas a sub form is shown prompting the user to choose the event type and then to add in appropriate details of that event an add button exists to allow for the entered information to be sent to a method which formats it and puts it into the SOAP XML file. If the user clicks on one of the events they are able to review the information about them a message box appears giving them the information that the event holds along with this a radius is drawn around where the centre of the event occurred. A method was created in order to differentiate between what event is being accessed in order to displays the correct information.

**Features not fully implemented:**

* **Task 5:** I wasn’t able to make it so that the information in the track logs is available by clicking on the line that I drew. I believe a work around for this would be to create an object at each end of the line which is clickable, the graphic that I drew would not let me click it and display its information though. But a line is shown on the map which relates to the track logs information.

**Features not implemented:**

I did not get to implement any of the advanced features for this particular assignment.

# Brief Description of my Solution

For this part of my documentation I will break up each task and explain them separately as to how I achieved them and what I did to get them to function as requested.

**Task 01:** This task was not difficult and only required grabbing the example SOAP XML file which was given to me in the assignment sheet and modifying it to make it contain at least 20 different events with two of them being track logs. I followed the same structure of the example as I thought it was a good layout and just added several different events with differing coordinates and information so that later when they are displayed there is a diverse range of events in different spots.

**Task 02:** This was also a very simple task simply creating a form which would hold all of my events. The form1 is created automatically when I created the whole project from there I changed the size of the form to be 1000, 1000 so that coordinates could be within those parameters. I went onto google maps and took a screenshot of a certain section of Perth and made that the background image of my form the image is also 1000, 1000 in order to keep aspect ratios and make it not look stretched. Pre-emptively I added a button which is the same size as the canvas and made it invisible this is for the later task of adding events onto the canvas when it is clicked. I believe there is probably better ways to achieve this, but I chose to do it this way and it works nicely. The button is the second element on the screen and the events are layered on top meaning you can click events and view them and then click empty spots in order to add new events to the application.

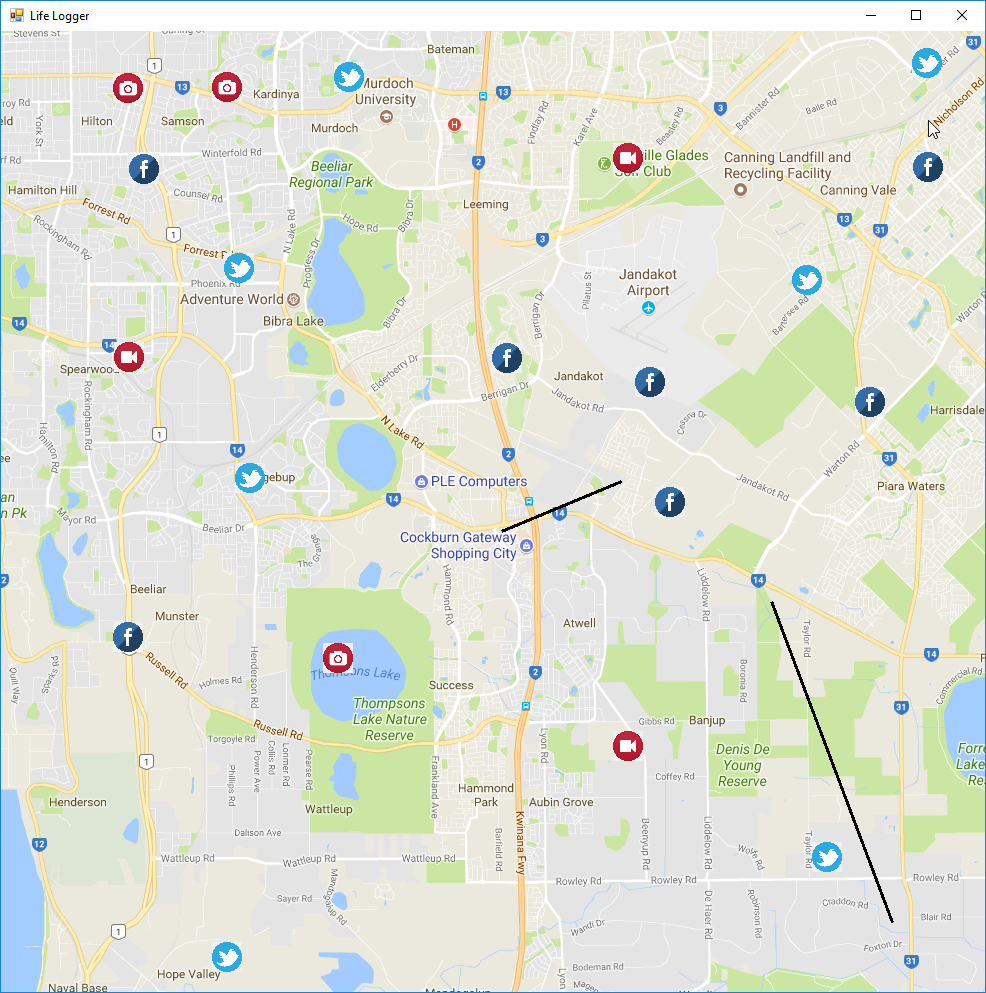
**Task 03:** Firstly I created a new Interface class in order to be my event class I chose an interface as I wanted each event to follow a certain structure, each event needs to have an event id which is specified in the interface. The event class then holds five subclasses one for each different event all inheriting the event interface. I then have a class named ReadEvent which handles reading in all of the events and storing them in an appropriate collection of objects. The ReadEvent class holds all the Dictionaries which are used to format the collection of objects and it also contains a function named ReadInXML which is called at the start of the main program class. The XML file is loaded in using XElement.load and each element of that XML document is put into an IEnumerable<XElement> which can then be used to access each different element. Firstly there is one for each loop which will loop through every element or event in the SOAP file searching for the descendants of the event element, from here there are five for each loops nested in the one for each loop one for each different event. Each event is gotten by finding the descendants of the tweet event for example and where necessary any other descendants of that event are also found and set to any necessary variables for example location has descendant elements which need to be accessed. Depending on the type of event a new event object is created for example if the event was a tweet a new tweet event is created and the contents of that event are sent to that events constructor where the object is created, that object is then put into the associated dictionary for example a tweet event goes into the tweetDict dictionary the event id gotten from the XML file is used as the key in order to access the object. Each event has its own for each loop which follows this structure obviously adapting for the differing elements and then are all put into their associated dictionary.

**Task 04:** Displaying the events on the canvas was pretty simple the way that I did it, I just have five different for each loops located in the main method of my program which do the displaying. All of the events are similar in the way they are designed except for the track log which required something different. Each of the for each loops for tweet, facebook, photo and video follow this same structure that I am going to explain, it starts off with a for each key pair in the associated events dictionary, in the for each loop a new picture box is created and each different attribute is associated with what that specific event is in the xml file. The name of the picture box is set to the id of the event which if I decided to do so could be sued to remove that event and picture from the application, the size is just simply the size of my icons which is 30, 30 pixels, the location is then set using the latitude and longitude stored in that objects lat and longi variables which is the same as in the xml file, the image for that picture box is then set to the associated events icon for example tweet event would have the tweet-icon associated with it. I then add a mouse click event to that picture so that when the icons are clicked their information will be displayed I will explain this in task 05, the mouse click event usually only takes two parameters being sender and whatever arguments but for my solution I also sent off the latitude and longitude of the icon in order to display the radius for when the click the icon which is explained in task 05. Once the picture has been setup I called Form1.Controls.add(picturevar) which then adds that picture onto the form I also call bring to front function in order to make sure the picture boxes aren’t rendered in behind anything that may be on the form. For the track logs it was much simpler to implement I simple have a for each which just looks at every event in the associated dictionary and call Form1.Paint which also sends off that objects key to the Form\_Paint method which takes the objects data being different coordinates and uses them to draw a graphics line on the screen. For my application I only chose to have it take in 4 points which is just going to make a straight line, allowing it to be longer could have been done with a loop to check how long the data section was for that event and adjust how many points there are to suit.

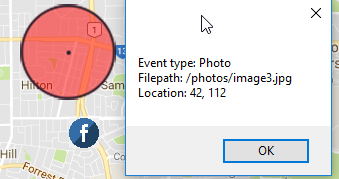
**Task 05:** On the Form1 I have a button which spans the whole size of the canvas and it is invisible so that it is not seen but can still be clicked on, this button has an associated event handler for when it is clicked this event handler grabs the x and y coordinates of the mouse which is then used for when adding a new event to get latitude and longitude, it also spawns a new form and sends mouse x and mouse y to that form for them to be used later. This new form is the form that is used in order to add new events to the application and the SOAP file. The second form contains five different radio buttons each associated with an event, there is also five panels which contain the fields for adding in information such as the text and date of a certain event. For each radio button there is an event handler which will hide every other panel and only display the one associated with the radio button for example clicking tracklog radio button will only show fields for adding stuff to a track log. The form also contains two buttons one for closing the form and another for adding the event to the soap xml file. The add button has a click listener handler on it which will determine what type of event has been chosen to be added and uses a switch case to then call the AddEvent class which contains functions for adding each type of event to the XML file, depending on the event things are sent such as the text in the texts boxes the mouse x and y coordinates are also sent to this function, once the function to add the event has been called the application is restarted in order for any changes to be shown. The form also closes after it has been submitted. The add event class contains five functions one for each type of event which gets the last event in the XML file in order to set a new ID which is just the last IDs number plus 1 making sure that every event ID added is unique, it then creates a variable which contains a bunch of XElements which contain details for the event for example a tweet event will have eventid, text, location and a datetimestamp which are the variables passed in from the add button clicked. The function also loads the XML file and adds the newly created event and then saves it. Each event follows a similar structure with the only changes being for the different elements of each event. Each picturebox contains an onclick event which creates a message box containing all of the information related to that event and also creates a new picture box which draws a radius around the event and when the message box is closed the image of the radius is removed.

# Solution Evidence

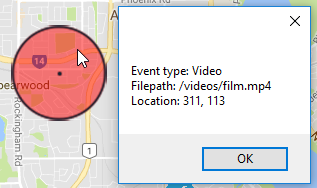
This is the main form of my program, it is a blank canvas with an image of a map of Perth as the background. All of the events and track logs are plotted onto the canvas depending on their latitude and longitude. Each of the events has their own icon and the track log is a drawn graphic line.



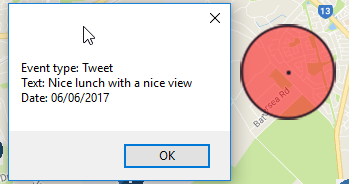
Clicking on a photo event will draw a radius around the location and then display a message box which displays any relevant details of that event. Closing the message box either through the x or ok button will make the radius disappear.



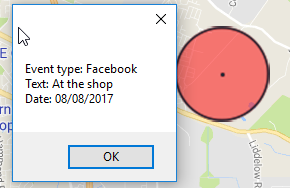
Clicking on a video event will draw a radius around the location and then display a message box which displays any relevant details of that event. Closing the message box either through the x or ok button will make the radius disappear.



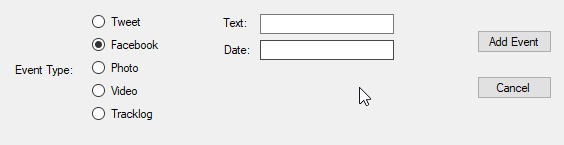
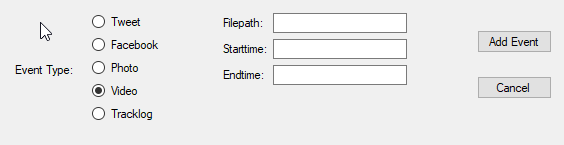
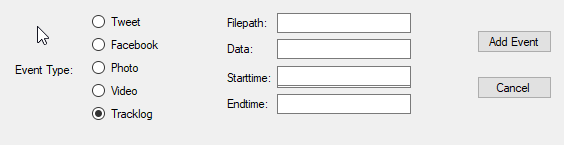
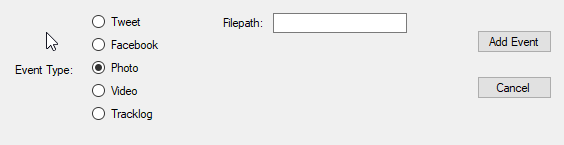
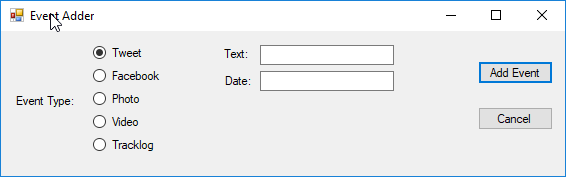
Clicking on a tweet event will draw a radius around the location and then display a message box which displays any relevant details of that event. Closing the message box either through the x or ok button will make the radius disappear.



Clicking on a Facebook event will draw a radius around the location and then display a message box which displays any relevant details of that event. Closing the message box either through the x or ok button will make the radius disappear.



Clicking anywhere on the canvas which does not already contain an event will spawn a form where the user is able to add in new events. The form defaults to a tweet event and uses radio buttons in order to change the event type. The form is several panels on top of each other which are all hidden at start-up and the radio buttons make them visible while hiding the other one.



Clicking the cancel or X button will just close the form. If the user enters in data and clicks add the event they chose to add is then added into the SOAP XML file and the application is restarted in order for the new event to be shown on the canvas. The below pictures show adding each type of event and how they show up on the canvas.

|  |
| --- |
| **Tweet Event being added** |
| C:\Users\Mitchell\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Assingment1_2017-10-01_14-09-03.png  C:\Users\Mitchell\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Assingment1_2017-10-01_14-13-58.png  C:\Users\Mitchell\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Assingment1_2017-10-01_14-14-04.png  C:\Users\Mitchell\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Assingment1_2017-10-01_14-09-23.png |
| **Facebook Event being added** |
|  |

|  |
| --- |
| **Photo Event being added** |
|  |

|  |
| --- |
| **Video Event being added** |
|  |
| **Track log Event being added** |
|  |