Enhanced Event Time-Lining for Digital Forensic Systems

Appendices

Colin Symon

Submitted in partial fulfilment of the requirements of Edinburgh Napier University for the Degree of Computer Networks & Distributed Systems (Hons)

School of Computing

November 2009
# Table of Contents

1  Project Overview, Review Form, Plan and Diaries ........................................3
   1.1  Project Overview ..................................................................................3
   1.2  Week 9 Review Form ...........................................................................5
   1.3  Project Plan ..........................................................................................7
   1.4  Diaries ..................................................................................................8
2  User Feedback Questionnaires and Feedback Email .................................24
   2.1  Completed Feedback Questionnaires ..................................................24
   2.2  Feedback Questionnaire Support Sheet ..............................................45
   2.3  Feedback Email ...................................................................................48
3  Prototype System Design and Source Code ..............................................49
   3.1  Initial Design Sketch of User Interface ..............................................49
   3.2  Source Code .........................................................................................50
       3.2.1  Windows Event Log Reader .........................................................50
       3.2.2  File Watcher ..................................................................................54
       3.2.3  Process Logger ..............................................................................58
       3.2.4  Log and Timeline Viewer ..............................................................62
1 Project Overview, Review Form, Plan and Diaries

1.1 Project Overview

Title: Enhanced Event Time-Lining for Digital Forensic System

Overview of Project Content and Milestones

Digital Event Time-lining provides a means for System Administrators and Digital Forensic Investigators to establish the sequence of events around a point of interest – such as an attack on a system. As computer systems become larger and more complex, a need has arisen for next generation Event Time-Lining systems which can gather event data and present it in innovative ways to assist in digital forensic investigations and assist the management of these complex systems. This project will cover research into current systems, ongoing advancements and develop a prototype system which enhances current methods of establishing digital event timelines.

The Main Deliverable(s)

Identify and come to conclusions about the current research into event logging/time-lining.
Design and production of a system which provides an enhanced view of digital events within time frames.
Design and perform tests. Evaluating and documenting the results from testing.
Reflect upon experiences and discuss areas for further development.

The Target Audience for the Deliverable(s)

System Administrators – such as network security engineers and those responsible for server security.
Digital forensic investigators.

The Work to be Undertaken

Research into existing time-lining and event logging tools.
Research into current problems and current research and development trends.
Investigate ways to visualise digital event timelines.
Design and implement an event logging/time-lining system.
Test and evaluate the system and its impact on the host systems performance and event evidence.
Additional Information / Knowledge Required

.NET framework – particularly Visual C#, XAML and/or similar/associated technologies. Knowledge of event logging systems and event log formats/storage.

Information Sources that Provide a Context for the Project

Microsoft documentation on the operation of Windows Event Logging features. Technical documents from similar projects.

The Importance of the Project

Current event logging systems often do not provide an efficient means for the system administrator or forensic investigator to establish links between events. Additionally, it would be beneficial for event time-lining tools to provide alternative means to view event data rather than the current method which often involves looking through thousands of lines of event records held in a text/xml format which can be difficult to quickly assimilate.

Digital crimes are likely to increase in the future and as systems become larger and more complex, the need for next generation event time-lining tools becomes very important.

The Key Challenge(s) to be Overcome

Gain a good understanding of event logging before developing a time-lining system. Acquire an understanding of new programming knowledge before developing the prototype system.
1.2 Week 9 Review Form

SOC10101 Honours Project (40 Credits)  
SOC10102 Honours Project (60 Credits) (please delete one)

Week 9 Report

Student Name: COLIN SYMON  
Matriculation Number: 05002526  
Programme (and any specialisation): BENG (HONS) COMPUTER NETWORKS AND DISTRIBUTED SYSTEMS FIT SWE  
Supervisor: Buchanan, Bill  
Second Marker: Graves, Jamie  
Date of Meeting: 4/11/09

Can the student provide evidence of attending supervision meetings by means of project diary sheets or other equivalent mechanism? [yes] [no]

If not, please comment on any reasons presented.

Please comment on the progress made so far:

Colin has made excellent progress, and has a clear idea of the problem domain & the potential solutions.

Is the progress satisfactory? [yes] [no]

Can the student articulate their aims and objectives? [yes] [no]

If yes then please comment on them, otherwise write down your suggestions:

Colin knows the baseline development objective, as well as the ideal outcome. He is also aware of the problems associated with development & has an realistic attitude.

* Please circle one answer; if no is circled then this must be amplified in the space provided.
Does the student have a plan of work?  
Yes  
No*  
If yes then please comment on that plan otherwise write down your suggestions.

It's achievable & feasible.

Does the student know how they are going to evaluate their work?  
Yes  
No*  
If yes then please comment otherwise write down your suggestions.

Colin knows & has outlined the different options. These range from multi-machine performance evaluation to scenario testing. All of which are good ways of evaluating this work.

Any other recommendations as to the future direction of the project:

Look at log management and analysis software, & investigate performance enhancements (especially for the database.

Signatures:  
Supervisor  
Second Marker  
Student

Please give the student a photocopy of this form immediately after the review meeting; the original should be lodged in the School Office with Leanne Clyde.

* Please circle one answer; if no is circled then this must be amplified in the space provided.
### 1.3 Project Plan

<table>
<thead>
<tr>
<th>Tasks</th>
<th>Start Date</th>
<th>Duration (days)</th>
<th>End Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>* Initial Project Overview *</td>
<td>02/02/2009</td>
<td>17</td>
<td>19/02/2009</td>
</tr>
<tr>
<td>* Week 4 Meeting Preparation *</td>
<td>16/02/2009</td>
<td>7</td>
<td>23/02/2009</td>
</tr>
<tr>
<td>Literature Review and Experimentation</td>
<td>24/02/2009</td>
<td>38</td>
<td>03/04/2009</td>
</tr>
<tr>
<td>* Week 9 Meeting Preparation *</td>
<td>31/03/2009</td>
<td>7</td>
<td>07/04/2009</td>
</tr>
<tr>
<td>Introduction</td>
<td>24/03/2009</td>
<td>10</td>
<td>03/04/2009</td>
</tr>
<tr>
<td>Design</td>
<td>08/04/2009</td>
<td>21</td>
<td>29/04/2009</td>
</tr>
<tr>
<td>Implementation</td>
<td>30/04/2009</td>
<td>151</td>
<td>28/09/2009</td>
</tr>
<tr>
<td>Conclusions</td>
<td>26/10/2009</td>
<td>21</td>
<td>16/11/2009</td>
</tr>
<tr>
<td>Checking References and Appendices</td>
<td>09/11/2009</td>
<td>7</td>
<td>16/11/2009</td>
</tr>
<tr>
<td>Submit to Turnitin and Tweaking</td>
<td>15/11/2009</td>
<td>14</td>
<td>29/11/2009</td>
</tr>
</tbody>
</table>

Note: *x* Indicates fixed end date
1.4 Diaries

NAPIER UNIVERSITY
SCHOOL OF COMPUTING
PROJECT DIARY

Student: COLIN SYMON
Date: 26th February 2009

Objectives:
1. Investigate Windows log files and reading from them.
2. Find papers/articles on event logging/time-stamping.
3. Look into identifying memory processes.

Progress:
1. Produced demo app to read from Windows Event Log files.
2. Found some references about logging - need to try and find references more specific to displaying as time logs.
3. Produced demo console app to log process start and stop, will try out different versions.

Supervisor’s Comments:
Good progress. Nice to see some demo of reading from the event log and on picking process information. Keep reviewing the literature, especially for logging and displaying and in displaying timelines. Think also about different viewpoints, such as users, network, process and file.
NAPIER UNIVERSITY
SCHOOL OF COMPUTING
PROJECT DIARY

Student: COLIN SYMON
Date: 12th March 2009

Objectives:
1. Find references related to retrieve/display guest information.
2. Look into gathering data for viewpoints such as user, network, process and file.

Supervisor: BILL BUCHANAN
Last diary date: 26th February 2009

Progress:
1. Found out how to gather process starts/stop and associated username.
2. Found suitable articles as potential sources for literature review.
   Made a start to literature review.
3. Formalised project plan.

Supervisor's Comments:

Very good progress! Think about your twelve, and especially the evaluate phase, and think about your experiments at this stage.

Version 2

Napier University
NAPIER UNIVERSITY
SCHOOL OF COMPUTING
PROJECT DIARY

Student: COLIN SYMON
Date: 4th March 2009

Objectives:
- Consider project plan timeline.
- Consider testing to be carried out on time-units system.

Progress:
- Re-evaluated timeline, adjusted periods for evaluation and write-ups.
- Thought about testing - more to do in this area.
- Found article on visualization which will be useful for literature review and for development ideas.

Supervisor’s Comments:
Good start with the literature review. Make sure you write down all the ideas behind the pages you are looking at. Think about your week 9 document.

Version 2
Napier University
PROJECT DIARY

Student: COLIN SYMON
Date: 23rd March 2009

Objectives:
- Browse papers.
- Continue with lit review.
- Consider document for writing review.

Progress:
- Continue with lit review.
- Have papers organised.
- Found a method of visualising data - requires further development to decide suitability.

Supervisor's Comments:
- Explore Virginia's interests and fit in with .NET 3.0. Make a stand on the Week 9 document. It should contain:
  - Introduction (Context, Background), Lit Review (some literature papers), With due to date, 
  - and a Grant Chart for your timing
- Think about evaluation techniques at this stage.
NAPIER UNIVERSITY
SCHOOL OF COMPUTING
PROJECT DIARY

Student: COLIN SYMON
Date: 1st April 2009

Objectives:

- Begin work on week 9 document.
- Continued lit review.
- Begin introduction.
- Revisited project plan chart.

Progress:

- Continued with lit review.
- Made a start on the introduction.
- Revisited Gantt chart for project plan.
- Experimented with using a SQL database for storing log entries.

Supervisor's Comments:

Investigate existing products such as MapInfo and eCase. Finish your week 9 document, and send a draft to me, and we can finalise it. Good progress on investigating the database side. Investigate the system on a tagged approach: data storage, data acquisition, filtering/processing, and data interface.

Version 2
NAPIER UNIVERSITY
SCHOOL OF COMPUTING
PROJECT DIARY

Student: COLIN SYMON
Date: 29th April 2009

Objectives:

Think about layer approach - storage, gathering and user interface.

Outline similar products.

Progress:

Looked at similar products for inspiration. Ingenium is a similar product which provides optional reporting features.

Began work on storage (database) design.

Supervisor’s Comments:

Good to see a start on the database design. Have a think about how you would interface with other database systems.

Version 2

Napier University
NAPIER UNIVERSITY
SCHOOL OF COMPUTING
PROJECT DIARY

Student: COLIN SYMON
Date: 5th May 2009

Objectives:
Consider evaluation of database systems.
Work on file, process, user view of system and data
logging.

Progress:
Experimented with file system watcher class to monitor
changes to files. Updated class design to restrict this.
More work to do on evaluation.

Supervisor’s Comments:
Good to see the angle of the Watcher class. This might be useful for evaluat-
ing expert in creating a robot application
with diving activity (macro-mode help). Thnk carefully about timing activity.
Also think about filling the date into a DateTable.

Version 2
NAPIER UNIVERSITY
SCHOOL OF COMPUTING
PROJECT DIARY

Student: COLIN SYMON
Date: 13th May 2009

Objectives:

Consider how to use flexmatch for merging activities.
Investigate who would be available to fill data grid.
Work on design.

Progress:

Created demo of database using methods discussed.
Began work on initial design of user interface.
Still work to do on monitoring activities. Refers to
classes logged already. Continues to develop more of topics.
Validating this.

Supervisor’s Comments:

Good progress on integrity with MS SQL.
Have a think about using another type of database so that you can evaluate different performance issues, such as retrieval
speed of access and overall database size.

Version 2
Napier University
NAPIER UNIVERSITY
SCHOOL OF COMPUTING
PROJECT DIARY

Student: COLIN SYMON
Date: 8th September 2009

Supervisor: BILL BUCHANAN
Last diary date: 28th May 2009

Objectives:

Work on implementation as soon as design.
Continue to develop report to document progress.
Further develop testing ideas.

Progress:

Made significant progress on implementation as
design. Have file and process working programs
working and have guest viewer working with
utilisation (bar graph) using 2D graph.
Have worked on report, added further review and
design sections.
Continued with testing ideas, more to develop on this.

Supervisor's Comments:

Good progress. Think about the evaluation
parameters such as speed of update, CPU utilisation, memory utilisation and
user perception of graphical presentation.

Version 2
Napier University
NAPIER UNIVERSITY
SCHOOL OF COMPUTING
PROJECT DIARY

Student: COLIN SYMON
Date: 13th September 2009
Objectives:

consider testing and evaluation of the system with view to gathering an impact on system and user perception.

Progress:

made further enhancements to user interface by improving graphical features, planned testing options and ways to compare to existing sections.

Supervisor's Comments:

Good progress. Think deeply about the evaluation path, especially for user and technical aspects. Try to set some early evaluation to see where the main issues are likely to be.

Version 2
Napier University
NAPIER UNIVERSITY
SCHOOL OF COMPUTING
PROJECT DIARY

Student: COLIN SYMON
Date: 27/9/2009

Objectives:
- Plan how to evaluate the system from a user and technical perspective.
- Consider adding in more search options to aid in building time lines as events.

Supervisor: BILL BUCHANAN
Last diary date: 15/9/2009

Progress:
- Added more search options for users of the system.
- Thought about evaluation, considering use of questionnaires to ask testers for feedback.
- Have some plans for testing system performance - more to do on this.

Supervisor’s Comments:
- Good progress. Think about a single question for users to complete, based on certain hypotheses. Define a basic method for each experiment with a aim ad an experimental setup.
NAPIER UNIVERSITY
SCHOOL OF COMPUTING
PROJECT DIARY

Student: COLIN SYMON
Date: 29th September 2009
Objectives:

Continue with designing tests with aims, objectives and methods.
Develop questionnaire for needing user feedback.
Investigate how to measure technical aspects.

Progress:

Continued to develop testing plan.
Worked on questionnaire.
Investigated use of Windows Performance Monitor
and was 95% measuring execution time for technical testing.
Agreed to meet with an external IT professional who specializes in security to get feedback for evaluation.

Supervisor’s Comments:

Good progress on the evaluation aspect from a user point-of-view and from a performance point-of-view. Make sure you think of experiments which have a heavy performance but so that you can examine stress points.

Version 2
Napier University
NAPIER UNIVERSITY
SCHOOL OF COMPUTING

PROJECT DIARY

Student: COLIN SYMON
Supervisor: BILL BUCHANAN

Date: 6th October 2009
Last diary date: 24th September 2009

Objectives:

- Work on methods of testing especially those which
given the system.
- Work on user feedback questionnaire.

Progress:

- Personal initial testing.
- Have set date to meet with Suja at Distillation
  Centre for feedback and remarks.
- Worked on questionnaires.
- Proposed scenarios for user testing and feedback.

Supervisor’s Comments:

Good progress on the evaluation sheet.
Try to focus on comparative analysis
on Eval 1, such as trying screen
shorts at activity, and ask the users
to rate them for their usefulness.
(1 - 5)

Version 2
Napier University
PROJECT DIARY

Student: COLIN SYMON
Date: 28th October 2009

Objectives:

Improve user questionnaire so as to allow comparisons to be made.
Work on report.

Progress:

Developed feedback questionnaire with questions designed to allow for conclusions to be made.
Worked on report implementation and testing sections, more to do on this.

Supervisor's Comments:

Great to see some evaluation on a rating system for the evaluators.
NAPIER UNIVERSITY
SCHOOL OF COMPUTING
PROJECT DIARY

Student: COLIN SYMON
Date: 20th October 2009

Objectives:

Carry out user feedback evaluation.
Meet with Sonya Buxton (Security Officer at Bell)
Continue work on report.

Progress:

Met Sonya and received feedback regarding the prototype system and how such a system would fit into her organisation.
Worked on report - written up implementation of the prototype.
Carried out use @declue - more to do on this e.g.
technical testing.

Supervisor’s Comments:

Good to see some feedback from the evaluator. Make sure you start on your report quite soon. Try to schedule 2-3 days before delivery for a near complete draft.

Version 2  Napier University
NAPIER UNIVERSITY
SCHOOL OF COMPUTING
PROJECT DIARY

Student: COLIN SYMON
Date: 13th November 2009

Objectives:

Continue to work on report.
Carry out further user feedback.

Supervisor: BILL BUCHANAN
Last diary date: 20th October 2009

Progress:

Worked on report - finished writing up technical implementation.
Testing...

Carried out more user feedback - need to complete this
soon for evaluation. At stage of writing up results.

Supervisor's Comments:

The report has a good flow, but there needs to be more abstract
of key concepts, and overall design.
Include more code snippets, e.g.,
for XAML and graphics components...

Version 2

Napier University
2 User Feedback Questionnaires and Feedback Email

2.1 Completed Feedback Questionnaires

User Feedback Questionnaire

Name: Sonya Ruiz

Thank you for taking part in this evaluation and feedback session.
Part 1 asks about your experience with the prototype system.
Part 2 contains more general questions and asks you to indicate your preferences regarding ways of viewing event data.
## Part 1 – Overall impression of the prototype

<table>
<thead>
<tr>
<th></th>
<th>In this section please indicate how difficult you found performing the tasks</th>
<th>easy</th>
<th>neither easy nor hard</th>
<th>hard</th>
<th>Comments:</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>How easy did you find it to familiarise yourself with the application and its features?</td>
<td>✔</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>How easy did you find it to create xml log files using the console applications?</td>
<td>✔</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>In this section indicate how useful you found the features</th>
<th>useless</th>
<th>neither useless nor useful</th>
<th>useful</th>
<th>Comments:</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>The dashboard.</td>
<td></td>
<td></td>
<td>✔</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>The advanced query option.</td>
<td></td>
<td></td>
<td>✔</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>For question 5 indicate how long you would wait.</th>
<th>Less than 5 minutes</th>
<th>5 to 10 minutes</th>
<th>More than 10 minutes</th>
<th>Comments:</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>How long would you be prepared to wait while the application loaded a large amount of log information into the database?</td>
<td>✔</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**Part 2 – Preferences**

<table>
<thead>
<tr>
<th></th>
<th>Text File</th>
<th>Event Viewer</th>
<th>Prototype</th>
<th>Comments:</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>7</td>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td>2</td>
<td>3</td>
</tr>
</tbody>
</table>

**Use section 2 of the support sheet to answer these questions.**

**Which chart do you prefer for spotting anomalies?**

<table>
<thead>
<tr>
<th></th>
<th>Pie Chart</th>
<th>Bar Chart</th>
<th>Line Chart</th>
<th>Comments:</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Use section 3 of the support sheet to answer this question.**

**Charts displaying user login data over a 24hr period.**

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Would be useful to have a way to export to excel for example and some further reporting.**

---

Thank you for your participation.
User Feedback Questionnaire

Name: Dario Ackee

Position: NURSING STUDENT

Thank you for taking part in this evaluation and feedback session.
Part 1 asks about your experience with the prototype system.
Part 2 contains more general questions and asks you to indicate your preferences regarding ways of viewing event data.
### Part 1 – Overall impression of the prototype

<table>
<thead>
<tr>
<th></th>
<th>In this section please indicate how difficult you found performing the tasks</th>
<th></th>
<th></th>
<th></th>
<th>Comments:</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>How easy did you find it to familiarise yourself with the application and its features?</td>
<td>easy</td>
<td>neither easy nor hard</td>
<td>hard</td>
<td>It was easy to understand the system.</td>
</tr>
<tr>
<td>2</td>
<td>How easy did you find it to create and log files using the console applications?</td>
<td></td>
<td></td>
<td></td>
<td>But more explanation needed.</td>
</tr>
<tr>
<td></td>
<td>In this section indicate how useful you found the features</td>
<td>useless</td>
<td>neither useless nor useful</td>
<td>useful</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>The dashboard.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>The advanced query option.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>For question 5 indicate how long you would wait.</td>
<td>Less than 5 minutes</td>
<td>5 to 10 minutes</td>
<td>More than 10 minutes</td>
<td>Comments:</td>
</tr>
<tr>
<td></td>
<td>How long would you be prepared to wait while the application processed a large amount of log information into the database?</td>
<td></td>
<td></td>
<td></td>
<td>It would be useful for someone with knowledge of SQL queries.</td>
</tr>
</tbody>
</table>
### Part 2 – Preferences

<table>
<thead>
<tr>
<th></th>
<th>Use section 1 of the support sheet to answer these questions. Rank in order of preference (1 being most preferred).</th>
<th>Text file</th>
<th>Event Viewer</th>
<th>Prototype</th>
<th>Comments:</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>Indicate your preference regarding getting an overview of event data.</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Indicate your preference regarding filtering log data.</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Indicate your preference for viewing detailed log data.</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>

Use section 2 of the support sheet to this question.

<table>
<thead>
<tr>
<th></th>
<th>With colour</th>
<th>Without colour</th>
<th>Comments:</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Use section 3 of the support sheet to answer this question. Rank in order of preference (1 being most preferred).

<table>
<thead>
<tr>
<th></th>
<th>Pie Chart</th>
<th>Bar Chart</th>
<th>Line Chart</th>
<th>Comments:</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Charts displaying user login data over a 24hr period.

<table>
<thead>
<tr>
<th></th>
<th>Comments:</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td></td>
</tr>
</tbody>
</table>

Do you have any other comments or suggestions?

*File names for log to include machine name*

Thank you for your participation.
User Feedback Questionnaire

Name: Owen Lo
Position: Computing Student

Thank you for taking part in this evaluation and feedback session.

Part 1 asks about your experience with the prototype system.

Part 2 contains more general questions and asks you to indicate your preferences regarding ways of viewing event data.
### Part 1 – Overall impression of the prototype

<table>
<thead>
<tr>
<th></th>
<th>In this section please indicate how difficult you found performing the task</th>
<th>easy</th>
<th>neither easy nor hard</th>
<th>hard</th>
<th>Comments:</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>How easy did you find it to familiarise yourself with the application and its features?</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>How easy did you find it to create and log files using the console applications?</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>In this section indicate how useful you found the features</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>The dashboard.</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>The advanced query option.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>For question 5 indicate how long you would wait.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

For question 5 indicate how long you would wait:

- Less than 5 minutes
- 5 to 10 minutes
- More than 10 minutes

Comments: 

- Would twice time to learn SQL
### Part 2 - Preferences

<table>
<thead>
<tr>
<th>Question</th>
<th>Text file</th>
<th>Event Viewer</th>
<th>Prototype</th>
<th>Comments:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use section 1 of the support sheet to answer these questions. Rank in order of preference (3 being most preferred).</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6 Indicate your preference regarding getting an overview of event data.</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>7 Indicate your preference regarding filtering log data.</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>8 Indicate your preference for viewing detailed log data.</td>
<td></td>
<td></td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>Use section 2 of the support sheet to this question.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>With colour</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Without colour</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Comments:</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9 Which chart do you prefer for spotting anomalies?</td>
<td></td>
<td></td>
<td></td>
<td>yes</td>
</tr>
<tr>
<td>Use section 3 of the support sheet to answer this question. Rank in order of preference (3 being most preferred). Charis displaying user login data over a 24hr period.</td>
<td></td>
<td>Pie Chart</td>
<td>Bar Chart</td>
<td>Line Chart</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td></td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>Comments:</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11 Do you have any other comments or suggestions?</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Thank you for your participation.
User Feedback Questionnaire

Thank you for taking part in this evaluation and feedback session.

Part 1 asks about your experience with the prototype system.

Part 2 contains more general questions and asks you to indicate your preferences regarding ways of viewing event data.

Name:
Position:

[Signatures]

Colin Symon
05002526
November 2009
### Part 1 – Overall impression of the prototype

<table>
<thead>
<tr>
<th></th>
<th>In this section please indicate how difficult you found performing the tasks.</th>
<th>easy</th>
<th>neither easy nor hard</th>
<th>hard</th>
<th>Comments:</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>How easy did you find it to familiarise yourself with the application and its features?</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>How easy did you find it to create xml log files using the console applications?</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>In this section indicate how useful you found the features.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>The dashboard.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>The advanced query option.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>For question 5 indicate how long you would wait.</td>
<td>Less than 5 minutes</td>
<td>5 to 10 minutes</td>
<td>More than 10 minutes</td>
<td></td>
</tr>
</tbody>
</table>
## Part 2 – Preferences

<table>
<thead>
<tr>
<th>Use section 1 of the support sheet to answer these questions. Rank in order of preference (3 being most preferred).</th>
<th>Text file</th>
<th>Event Viewer</th>
<th>Prototype</th>
<th>Comments:</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>Indicate your preference regarding getting an overview of event data.</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>7</td>
<td>Indicate your preference regarding filtering log data.</td>
<td>1</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>8</td>
<td>Indicate your preference for viewing detailed log data.</td>
<td>With colour</td>
<td>Without colour</td>
<td>2</td>
</tr>
<tr>
<td>Use section 2 of the support sheet to this question.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Which chart do you prefer for spotting anomalies?</td>
<td>Pie Chart</td>
<td>Bar Chart</td>
<td>Line Chart</td>
</tr>
<tr>
<td>Use section 3 of the support sheet to answer this question. Rank in order of preference (3 being most preferred). Charts displaying user login data over a 24hr period.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>Do you have any other comments or suggestions?</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The application flows quite well. I would say a future version should make log importing smoother & make the query a bit English-like.

Thank you for your participation.
User Feedback Questionnaire

Name: RICH MACFARLANE  Position: Pt Lecturer

Thank you for taking part in this evaluation and feedback session.
Part 1 asks about your experience with the prototype system.
Part 2 contains more general questions and asks you to indicate your preferences regarding ways of viewing event data.
### Part 1 – Overall impression of the prototype

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>easy</th>
<th>neither easy nor hard</th>
<th>hard</th>
<th>Comments:</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>How easy did you find it to familiarise yourself with the application and its features?</td>
<td>✔️</td>
<td></td>
<td></td>
<td>Load Logs Tab could be tidied up with drop down exclusive choices. (LTV)</td>
</tr>
<tr>
<td>2</td>
<td>How easy did you find it to create zrtf log tiles using the console applications?</td>
<td>✔️</td>
<td></td>
<td></td>
<td>Could add asynchronous console apps from single console to create ZT at a time.</td>
</tr>
<tr>
<td>3</td>
<td>In this section indicate how useful you found the features</td>
<td>useless</td>
<td>neither useful nor useful</td>
<td></td>
<td>Would you want the data to be going over the web to google. Secure COM’s?? Use db procedures or better validation to prevent SQL Attack.</td>
</tr>
<tr>
<td>4</td>
<td>The dashboard.</td>
<td>✔️</td>
<td></td>
<td></td>
<td>❌ could have dynamic sheet sheet showing columns etc.</td>
</tr>
<tr>
<td>5</td>
<td>The advanced query option.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Part 2 - Preferences

<table>
<thead>
<tr>
<th>Use section 1 of the support sheet to answer these questions. Rank in order of preference (3 being most preferred).</th>
<th>Text</th>
<th>Event Viewer</th>
<th>Prototype</th>
<th>Comments:</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>What is the support sheet?</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Indicate your preference regarding getting an overview of event data.</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Indicate your preference regarding filtering log data.</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>Using vi editor.</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Indicate your preference for viewing detailed log data. Use section 2 of the support sheet to answer this question.</th>
<th>With colour</th>
<th>Without colour</th>
<th>Comments:</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Which chart do you prefer for spotting anomalies? Use section 3 of the support sheet to answer this question. Rank in order of preference (3 being most preferred).</th>
<th>Pie Chart</th>
<th>Bar Chart</th>
<th>Line Chart</th>
<th>Comments:</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td></td>
<td></td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>

| Charts displaying user login data over a 24hr period. Comments: |
|---|---|
| 11 | 3 | 2 | |

<table>
<thead>
<tr>
<th>Do you have any other comments or suggestions?</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

Thank you for your participation.
User Feedback Questionnaire

Name: Mark McGowan

Position: Computing Student

Thank you for taking part in this evaluation and feedback session.
Part 1 asks about your experience with the prototype system.
Part 2 contains more general questions and asks you to indicate your preferences regarding ways of viewing event data.
### Part 1 – Overall impression of the prototype

<table>
<thead>
<tr>
<th></th>
<th>easy</th>
<th>neither easy nor hard</th>
<th>hard</th>
<th>Comments:</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td>Feedback once log is loaded to say where to go/what to do would be helpful. Very simple, no issues</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td>To show data has been loaded and a quick reference for what has been monitored. Although not essential to use, allows user to dwell down for particular event/machine. Needs knowledge of SAQ questions to work well.</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td>Obviously went into past, but due to nature of this application I would want the data and would be willing to wait 30+min. If it retrieved the current data and was easy to use.</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**Part 2 – Preferences**

<table>
<thead>
<tr>
<th>Use section 1 of the support sheet to answer these questions. Rank in order of preference (3 being most preferred).</th>
<th>Text file</th>
<th>Event Viewer</th>
<th>Prototype</th>
<th>Comments:</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>Indicate your preference regarding getting an overview of event data.</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>7</td>
<td>Indicate your preference regarding filtering log data.</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>8</td>
<td>Indicate your preference for viewing detailed log data. Use section 2 of the support sheet to this question.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>With colour</td>
<td>Without colour</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Which chart do you prefer for spotting anomalies? Use section 3 of the support sheet to answer this question. Rank in order of preference (3 being most preferred).</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>Charts displaying user login data over a 24hr period.</td>
<td>1</td>
<td>3</td>
<td>2</td>
</tr>
</tbody>
</table>

**Comments:**

11 Do you have any other comments or suggestions?

Thank you for your participation.
User Feedback Questionnaire

Name: Leigh Soothers  Position: Student

Thank you for taking part in this evaluation and feedback session.
Part 1 asks about your experience with the prototype system.
Part 2 contains more general questions and asks you to indicate your preferences regarding ways of viewing event data.
## Part 1 – Overall impression of the prototype

<table>
<thead>
<tr>
<th></th>
<th>In this section please indicate how difficult you found performing the task</th>
<th>easy</th>
<th>neither easy nor hard</th>
<th>hard</th>
<th>Comments:</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>How easy did you find it to familiarise yourself with the application and its features?</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>How easy did you find it to create xml log files using the console applications?</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

In this section indicate how useful you found the features

<table>
<thead>
<tr>
<th></th>
<th>useless</th>
<th>neither useless nor useful</th>
<th>useful</th>
<th>Comments:</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>The dashboard.</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>The advanced query option.</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

For question 5 indicate how long you would wait.

<table>
<thead>
<tr>
<th></th>
<th>Less than 5 minutes</th>
<th>5 to 10 minutes</th>
<th>More than 10 minutes</th>
<th>Comments:</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>How long would you be prepared to wait while the application loaded a large amount of log information into the database?</td>
<td>✓</td>
<td></td>
<td>Possibly a progress bar to indicate the current status of the progress.</td>
</tr>
</tbody>
</table>
Part 2 – Preferences

<table>
<thead>
<tr>
<th></th>
<th>Use section 1 of the support sheet to answer these questions. Rank in order of preference (3 being most preferred).</th>
<th>Text file</th>
<th>Event Viewer</th>
<th>Prototype</th>
<th>Comments:</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>Indicate your preference regarding getting an overview of event data.</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Indicate your preference regarding filtering log data.</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Indicate your preference for viewing detailed log data.</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Use section 2 of the support sheet to this question.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Which chart do you prefer for spotting anomalies?</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

|   | Use section 3 of the support sheet to answer this question. Rank in order of preference (3 being most preferred). Charts displaying user login data over a 24hr period. | Pie Chart | Bar Chart   | Line Chart | Comments: |
|10 |                                                                                                                   |          |             |            |           |
|   |                                                                                                                   | 1        | 3           | 2         |           |

|11 | Do you have any other comments or suggestions?                                                                      |          |             |            |           |

Thank you for your participation.
2.2 Feedback Questionnaire Support Sheet
2.3 Feedback Email

From: Buczyn, Sonya [sbuczyn@eastlothian.gov.uk]
Sent: 14 October 2009 11:23
To: Symon, Colin
Subject: RE: Napier Honours Project Feedback

Hi Colin,

Thanks for coming in and showing me your application. As I said when you were here I think it looks like a very useful tool and very easy to use. The way the information is presented is very clear and the charts are an added bonus. As you know we use encase here for any internal investigations which can be pretty complex and take a long time to search for evidence. I can certainly see a place for using your application alongside encase as this would give a really quick method for searching for specific events. Your app would be a great addition to my investigation ‘toolkit’

Also all UK councils have a connection to a central government network which provides access to some central services. To be part of this network we have to meet a strict Code of Connection and I know the next version of this code expects us all to have in place a central method of logging events and being able to interrogate these events if need be. So logging is a ‘hot’ area just now within government circles. I’ve seen a couple of logging applications recently and based on what I saw of your app today, yours compares favourably with the commercial ones for the actual core logging and searching process.

If you need anything else or are ever looking for someone to test your app in a live situation then let me know. Good luck for the future.

Regards
Sonya

Sonya Buczyn
IT Security Officer
IT Department, East Lothian Council
3 Prototype System Design and Source Code

3.1 Initial Design Sketch of User Interface
3.2 Source Code

3.2.1 Windows Event Log Reader

Program.cs

```csharp
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Diagnostics;
using System.Xml;
using System.Security;

namespace Project_EventReader
{
    class Program
    {
        static void Main(string[] args)
        {
            // Main method presents a menu of options to the user and actions the response
            string menuChoice = "";
            Console.WriteLine("--------------------------------------------
\nEvent Reader\n--------------------------------------------");
            menuChoice = chooseMenu();
            if (menuChoice == "Q" || menuChoice == "q")
            {
                Environment.Exit(Environment.ExitCode);
            }
            if (menuChoice == "1")
            {
                processLog("Application");
            }
            if (menuChoice == "2")
            {
                processLog("System");
            }
            if (menuChoice == "3")
            {
                processLog("Security");
            }
            else
            {
                Console.WriteLine("nInvalid option entered. Please try again.
\n")
                menuChoice = chooseMenu();
            }
        }

        private static string chooseMenu()
        {
            // lists menu options
            string menuChoice = "";
            Console.Write("Key 1 for Application Log\nKey 2 for System Log\nKey 3 for Security Log\nKey Q to Quit\n\nOption: ");
            menuChoice = Console.ReadLine();
            return menuChoice;
        }
    }
}
```
public static string encodeString(string original) {
    //Based on method by Rick van den Bosch. Found at http://bloggingabout.net/blogs/rick/archive/2005/05/18/4118.aspx
    Byte[] originalBytes;
    Byte[] encodedBytes;

    //Generate hash
    SHA256 sha = new SHA256CryptoServiceProvider();
    originalBytes = ASCIIEncoding.Default.GetBytes(original);
    encodedBytes = sha.ComputeHash(originalBytes);

    //Convert encoded bytes back to a string
    return BitConverter.ToString(encodedBytes);
}

private static void processLog(string log_type) {
    string event_data = "";
    string user = "";
    DateTime timestamp = DateTime.Now.ToUniversalTime();
    string formatted_timestamp = timestamp.ToString("yyyy-MM-dd HH:mm:ss");
    DateTime entry_time;
    string filename = log_type + "Log_" + formatted_timestamp.Replace(":", "-") + ".xml"; //have to remove : character as it is illegal in a filename
    string entry_formatted_time = "";
    int percentage = 0;
    EventLog aLog = new EventLog(log_type);
    bool last = false;
    DateTime was = DateTime.Now;
    int total = aLog.Entries.Count;
    int count = 0;

    Console.WriteLine("\nLoading in Log (this may take a minute...)");

    try {
        XmlDocument xmlDoc = new XmlDocument();
        try {
            xmlDoc.Load(filename);
        }
        catch (System.IO.FileNotFoundException)
        {
            //make file if it cannot be found
            XmlTextWriter xmlWriter = new XmlTextWriter(filename, System.Text.Encoding.UTF8);
            xmlWriter.Formatting = Formatting.Indented;
            xmlWriter.WriteProcessingInstruction("xml", "version='1.0' encoding='UTF-8'");
            xmlWriter.WriteStartElement("eventlog");
            xmlWriter.Close();
            xmlDoc.Load(filename);
        }

        foreach (EventLogEntry entry in aLog.Entries)
event_data = entry.Message;

//remove line breaks
event_data = event_data.Replace("\r\n", "");

if (entry.UserName == null)
{
    user = "Not Recorded";
}
else
{
    user = entry.UserName;
}

entry_time = entry.TimeGenerated.ToUniversalTime();

entry_formatted_time = entry_time.Year + "-" + entry_time.Month + "-" + entry_time.Day + " " + entry_time.TimeOfDay;

string checksum = entry_formatted_time +
entry.MachineName + user + entry.EventID.ToString() + event_data;

count++;

if (count == total)
{
    last = true;
}

XmlNode root = xmlDoc.DocumentElement;
XmlElement childNode =
xmlDoc.CreateElement("recorded_event");
XmlElement childNode2 =
xmlDoc.CreateElement("event_time");
XmlElement childNode3 =
xmlDoc.CreateElement("machine_name");
XmlElement childNode4 =
xmlDoc.CreateElement("user_account");
XmlElement childNode5 =
xmlDoc.CreateElement("event_id");
XmlElement childNode6 =
xmlDoc.CreateElement("event_data");
XmlElement childNode7 =
xmlDoc.CreateElement("checksum");

XmlText textNode2 =
xmlDoc.CreateTextNode(entry_formatted_time);

XmlNode childNode3 =
xmlDoc.CreateTextNode(entry.MachineName);

XmlNode textNode4 =
xmlDoc.CreateTextNode(user);

XmlNode textNode5 =
xmlDoc.CreateTextNode(entry.EventID.ToString());

XmlNode textNode6 =
xmlDoc.CreateTextNode(event_data);

XmlNode textNode7 =
xmlDoc.CreateTextNode(checksum);

root.AppendChild(childNode);

childNode.AppendChild(childNode2);

childNode2.AppendChild(textNode2);

childNode.AppendChild(childNode3);

childNode3.AppendChild(textNode3);

childNode.AppendChild(childNode4);
childNodes4.AppendChild(textNode4);
childNode.AppendChild(childNode5);
childNode5.AppendChild(textNode5);
childNode.AppendChild(childNode6);
childNode6.AppendChild(textNode6);
childNode.AppendChild(childNode7);
childNode7.AppendChild(textNode7);
if (last == true)
{
    // save the file if all records have been added
    xmlDoc.Save(filename);
}
if (was.AddSeconds(1.00) < DateTime.Now || count == 1)
// update on first loop then update once a second, stops output flickering
{
    // if 1 second has passed inform the user on progress
    percentage = (count * 100) / total;
    Console.Write(percentage.ToString() + "%");
    was = DateTime.Now;
}
Console.WriteLine("\nFinished writing log. Filename is: " + filename + "\n\nPress return key to exit.");
Console.ReadLine();
Environment.Exit(Environment.ExitCode);
}
catch (Exception)
// write out exception message to console if writing xml fails
{
    Console.WriteLine("Error writing to log.\nCheck folder permissions to ensure program can write files and try again. \n\nPress any key to exit.");
    Console.ReadLine();
    Environment.Exit(Environment.ExitCode);
}
}
3.2.2 File Watcher

Program.cs

```csharp
using System;
using System.Collections.Generic;
using System.Text;
using System.IO;

namespace Project_FileWatcher
{
    class Program
    {
        static void Main(string[] args)
        {
            //set the directory to watch
            string username = System.Environment.GetEnvironmentVariable("USERNAME"); //ensures file
            //watcher will watch current users files rather than hard coded
            //the following path can be used to monitor temporary internet
            //files created by Windows Internet Explorer
            //string path = "C:\\Users\\"+username+"\\AppData\\Local\\Microsoft\\Windows\\Temporary
            //Internet Files";
            string path = "C:\\testfiles";
            //create new filewatcher and pass it the path to monitor
            FileWatcher doIt = new FileWatcher(path);
            //inform user that the program is running and the path being
            monitored
            Console.WriteLine("File Watcher is Running");
            Console.WriteLine("Path being watched: " + path);
            Console.ReadLine();
        }
    }
}
```

FileWatcher.cs

```csharp
using System;
using System.Collections.Generic;
using System.Text;
using System.Xml;
using System.IO;
using System.Security;
using System.Threading;

namespace Project_FileWatcher
{
    class FileWatcher
    {
        //timestamp to use when naming log file
        static string formatted_timestamp = DateTime.Now.ToUniversalTime().ToString("yyyy-MM-dd HH:mm:ss"); //universal
        //time

        //location of log file is set in the following line
        static string filename = "logs\\FileLog_" +
        formatted_timestamp.Replace( ":", "-" ) + ".xml"; //have to remove :
        //character as it is illegal in a filename
    }
}
```
public FileWatcher(string path)
{
    //configuration of the FileSystemWatcher
    //set to monitor file creation, rename, deletion and change events
    string directory = path;
    FileSystemWatcher WatchFile = new FileSystemWatcher(directory);
    WatchFile.Created += new FileSystemEventHandler(FileCreated);
    WatchFile.Renamed += new RenamedEventHandler(FileReNamed);
    WatchFile.Deleted += new FileSystemEventHandler(FileDeleted);
    WatchFile.Changed += new FileSystemEventHandler(FileChanged);
    WatchFile.EnableRaisingEvents = true;
    WatchFile.IncludeSubdirectories = true;
}

public void FileCreated(object sender, FileSystemEventArgs e)
{
    WriteXML(e.ChangeType.ToString(), e.FullPath);
}

public void FileReNamed(object sender, RenamedEventArgs e)
{
    WriteXML(e.ChangeType.ToString(), e.FullPath);
}

public void FileDeleted(object sender, FileSystemEventArgs e)
{
    WriteXML(e.ChangeType.ToString(), e.FullPath);
}

public void FileChanged(object sender, FileSystemEventArgs e)
{
    WriteXML(e.ChangeType.ToString(), e.FullPath);
}

public static string encodeString(string original)
{
    //Based on method by Rick van den Bosch. Found at http://bloggingabout.net/blogs/rick/archive/2005/05/18/4118.aspx
    Byte[] originalBytes;
    Byte[] encodedBytes;

    //Generate hash
    SHA256 sha = new SHA256CryptoServiceProvider();
    originalBytes = ASCIIEncoding.Default.GetBytes(original);
    encodedBytes = sha.ComputeHash(originalBytes);

    //Convert encoded bytes back to a string
    return BitConverter.ToString(encodedBytes);
}

private static void WriteXML(string eventType, string filePath)
{
    //This method creates the XML log file if required and writes new entries to the file
    try
    {
        XmlDocument xmlDoc = new XmlDocument();

        try
        {
            xmlDoc.Load(filename);
        }
        catch
        {
            xmlDoc = new XmlDocument();
        }

        //Create new node
        XmlElement xmlEvent = xmlDoc.CreateElement("event");
        xmlEvent.SetAttribute("type", eventType);
        xmlEvent.SetAttribute("path", filePath);
        xmlDoc.DocumentElement.AppendChild(xmlEvent);

        //Save the file
        xmlDoc.Save(filename);
    }
    catch
    {
        //Catch any exceptions
        throw;
    }
}
try
{
    catch (System.IO.FileNotFoundException)
    {
        // make file if it cannot be found
        XmlTextWriter xmlWriter = new XmlTextWriter(filename,
System.Text.Encoding.UTF8);
        xmlWriter.Formatting = Formatting.Indented;
        xmlWriter.WriteProcessingInstruction("xml",
"version='1.0' encoding='UTF-8'");
        xmlWriter.WriteStartElement("filelog");
        xmlWriter.Close();
        xmlDoc.Load(filename);
    }
    DateTime timestamp = DateTime.Now.ToUniversalTime();

    string formattedTimestamp = timestamp.ToString("yyyyMMddHH:mm:ss");
    string checksum = formattedTimestamp +
System.Environment.UserName + SecurityElement.Escape(eventType) +
SecurityElement.Escape(filePath);
    // create sha-256 hash of record
    checksum = encodeString(checksum);

    XmlNode root = xmlDoc.DocumentElement;
    XmlElement childNode =
xmlDoc.CreateElement("recorded_event");
    XmlElement childNode2 = xmlDoc.CreateElement("event_time");
    XmlElement childNode3 =
xmlDoc.CreateElement("machine_name");
    XmlElement childNode4 =
xmlDoc.CreateElement("user_account");
    XmlElement childNode5 = xmlDoc.CreateElement("event");
    XmlElement childNode6 = xmlDoc.CreateElement("file_path");
    XmlElement childNode7 = xmlDoc.CreateElement("checksum");
    XmlText textNode2 =
xmlDoc.CreateTextNode(formattedTimestamp);
    XmlText textNode3 =
xmlDoc.CreateTextNode(System.Environment.MachineName);
    XmlText textNode4 =
System.Environment.UserName); 
    XmlText textNode5 =
xmlDoc.CreateTextNode(SecurityElement.Escape(eventType));
    XmlText textNode6 =
xmlDoc.CreateTextNode(SecurityElement.Escape(filePath));
    XmlText textNode7 = xmlDoc.CreateTextNode(checksum);
    root.AppendChild(childNode);
    childNode.AppendChild(childNode2);
    childNode2.AppendChild(textNode2);
    childNode.AppendChild(childNode3);
    childNode3.AppendChild(textNode3);
    childNode.AppendChild(childNode4);
    childNode4.AppendChild(textNode4);
    childNode.AppendChild(childNode5);
    childNode5.AppendChild(textNode5);
    childNode.AppendChild(childNode6);
    childNode6.AppendChild(textNode6);
    childNode.AppendChild(childNode7);
    childNode7.AppendChild(textNode7);
    try
{ xmlDoc.Save(filename); }
catch  //wait 10 milliseconds and try saving the file again if it failed the first time
{
    //appears to solve and issue whereby saving the XML file fails at the first attempt
    Thread.Sleep(10);
    xmlDoc.Save(filename);
}
catch (Exception ex)
//write out exception message to console if writing XML fails
{
    Console.WriteLine("Error writing to log \n " + ex);
}
}
### 3.2.3 Process Logger

**Program.cs**

```csharp
namespace Project_ProcessLogger
{
    class Program
    {
        static void Main(string[] args)
        {
            //create a new logger object
            Logger log = new Logger();
        }
    }
}
```

**Logger.cs**

```csharp
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Xml;
using System.Security;
using System.Threading;
using System.Diagnostics;
using System.Collections;
namespace Project_ProcessLogger
{
    class Logger
    {
        //timestamp set to standard format for entries in XML
        static string formatted_timestamp = DateTime.Now.ToUniversalTime().ToString("yyyy-MM-dd HH:mm:ss"); //universal
time
        //set the filepath to the log storage directory and filename to be
        //filename includes timestamp of when the logger was launched, not
        the time of the first recorded event
        static string filename = "logs\ProcessLog_" + formatted_timestamp.Replace(":", "-") + ".xml"; //have to remove :
        //character as it is illegal in a filename

        public Logger()
        {
            //upon creation run the main method to start logging
            Run();
        }

        private static void Run()
        {
            //Based on "Writing a simple Windows Process Logger in C#" from
            //run logger
            int iCycleInterval = 1000; // 1 second
            List<string> processesBook = new List<string>();
        }
    }
}
```
bool bFirstCycle = true;

Console.WriteLine("Process Logger is Running");

while (true)
{
    Process[] currentlyRunning = Process.GetProcesses();

    List<string> currentlyRunningProcessNames = new List<string>();
    // Record starting time for new processes

    foreach (Process process in currentlyRunning)
    {
        if (!processesBook.Contains(process.ProcessName)) // A new process started
        {
            // Log the Now as the start time for this process
            (Don't log if this is the first cycle)
            if (!bFirstCycle)
            {
                WriteXML("started", process.ProcessName);
                // Book-keep the new process
                processesBook.Add(process.ProcessName);
            }
            // Record the currently running process name for the next for comparison with our book (see he foreach loop below)
            currentlyRunningProcessNames.Add(process.ProcessName);
        }
    }

    // We looped already once, so turn this flag off
    bFirstCycle = false;

    // Record ending time for finished processes
    foreach (String bookKeptProcess in processesBook)
    {
        if (!currentlyRunningProcessNames.Contains(bookKeptProcess))
        {
            // Log the Now as the end time for this process
            WriteXML("ended", bookKeptProcess);
        }
    }

    // Now Update the process book
processesBook = currentlyRunningProcessNames;

// Sleep till the next loop
Thread.Sleep(iCycleInterval);

public static string encodeString(string original)
{
    // Based on method by Rick van den Bosch. Found at http://bloggingabout.net/blogs/rick/archive/2005/05/18/4118.aspx
    Byte[] originalBytes;
    Byte[] encodedBytes;

    // Generate hash
    SHA256 sha = new SHA256CryptoServiceProvider();
    originalBytes = ASCIIEncoding.Default.GetBytes(original);
    encodedBytes = sha.ComputeHash(originalBytes);

    // Convert encoded bytes back to a string
    return BitConverter.ToString(encodedBytes);
}

private static void WriteXML(string eventType, string processName)
{
    // This class creates the XML log file if required and writes new entries to the file
    try
    {
        XmlDocument xmlDoc = new XmlDocument();

        try
        {
            xmlDoc.Load(filename);
        }
        catch (System.IO.FileNotFoundException)
        {
            // Make file if it cannot be found
            XmlTextWriter xmlWriter = new XmlTextWriter(filename, System.Text.Encoding.UTF8);
            xmlWriter.Formatting = Formatting.Indented;
            xmlWriter.WriteProcessingInstruction("xml", "version='1.0' encoding='UTF-8'");
            xmlWriter.WriteStartElement("processlog");
            xmlWriter.Close();
            xmlDoc.Load(filename);
        }

        DateTime timestamp = DateTime.Now.ToUniversalTime();

        string formattedTimestamp = timestamp.ToString("yyyy-MM-dd HH:mm:ss");

        // Create sha-256 hash
        checksum = encodeString(checksum);
XmlNode root = xmlDoc.DocumentElement;
XmlElement childNode = 
xmlDoc.CreateElement("recorded_event");
XmlElement childNode2 = xmlDoc.CreateElement("event_time");
XmlElement childNode3 = 
xmlDoc.CreateElement("machine_name");
XmlElement childNode4 = 
xmlDoc.CreateElement("user_account");
XmlElement childNode5 = xmlDoc.CreateElement("event");
XmlElement childNode6 = 
xmlDoc.CreateElement("process_name");
XmlElement childNode7 = xmlDoc.CreateElement("checksum");
XmlText textNode2 = 
xmlDoc.CreateTextNode(formattedTimestamp);
XmlText textNode3 =
xmlDoc.CreateTextNode(System.Environment.MachineName);
XmlText textNode4 =
System.Environment.UserName);
XmlText textNode5 = 
xmlDoc.CreateTextNode(SecurityElement.Escape(eventType));
XmlText textNode6 = 
xmlDoc.CreateTextNode(SecurityElement.Escape(processName));
XmlNode textNode7 = xmlDoc.CreateTextNode(checksum);
root.AppendChild(childNode);
childNode.AppendChild(childNode2);
childNode2.AppendChild(textNode2);
childNode.AppendChild(childNode3);
childNode3.AppendChild(textNode3);
childNode.AppendChild(childNode4);
childNode4.AppendChild(textNode4);
childNode.AppendChild(childNode5);
childNode5.AppendChild(textNode5);
childNode.AppendChild(childNode6);
childNode6.AppendChild(textNode6);
childNode.AppendChild(childNode7);
childNode7.AppendChild(textNode7);
//save the file once new entries have been inserted
xmlDoc.Save(filename);
}
catch (Exception)
//write out exception message to console if writing xml fails
{
    Console.WriteLine("Error writing to log");
}
3.2.4 Log and Timeline Viewer

Window1.xaml

<Window x:Class="Project_Viewer.Window1"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:dg="http://schemas.microsoft.com/wpf/2008/toolkit"
    xmlns:zed="clr-namespace:ZedGraph;assembly=ZedGraph"
    Title="Log and Timeline Viewer"
    Height="700"
    Width="900"
    MinWidth="900"
    MinHeight="700"
    WindowStartupLocation="CenterScreen">
    <Grid>
        <TabControl Name="tabMyTabs" Margin="10">
            <TabItem Header="Dashboard">
                <Grid>
                    <Label Height="37" Margin="34,74,203,0" Name="dbstatsLabel" VerticalAlignment="Top" Panel.ZIndex="1"
                        FontSize="18">Dashboard</Label>
                    <Rectangle Name="rectangle1" Stroke="LightGray" Margin="6" Fill="White" />
                    <Label Height="28" HorizontalAlignment="Left" Margin="34,117,0,0" Name="numberRecordsLabel" VerticalAlignment="Top"
                        Width="311">0 records in total</Label>
                    <Label Height="28" HorizontalAlignment="Left" Margin="34,151,0,0" Name="numberApplicationLogRecordsLabel"
                        VerticalAlignment="Top" Width="311">0 application log records</Label>
                    <Label Height="28" HorizontalAlignment="Left" Margin="34,185,0,0" Name="numberSystemLogRecordsLabel"
                        VerticalAlignment="Top" Width="311">0 system log records</Label>
                    <Label Height="28" HorizontalAlignment="Left" Margin="34,219,0,0" Name="numberSecurityLogRecordsLabel"
                        VerticalAlignment="Top" Width="311">0 security log records</Label>
                    <Label Height="28" HorizontalAlignment="Left" Margin="34,253,0,0" Name="numberFileLogRecordsLabel"
                        VerticalAlignment="Top" Width="311">0 file log records</Label>
                    <Image Margin="330,151,98,0" Name="dashPieChart" Stretch="Fill" Height="150"
                        VerticalAlignment="Top" Width="420" />
                    <Label Height="28" Margin="520,0,98,275" Name="googleChartLabel" VerticalAlignment="Bottom"
                        FontSize="10">Google Charts Pie Chart (requires internet access)</Label>
                </Grid>
            </TabItem>
            <TabItem Header="Load Logs" Name="loadTab">
                <Grid>
                    <Rectangle Name="rectangle2" Stroke="LightGray" Margin="6" Fill="White" Panel.ZIndex="-1" />
                    <Button Height="23" Margin="321,159,0,0" Name="loadFileLogButton" VerticalAlignment="Top" HorizontalAlignment="Left"
                        Width="75" Visibility="Hidden" Click="loadFileLogButton_Click">Load</Button>
                    <TextBox Height="23" Margin="34,159,0,0" Name="loadFileLogTextBox" VerticalAlignment="Top"
                        IsReadOnly="True" HorizontalAlignment="Left" Width="200" FlowDirection="RightToLeft" />
                    <Button Height="23" Margin="240,159,0,0" Name="browseFileLogButton" VerticalAlignment="Top"
                        HorizontalAlignment="Left" Width="75" Click="browseFileLogButton_Click">Browse</Button>
                </Grid>
            </TabItem>
        </TabControl>
    </Grid>
</Window>
<Label FontSize="18" Height="37" Margin="34,74,203,0" Name="loadLogsLabel" Panel.ZIndex="1" VerticalAlignment="Top">Load Logs</Label>
</Grid>

<TabItem Header="Explore Data" Name="exploreTab" Visibility="Hidden">
  <Grid>
    <dg:DataGrid Margin="6,157,6,296" Name="logDataGrid" ItemsSource="{(Binding)}" AutoGenerateColumns="True" IsReadOnly="True" VerticalGridLinesBrush="LightGray" HorizontalGridLinesBrush="LightGray" IsEnabled="True" IsHitTestVisible="True" HeadersVisibility="Column" GridLinesVisibility="Horizontal" AlternationCount="0" Background="#FFF0F0F0" RowBackground="White" AlternatingRowBackground="WhiteSmoke" IsTextSearchEnabled="True" />
  </Grid>
</TabItem>

<GroupBox Header="Chart and Filter Settings" Height="145" Margin="6,6,6,0" Name="filterGroupBox" VerticalAlignment="Top">
  <Grid Width="824" Height="109">
    <ComboBox HorizontalAlignment="Right" Margin="0,11,582,0" Name="selectRecordComboBox" IsReadOnly="True" SelectedIndex="0" Margin="0,11,582,0" Height="24" VerticalAlignment="Top">
      <ComboBoxItem>All Records</ComboBoxItem>
      <ComboBoxItem>File and Process Logs</ComboBoxItem>
      <ComboBoxItem>Application Event Log</ComboBoxItem>
      <ComboBoxItem>Security Event Log</ComboBoxItem>
      <ComboBoxItem>System Event Log</ComboBoxItem>
      <ComboBoxItem>File Watcher Log</ComboBoxItem>
      <ComboBoxItem>Process Log</ComboBoxItem>
    </ComboBox>
  </Grid>
</GroupBox>

<Label HorizontalAlignment="Left" Margin="-2,9,0,0" Name="selectionRecordsLabel" Width="98" Height="26" VerticalAlignment="Top">Select Records:</Label>
<Label Margin="268,8,0,0" Name="dateFromLabel" Width="75" HorizontalAlignment="Left" Height="26" VerticalAlignment="Top">Date From:</Label>
<Label Margin="0,8,280,0" Name="dateToLabel" HorizontalAlignment="Right" Width="60" Height="24" VerticalAlignment="Top">Date To:</Label>
<Button HorizontalAlignment="Right" Margin="0,11,17,39.96" Name="filterButton" IsDefault="True">Filter</Button>
<dg:DatePicker Margin="345,10,366,0" Name="dateFromPicker" SelectedDateChanged="dateFromPicker_SelectedDateChanged" Height="24.04" VerticalAlignment="Top" />
<dg:DatePicker Margin="0,10,161,0" Name="dateToPicker" Width="113" HorizontalAlignment="Right" Height="24" VerticalAlignment="Top" />
<Label Height="28" HorizontalAlignment="Left" Margin="6,0,0,2" Name="advancedQueryLabel" VerticalAlignment="Bottom" Width="156" Advanced Query (optional):</Label>
<TextBox Height="24" Margin="168,0,161,0" Name="filterTextBox" VerticalAlignment="Bottom" />
<Button Height="24" Margin="0,0,140,0" Name="helpButton1" VerticalAlignment="Bottom" Width="15" Click="helpButton1_Click">?</Button>
<ComboBox Margin="0,47,366,38" Name="chartTypeComboBox" SelectedIndex="0" HorizontalAlignment="Left" Width="150" SelectionChanged="chartTypeComboBox_SelectionChanged">
    <ComboBoxItem>Bar Chart</ComboBoxItem>
    <ComboBoxItem>Line Chart</ComboBoxItem>
    <ComboBoxItem>Scatter Chart</ComboBoxItem>
</ComboBox>
<Label Margin="15,47,0,36" Name="chartTypeLabel" Width="71">Chart Type:</Label>
<ComboBox Margin="0,47,162,38" Name="redLevelComboBox" SelectedIndex="6" HorizontalAlignment="Right" Width="112">
    <ComboBoxItem>5</ComboBoxItem>
    <ComboBoxItem>10</ComboBoxItem>
    <ComboBoxItem>20</ComboBoxItem>
    <ComboBoxItem>50</ComboBoxItem>
    <ComboBoxItem>100</ComboBoxItem>
    <ComboBoxItem>200</ComboBoxItem>
    <ComboBoxItem>500</ComboBoxItem>
    <ComboBoxItem>1000</ComboBoxItem>
    <ComboBoxItem>2000</ComboBoxItem>
    <ComboBoxItem>5000</ComboBoxItem>
</ComboBox>
<Label Margin="0,47,276,38" Name="redLevelLabel" Width="64">Red Level:</Label>
<ComboBox Margin="0,47,366,38" Name="resetComboBox" SelectedIndex="0" HorizontalAlignment="Right" Width="84" IsEnabled="False">
    <ComboBoxItem>Yes</ComboBoxItem>
    <ComboBoxItem>No</ComboBoxItem>
</ComboBox>
</Grid>
</GroupBox>
</TabControl>
</Window>

**Window2.xaml**

```xml
<Window x:Class="Project_Viewer.Window2"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    Title="Help" Height="300" Width="400"
    WindowStartupLocation="CenterScreen">
    <Grid Background="WhiteSmoke">
        <TextBox Margin="12" Name="textBox1" TextWrapping="Wrap" />
    </Grid>
</Window>
```
Window1.xaml.cs

using System;
using System.Collections.Generic;
using System.Text;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Shapes;
using Microsoft.Win32;
using System.Xml;
using System.Data.SqlClient;
using System.Data;
using ZedGraph;
using System.Drawing;
using System.Threading;
using System.Diagnostics;
using System.Security;

namespace ProjectViewer
{
    /// <summary>
    /// Interaction logic for Window1.xaml
    /// </summary>
    public partial class Window1 : Window
    {
        //This is the main class which handles contains all the code behind
        UI elements,
        //importing log information and displaying results
        //sql connection properties
        public SqlConnection MyConnection;

        string connectionString = @"Data Source=localhost;Initial Catalog=Project; Trusted_Connection=True; Connection Timeout=1;";

        public delegate void MyDelegate();

        public Window1()
        {
            InitializeComponent();
            //check to see if the database is running
            checkDatabase();
        }

        private void makeDash()
        {
            //make a new dashboard, update user interface with totals and
            new pie chart
            Dashboard dash = new Dashboard();
            numberRecordsLabel.Content = dash.getTotalRecords().ToString() + " records in total";
            numberApplicationLogRecordsLabel.Content = dash.getApplicationRecords().ToString() + " application log records";
numberSecurityLogRecordsLabel.Content = dash.getSecurityRecords().ToString() + " security log records";
numberSystemLogRecordsLabel.Content = dash.getSystemRecords().ToString() + " system log records";
numberFileLogRecordsLabel.Content = dash.getFileRecords().ToString() + " file log records";
numberProcessLogRecordsLabel.Content = dash.getProcessRecords().ToString() + " process log records";
dashPieChart.Source = new BitmapImage(new Uri(dash.getAddress(), UriKind.Absolute));
dbstatsLabel.Content = "Dashboard - " + DateTime.UtcNow + " UTC";

private void checkDatabase()
// check to see if a connection can be made to the database
// if yes - check for rows - encourage user to import data if database is empty
// if no - warn user with message box
{
    MyConnection = new SqlConnection(connectionString);
    try
    {
        MyConnection.Open();
        SqlCommand myCommand = new SqlCommand("SELECT id FROM Hub", MyConnection);
        SqlDataReader DbReader = myCommand.ExecuteReader();
        DbReader.Read(); // no need to run in a loop as only wanting one row
        if (DbReader.HasRows != true) // test to check results can be found, if not query cannot find dates and will return NULL
        {
            MessageBox.Show("Database does not yet contain records. Load in some log data before filtering logs.", "Warning",
                           MessageBoxButtons.OK, MessageBoxIcon.Warning);
            exploreTab.Visibility = Visibility.Hidden; // hide the explore data tab if there are no records in the database
            tabMyTabs.SelectedItem = loadTab; // set tab to the loading tab to encourage user to load in data
        }
        else
        {
            fillFilterDates();
        }
    }
    catch
    {
        // connection cannot be made to database, warn user
        MessageBox.Show("Could not connect to database. Check database is running and try again.", "Database Error",
                           MessageBoxButtons.OK, MessageBoxIcon.Error);
    
    
filterLogs(chartTypeComboBox.SelectionBoxItem.ToString(),
redLevelComboBox.SelectionBoxItem.ToString());
makeDash();
exploreTab.Visibility = Visibility.Visible; // make the explore data tab visible as there should be something to filter
}
Application.Current.Shutdown(); // close application was it cannot function with the database
}
finally
{
    MyConnection.Close();
}

private void fillFilterDates()
{
    // set the available date range for the calendar elements based on min and max dates in database
    string minDate = "";
    string maxDate = "";

    MyConnection = new SqlConnection(connectionString);
    try
    {
        MyConnection.Open();
        SqlCommand myCommand = new SqlCommand("SELECT MIN(CONVERT(varchar,event_time,111)), MAX(CONVERT(varchar,event_time,111)) FROM Hub", MyConnection);

        SqlDataReader DbReader = myCommand.ExecuteReader();
        DbReader.Read(); // no need to run in a loop as only wanting one row

        if (DbReader.IsDBNull(0) != true) // test to check results can be found, if not then query cannot find dates and will return NULL
        {
            minDate = DbReader[0].ToString();
            maxDate = DbReader[1].ToString();

            DateTime myDateTime = DateTime.Parse(minDate);
            DateTime myDateTime2 = DateTime.Parse(maxDate);
            dateFromPicker.DisplayDateStart = myDateTime;
            dateFromPicker.DisplayDateEnd = myDateTime2;
            dateToPicker.DisplayDateStart = myDateTime;
            dateToPicker.DisplayDateEnd = myDateTime2;
            dateFromPicker.SelectedDate = myDateTime;
            dateToPicker.SelectedDate = myDateTime2;
        }
    }
    catch
    {
        // Warn user if database connection could not be setup, unlikely to get here as connection is test at application startup
        MessageBox.Show("Could not get dates from database. Perhaps the database does not yet contain log data.", "Error", MessageBoxButton.OK, MessageBoxImage.Error);
    }
}
MessageBox.Show("Could setup filter options. Check database server is running.", "Error", MessageBoxButton.OK, MessageBoxImage.Error);
return;
}
finally
{
    //ensure connection to database is closed
    MyConnection.Close();
}
}

private void browseFileLogButton_Click(object sender, RoutedEventArgs e)
{
    //show file browse dialogue when user clicks on browse button
    Microsoft.Win32.OpenFileDialog browseFile = new OpenFileDialog();
    browseFile.Filter = "XML Files (*.xml)|*.xml";
    browseFile.Title = "Browse XML file";
    if (browseFile.ShowDialog().Value)
    {
        try
        {
            //populate textbox with selected file name
            loadFileLogTextBox.Text = browseFile.FileName;
        }
        catch (Exception)
        {
            //handle any exceptions and display message to user
            MessageBox.Show("Error opening file. Check access privilages.", "File Warning", MessageBoxButton.OK, MessageBoxImage.Warning);
        }
    }
}

private void loadFileLogButton_Click(object sender, RoutedEventArgs e)
{
    //import log data into the database
    loadFileLogButton.Visibility = Visibility.Hidden;
    string latest = loadFileLogTextBox.Text;
    //new thread used as the UI thread will become unresponsive to the user otherwise
    Thread thread_run = new Thread(delegate()
    {
        loadXMLLog(latest, "filelog");
    });
    //set thread priority to background, UI should have higher priority
    thread_run.IsBackground = true;
    thread_run.Start();
}

public void loadXMLLog(string filePath, string log_type)
{
    //Method carries out the importing of XML log data into the appropriate database table
string insertHub = "";
string insertLog = "";
MyConnection = new SqlConnection(connectionString);
MyConnection.Open();
int i = 0;
//display loading message to user
updateLoaded(true);
try {
    XmlDocument xDoc = new XmlDocument();
xDoc.Load(filePath);
XmlNodeList n1 =
    xDoc.GetElementsByTagName("recorded_event");

    foreach (XmlNode process in n1) {
        XmlNodeList event_time =
            xDoc.GetElementsByTagName("event_time");
        XmlNodeList machine_name =
            xDoc.GetElementsByTagName("machine_name");
        XmlNodeList user_account =
            xDoc.GetElementsByTagName("user_account");
        XmlNodeList checksum =
            xDoc.GetElementsByTagName("checksum");
        if (log_type == "filelog") {
            XmlNodeList action =
                xDoc.GetElementsByTagName("event");
            XmlNodeList file_path =
                xDoc.GetElementsByTagName("file_path");
            insertHub = "INSERT INTO Hub (log_type, event_time,
                machine_name, user_account) VALUES ('FileLog','" +
                event_time[i].InnerText +
                ", '" + machine_name[i].InnerText +
                ", '" + user_account[i].InnerText +
                ");";
            insertLog = "INSERT INTO FileLog (id,
                event,file_path) VALUES (IDENT_CURRENT('Hub')," +
                action[i].InnerText +
                ", '" + file_path[i].InnerText +
                ");";
            i++;
        } else if (log_type == "applicationlog") {
            XmlNodeList event_id =
                xDoc.GetElementsByTagName("event_id");
            XmlNodeList event_data =
                xDoc.GetElementsByTagName("event_data");
            string safe_event_data =
                event_data[i].InnerText.Replace("'", "\'";
            // remove ' as they will
            // escape the SQL statement
            insertHub = "INSERT INTO Hub (log_type, event_time,
                machine_name, user_account) VALUES ('ApplicationLog','" +
                event_time[i].InnerText +
                ", '" + machine_name[i].InnerText +
                ", '" + user_account[i].InnerText +
                ");";
            insertLog = "INSERT INTO ApplicationLog (id,
                event_id, event_data) VALUES (IDENT_CURRENT('Hub')," +
                event_id[i].InnerText +
                ", " + safe_event_data + 
                ");";
            i++;
        } else if (log_type == "systemlog") {
            XmlNodeList event_id =
                xDoc.GetElementsByTagName("event_id");
XmlNodeList event_data =
    xDoc.GetElementsByTagName("event_data");

string safe_event_data =
    event_data[i].InnerText.Replace("'", "\'"); // remove ' as they will escape the SQL statement

    insertHub = "INSERT INTO Hub (log_type, event_time,
    machine_name, user_account) VALUES ('SystemLog','" +
    event_time[i].InnerText + "," + machine_name[i].InnerText + "," +
    user_account[i].InnerText + ");";

    insertLog = "INSERT INTO SystemLog (id, event_id,
    event_data) VALUES (IDENT_CURRENT('Hub')," + event_id[i].InnerText + "," +
    safe_event_data + ");";

    i++;
} else if (log_type == "securitylog")
{
    XmlNodeList event_id =
    xDoc.GetElementsByTagName("event_id");
    XmlNodeList event_data =
    xDoc.GetElementsByTagName("event_data");

    string safe_event_data =
    event_data[i].InnerText.Replace("'", "\'"); // remove ' as they will escape the SQL statement

    insertHub = "INSERT INTO Hub (log_type, event_time,
    machine_name, user_account) VALUES ('SecurityLog','" +
    event_time[i].InnerText + "," + machine_name[i].InnerText + "," +
    user_account[i].InnerText + ");";

    insertLog = "INSERT INTO SecurityLog (id, event_id,
    event_data) VALUES (IDENT_CURRENT('Hub')," + event_id[i].InnerText + "," +
    safe_event_data + ");";

    i++;
} else if (log_type == "processlog")
{
    XmlNodeList action =
    xDoc.GetElementsByTagName("event");
    XmlNodeList process_name =
    xDoc.GetElementsByTagName("process_name");

    insertHub = "INSERT INTO Hub (log_type, event_time,
    machine_name, user_account) VALUES ('ProcessLog','" +
    event_time[i].InnerText + "," + machine_name[i].InnerText + "," +
    user_account[i].InnerText + ");";

    insertLog = "INSERT INTO ProcessLog (id, event,
    process_name) VALUES (IDENT_CURRENT('Hub')," + action[i].InnerText + "," +
    process_name[i].InnerText + ");";

    i++;
}

//build SQL command for inserting data
//log data inserted into Hub and appropriate log tables
SqlCommand MyCommand =
    new SqlCommand(insertHub + insertLog,
    MyConnection);
MyCommand.ExecuteNonQuery();

} catch
{
    //handle any exceptions, likely cause is the wrong log type
    //has been selected. Message box to warn user of problem.
    MessageBox.Show("Error processing file\nCheck file selected matches the log type in question.", "File Processing Error",
    MessageBoxButtons.OK, MessageBoxIcon.Error);
finally
{
    // close database connection
    MyConnection.Close();
    // stop displaying loading message
    updateLoaded(false);
}

private void updateLoaded(Boolean loaded)
{
    // display message to user when loading in log files, set cursor
to wait style
    // delegate used to update UI elements
    if (loaded == true)
    {
        MyDelegate methodForUiThread = delegate
        {
            logLoadingLabel.Visibility = Visibility.Visible;
            Cursor = Cursors.Wait;
        };
        this.Dispatcher.BeginInvoke(System.Windows.Threading.DispatcherPriority.Normal, methodForUiThread);
    }
    else
    {
        MyDelegate methodForUiThread = delegate
        {
            logLoadingLabel.Visibility = Visibility.Hidden;
            Cursor = Cursors.Arrow;
            checkDatabase(); // refresh the filtering settings now
            that new data has finished loading into the database
            loadApplicationLogTextBox.Text = "";
            loadSystemLogTextBox.Text = ""; // clear out the
textboxes showing file to load location as file has been loaded
            loadFileLogTextBox.Text = "";
            loadSecurityLogTextBox.Text = "";
            loadProcessLogTextBox.Text = "";
        };
        this.Dispatcher.BeginInvoke(System.Windows.Threading.DispatcherPriority.Normal, methodForUiThread);
    }
}

private void filterButton_Click(object sender, RoutedEventArgs e)
{
    string chartType =
    chartTypeComboBox.SelectionBoxItem.ToString();
    string redLevel = redLevelComboBox.SelectionBoxItem.ToString();
    filterLogs(chartType, redLevel);
}

private void filterLogs(string chartType, string redLevel)
{
    // get filter settings set by the user and request log
    information (datagrid and chart) from ChartGridFill class
//fills datagrid with contents of datatable returned from ChartGridFill
string dfdf = dateFromPicker.SelectedDate.Value.Year.ToString() + "-" + dateFromPicker.SelectedDate.Value.Month.ToString() + "-" + dateFromPicker.SelectedDate.Value.Day.ToString() + " 00:00:00";
string logType = "";
string filter = filterTextBox.Text;
bool reset;
if (resetComboBox.SelectedIndex == 0)
{
    reset = true;
}
else
{
    reset = false;
}
//remove characters which could interfere with sql statement
filter = filter.Replace(""'", "'");
filter = filter.Replace(";", "");
//pass filter options to appropriate method in ChartGridFill
if (selectRecordComboBox.SelectionBoxItem.ToString().Equals("File Watcher Log"))
{
    logType = "FileLog";
    ChartGridFill dataToFill = new ChartGridFill(zg1, dfdf, dtdt, logType, chartType, redLevel, filter, reset);
    logDataGrid.DataContext = dataToFill.dt;
}
else if (selectRecordComboBox.SelectionBoxItem.ToString().Equals("File and Process Logs"))
{
    logType = "FileProcess";
    ChartGridFill dataToFill = new ChartGridFill(zg1, dfdf, dtdt, logType, chartType, redLevel, filter, reset);
    logDataGrid.DataContext = dataToFill.dt;
}
else if (selectRecordComboBox.SelectionBoxItem.ToString().Equals("Application Event Log"))
{
    logType = "ApplicationLog";
    ChartGridFill dataToFill = new ChartGridFill(zg1, dfdf, dtdt, logType, chartType, redLevel, filter, reset);
    logDataGrid.DataContext = dataToFill.dt;
}
else if (selectRecordComboBox.SelectionBoxItem.ToString().Equals("System Event Log"))
{
    logType = "SystemLog";
    ChartGridFill dataToFill = new ChartGridFill(zg1, dfdf, dtdt, logType, chartType, redLevel, filter, reset);
    logDataGrid.DataContext = dataToFill.dt;
else if (selectRecordComboBox.SelectionBoxItem.ToString().Equals("Security Event Log"))
{
    logType = "SecurityLog";
    ChartGridFill dataToFill = new ChartGridFill(zg1, ddf, dtdt, logType, chartType, redLevel, filter, reset);
    logDataGrid.DataContext = dataToFill.dt;
}
else if (selectRecordComboBox.SelectionBoxItem.ToString().Equals("Process Log"))
{
    logType = "ProcessLog";
    ChartGridFill dataToFill = new ChartGridFill(zg1, ddf, dtdt, logType, chartType, redLevel, filter, reset);
    logDataGrid.DataContext = dataToFill.dt;
}
else
{
    ChartGridFill dataToFill = new ChartGridFill(zg1, ddf, dtdt, chartType, redLevel, filter, reset);
    logDataGrid.DataContext = dataToFill.dt;
}

private void browseApplicationLogButton_Click(object sender, RoutedEventArgs e)
{
    Microsoft.Win32.OpenFileDialog browseFile = new OpenFileDialog();
    browseFile.Filter = "XML Files (*.xml)|*.xml";
    browseFile.Title = "Browse XML file";
    if (browseFile.ShowDialog().Value)
    {
        try
        {
            loadApplicationLogTextBox.Text = browseFile.FileName;
            loadApplicationLogButton.Visibility = Visibility.Visible;
        }
        catch (Exception)
        {
            MessageBox.Show("Error opening file", "File Error", MessageBoxButton.OK, MessageBoxIcon.Error);
        }
    }
}

private void loadApplicationLogButton_Click(object sender, RoutedEventArgs e)
{
    loadApplicationLogButton.Visibility = Visibility.Hidden;
    string latest = loadApplicationLogTextBox.Text;

    Thread thread_run = new Thread(delegate()
    {

    }}
loadXMLLog(latest, "applicationlog");
}
thread_run.IsBackground = true;
thread_run.Start();
}

private void browseSystemLogButton_Click(object sender, RoutedEventArgs e)
{
    Microsoft.Win32.OpenFileDialog browseFile = new OpenFileDialog();
    browseFile.Filter = "XML Files (*.xml)|*.xml";
    browseFile.Title = "Browse XML file";
    if (browseFile.ShowDialog().Value)
    {
        try
        {
            loadSystemLogTextBox.Text = browseFile.FileName;
        }
        catch (Exception)
        {
            MessageBox.Show("Error opening file", "File Error", MessageBoxButton.OK, MessageBoxImage.Warning);
        }
    }
}

private void loadSystemLogButton_Click(object sender, RoutedEventArgs e)
{
    loadSystemLogButton.Visibility = Visibility.Hidden;
    string latest = loadSystemLogTextBox.Text;

    Thread thread_run = new Thread(delegate()
    {
        loadXMLLog(latest, "systemlog");
    });
    thread_run.IsBackground = true;
    thread_run.Start();
}

private void browseSecurityLogButton_Click(object sender, RoutedEventArgs e)
{
    Microsoft.Win32.OpenFileDialog browseFile = new OpenFileDialog();
    browseFile.Filter = "XML Files (*.xml)|*.xml";
    browseFile.Title = "Browse XML file";
    if (browseFile.ShowDialog().Value)
    {
        try
        {
            loadSecurityLogTextBox.Text = browseFile.FileName;
        }
        catch (Exception)
        {
            MessageBox.Show("Error opening file", "File Error", MessageBoxButton.OK, MessageBoxImage.Warning);
        }
    }
}
private void loadSecurityLogButton_Click(object sender, RoutedEventArgs e)
{
    string latest = loadSecurityLogTextBox.Text;
    Thread thread_run = new Thread(delegate()
    {
        loadXMLLog(latest, "securitylog");
    });
    thread_run.IsBackground = true;
    thread_run.Start();
}

private void browseProcessLogButton_Click(object sender, RoutedEventArgs e)
{
    Microsoft.Win32.OpenFileDialog browseFile = new OpenFileDialog();
    browseFile.Filter = "XML Files (*.xml)|*.xml";
    browseFile.Title = "Browse XML file";
    if (browseFile.ShowDialog().Value)
    {
        try
        {
            loadProcessLogTextBox.Text = browseFile.FileName;
        }
        catch (Exception)
        {
            MessageBox.Show("Error opening file", "File Error",
            MessageBoxButton.OK, MessageBoxIcon.Warning);
        }
    }
}

private void loadProcessLogButton_Click(object sender, RoutedEventArgs e)
{
    string latest = loadProcessLogTextBox.Text;
    Thread thread_run = new Thread(delegate()
    {
        loadXMLLog(latest, "processlog");
    });
    thread_run.IsBackground = true;
    thread_run.Start();
}
private void dateFromPicker_SelectedDateChanged(object sender, SelectionChangedEventArgs e)
{
    // This method ensures the user cannot select a date to that is
    // before the from date by updating
    // the date to minimum display date to equal the from selected
date
    // If date to is less than date from then date to is updated to
    // equal date from
    if (dateToPicker.SelectedDate < dateFromPicker.SelectedDate)
    {
        dateToPicker.SelectedDate = dateFromPicker.SelectedDate;
        dateToPicker.DisplayDateStart = dateFromPicker.SelectedDate;
    }
}

private void helpButton1_Click(object sender, RoutedEventArgs e)
{
    // Create a window
    Window2 window = new Window2();

    // Open a help window
    window.Show();
}

private void chartTypeComboBox_SelectionChanged(object sender, SelectionChangedEventArgs e)
{
    if (chartTypeComboBox.SelectedIndex == 1)
    {
        // if line chart allow user to change whether to reset the
        // graph or not when adding new graphs
        resetComboBox.IsEnabled = true;
    }
    if (chartTypeComboBox.SelectedIndex != 1)
    {
        if (resetComboBox is ComboBox)
        {
            ((ComboBox)resetComboBox).IsEnabled = false;
            ((ComboBox)resetComboBox).SelectedIndex = 0;
        }
    }
}
}
**Window2.xaml.cs**

```csharp
using System;
using System.Text;
using System.Windows;

namespace Project_Viewer
{
    /// <summary>
    /// Interaction logic for Window2.xaml
    /// </summary>
    public partial class Window2 : Window
    {
        public Window2()
        {
            InitializeComponent();
            //fill textbox with the help text
            fillTextBox();
        }

        private void fillTextBox()
        {
            //provides basic help to the user on how to use the advanced
            //query option
            textBox1.Text += "The advanced query option allows you to find
            specific results within the database. \n";
            textBox1.Text += "Querys should be entered in an SQL fashion
            for example:\n            t AND Machine_Name LIKE 'ExampleName'\n";
            textBox1.Text += "Fields which can be filtered using this
            method include:\nmachine_name, event_time, user_account, event, event_data,
            file_path and process_name";
        }
    }
}
```

**Dashboard.cs**

```csharp
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Data.SqlClient;
using System.Windows;
using ZedGraph;
using System.Drawing;

namespace Project_Viewer
{
    class Dashboard
    {
        //This class is concerned with getting the total records in the
database for the dashboard
        //Totals are and Google Chart address are returned to
        //Window1.xaml.cs which updates the UI with the new totals and pie chart

        public double ptotal = 0;
        public double papplication = 0;
        public double psecurity = 0;
        public double psystem = 0;
        public double pfile = 0;
        public double pprocess = 0;
        public string pieAddress = "";
    }
}
```
// sql connection properties

public SqlConnection MyConnection;

string connectionString = @"Data Source=localhost;Initial Catalog=Project; Trusted_Connection=True; Connection Timeout=1;";

public string getAddress()
{
    pieAddress = fillChart();
    return pieAddress;
}

public string getTotalRecords()
{
    string numberRecords = ""
    MyConnection = new SqlConnection(connectionString);
    try
    {
        MyConnection.Open();
        SqlCommand myCommand = new SqlCommand("SELECT COUNT (id) FROM Hub", MyConnection);
        SqlDataReader DbReader = myCommand.ExecuteReader();
        DbReader.Read(); // no need to run in a loop as only wanting one row

        if (DbReader.IsDBNull(0) != true) // test to check results can be found, if not then query cannot find any ids, will return NULL
        {
            numberRecords = DbReader[0].ToString();
            ptotal = Int32.Parse(DbReader[0].ToString());
        }
    }
    catch
    {
        numberRecords = "Unknown number of"; // shouldn't get to here as this method is not run unless database connection test has already been passed
    }
    finally
    {
        MyConnection.Close();
    }
    return numberRecords;
}

public string getApplicationRecords()
{
    string numberRecords = ""
    MyConnection = new SqlConnection(connectionString);
    try
    {
        MyConnection.Open();
        SqlCommand myCommand = new SqlCommand("SELECT COUNT (id) FROM ApplicationLog", MyConnection);
        SqlDataReader DbReader = myCommand.ExecuteReader();
        DbReader.Read(); // no need to run in a loop as only wanting one row

if (DbReader.IsDBNull(0) != true) // test to check results can be found, if not then query cannot find any ids, will return NULL
{
    numberRecords = DbReader[0].ToString();
    papplication = Int32.Parse(DbReader[0].ToString());
}
} catch
{
    numberRecords = "Unknown number of"; // shouldn't get to here as this method is not run unless database connection test has already been passed
} finally
{
    MyConnection.Close();
    return numberRecords;
}

public string getSystemRecords()
{
    string numberRecords = "";
    MyConnection = new SqlConnection(connectionString);
    try
    {
        MyConnection.Open();
        SqlCommand myCommand = new SqlCommand("SELECT COUNT (id)
FROM SystemLog", MyConnection);
        SqlDataReader DbReader = myCommand.ExecuteReader();
        DbReader.Read(); // no need to run in a loop as only wanting one row
        if (DbReader.IsDBNull(0) != true) // test to check results can be found, if not then query cannot find any ids, will return NULL
        {
            numberRecords = DbReader[0].ToString();
            psystem = Int32.Parse(DbReader[0].ToString());
        }
    } catch
    {
        numberRecords = "Unknown number of"; // shouldn't get to here as this method is not run unless database connection test has already been passed
    } finally
    {
        MyConnection.Close();
        return numberRecords;
    }
SqlCommand myCommand = new SqlCommand("SELECT COUNT (id)
FROM SecurityLog", MyConnection);

SqlDataReader DbReader = myCommand.ExecuteReader();
DbReader.Read(); //no need to run in a loop as only wanting one row

if (DbReader.IsDBNull(0) != true) //test to check results can be found, if not then query cannot find any ids, will return NULL
{
    numberRecords = DbReader[0].ToString();
    psecurity = Int32.Parse(DbReader[0].ToString());
}
} catch
{
    numberRecords = "Unknown number of"; // shouldn't get to here as this method is not run unless database connection test has already been passed
} finally
{
    MyConnection.Close();
}
return numberRecords;

public string getFileRecords()
{
    string numberRecords = "";
    MyConnection = new SqlConnection(connectionString);
    try
    {
        MyConnection.Open();
        SqlCommand myCommand = new SqlCommand("SELECT COUNT (id)
FROM FileLog", MyConnection);
        SqlDataReader DbReader = myCommand.ExecuteReader();
        DbReader.Read(); //no need to run in a loop as only wanting one row

        if (DbReader.IsDBNull(0) != true) //test to check results can be found, if not then query cannot find any ids, will return NULL
        {
            numberRecords = DbReader[0].ToString();
            pfile = Int32.Parse(DbReader[0].ToString());
        }
    } catch
    {
        numberRecords = "Unknown number of"; // shouldn't get to here as this method is not run unless database connection test has already been passed
    } finally
    {
        MyConnection.Close();
    }
    return numberRecords;
}
public string getProcessRecords()
{
    string numberRecords = "";
    SqlConnection MyConnection = new SqlConnection(connectionString);
    try
    {
        MyConnection.Open();
        SqlCommand myCommand = new SqlCommand("SELECT COUNT (id)
FROM ProcessLog", MyConnection);
        SqlDataReader DbReader = myCommand.ExecuteReader();
        DbReader.Read(); //no need to run in a loop as only wanting
        one row
        if (DbReader.IsDBNull(0) != true) //test to check results
        can be found, if not then query cannot find any ids, will return NULL
        {
            numberRecords = DbReader[0].ToString();
            pprocess = Int32.Parse(DbReader[0].ToString());
        }
    }
    catch
    {
        numberRecords = "Unknown number of"; // shouldn't get to
        here as this method is not run unless database connection test has already
        been passed
    }
    finally
    {
        MyConnection.Close();
    }
    return numberRecords;
}

private string fillChart()
{
    //this method produces a pie chart on the dashboard
    //the Google Charts api is used

    //work out totals in the form of percentages
    double tapplication = ((papplication * 100) / ptotal);
    double tsecurity = ((psecurity * 100) / ptotal);
    double tsystem = ((psystem * 100) / ptotal);
    double tfile = ((pfile * 100) / ptotal);
    double tprocess = ((pprocess * 100) / ptotal);
    string address = "";
    //return the address (png image)
    address = "http://chart.apis.google.com/chart?chco=6495ED&cht=p3&chl=t:" +
    tapplication + "," + tsecurity + "," + tsystem + "," + tfile + "," +
tprocess + ",chs=420x150&chl=App Log " + tapplication.ToString("#0.00") +
"%" + "|Sec Log " + tsecurity.ToString("#0.00") + "%" + "|Sys Log " +
tsystem.ToString("#0.00") + "%" + "|File Log " + tfile.ToString("#0.00") +
"%" + "|Proc Log " + tprocess.ToString("#0.00") + "%"; //convert to 2
decimal points
    return address;
}
using ZedGraph;
using System;
using System.Windows;
using System.Data.SqlClient;
using System.Data;

public class ChartGridFill
{
    //sql connection properties
    public SqlConnection MyConnection;

    string connectionString = @"Data Source=localhost;Initial Catalog=Project; Trusted_Connection=True; Connection Timeout=1;";

    //create a datatable to store SQL query results in (log data)
    public DataTable dt = new DataTable();

    public ZedGraphControl z1;
    public string dateFrom;
    public string dateTo;
    public string chartType;
    public string whichLog;
    public string redLevel;
    public string filter;
    public bool reset;

    public ChartGridFill(ZedGraphControl z1, string dateFrom, string dateTo, string chartType, string redLevel, string filter, bool reset)
    {
        this.z1 = z1;
        this.dateFrom = dateFrom;
        this.dateTo = dateTo;
        this.chartType = chartType;
        this.redLevel = redLevel;
        this.filter = filter;
        this.reset = reset;
        this.whichLog = "All";

        fillChart();
        fillGrid();
    }

    public ChartGridFill(ZedGraphControl z1, string dateFrom, string dateTo, string whichLog, string chartType, string redLevel, string filter, bool reset)
    {
        this.z1 = z1;
        this.dateFrom = dateFrom;
        this.dateTo = dateTo;
        this.chartType = chartType;
        this.redLevel = redLevel;
        this.filter = filter;
        this.reset = reset;
        this.whichLog = whichLog;

        fillChart();
        fillGrid();
    }
}
public void fillChart()
{
    //This method creates the chart based on the log data requested
    //Uses the ZedGraph library to produce chart

    //SQL query to get totals for each day based on log type user has selected

    string sql = "";
    if (whichLog == "FileLog")
    {
        sql = "SELECT CONVERT (varchar,Hub.event_time,102) as dates,
        COUNT (event_time) as 'counted' FROM Hub, FileLog WHERE Hub.event_time
        BETWEEN " + dateFrom + " AND " + dateTo + " AND Hub.id = FileLog.id 
        + filter + " GROUP BY CONVERT (varchar,Hub.event_time,102) ORDER BY dates";
    }
    else if (whichLog == "FileProcess")
    {
        sql = "SELECT CONVERT (varchar,Hub.event_time,102) as dates,
        COUNT (Hub.id) as 'counted' FROM Hub FULL OUTER JOIN FileLog ON Hub.id =
        ProcessLog.id FULL OUTER JOIN FileLog ON Hub.id = FileLog.id WHERE
        Hub.log_type = 'FileLog' OR Hub.log_type = 'ProcessLog' AND Hub.event_time
        BETWEEN " + dateFrom + " AND " + dateTo + " + filter + " GROUP BY
        CONVERT (varchar,Hub.event_time,102) ORDER BY dates";
    }
    else if (whichLog == "ApplicationLog")
    {
        sql = "SELECT CONVERT (varchar,Hub.event_time,102) as dates,
        COUNT (event_time) as 'counted' FROM Hub, ApplicationLog WHERE
        Hub.event_time BETWEEN " + dateFrom + " AND " + dateTo + " AND Hub.id =
        ApplicationLog.id " + filter + " GROUP BY CONVERT (varchar,Hub.event_time,102) ORDER BY dates";
    }
    else if (whichLog == "SystemLog")
    {
        sql = "SELECT CONVERT (varchar,Hub.event_time,102) as dates,
        COUNT (event_time) as 'counted' FROM Hub, SystemLog WHERE
        Hub.event_time BETWEEN " + dateFrom + " AND " + dateTo + " AND Hub.id =
        SystemLog.id " + filter + " GROUP BY CONVERT (varchar,Hub.event_time,102) ORDER BY dates";
    }
    else if (whichLog == "SecurityLog")
    {
        sql = "SELECT CONVERT (varchar,Hub.event_time,102) as dates,
        COUNT (event_time) as 'counted' FROM Hub, SecurityLog WHERE
        Hub.event_time BETWEEN " + dateFrom + " AND " + dateTo + " AND Hub.id =
        SecurityLog.id " + filter + " GROUP BY CONVERT (varchar,Hub.event_time,102) ORDER BY dates";
    }
    else if (whichLog == "ProcessLog")
    {
        sql = "SELECT CONVERT (varchar,Hub.event_time,102) as dates,
        COUNT (event_time) as 'counted' FROM Hub, ProcessLog WHERE Hub.event_time
        BETWEEN " + dateFrom + " AND " + dateTo + " AND Hub.id = ProcessLog.id 
        + filter + " GROUP BY CONVERT (varchar,Hub.event_time,102) ORDER BY dates";
    }
    else
    {
        sql = "SELECT CONVERT (varchar,Hub.event_time,102) as dates,
        COUNT (event_time) as 'counted' FROM Hub WHERE Hub.event_time BETWEEN " +
        dateFrom + " AND " + dateTo + " + filter + " GROUP BY CONVERT
(varchar,Hub.event_time,102) ORDER BY dates"; // order by dates makes the query return in date order which allows the graph to scale correctly
}

// clear out any existing points if true to reset
if (reset == true)
{
    z1.GraphPane.CurveList.Clear();
}

GraphPane myPane = z1.GraphPane;

// configuration of chart
myPane.Title.Text = "Timeline";
myPane.XAxis.Title.Text = "Date";
myPane.YAxis.Title.Text = "Number of events";

string toolhint = "";
PointPairList list = new PointPairList();

string was = "";
double x = 0;
double y = 0;

MyConnection = new SqlConnection(connectionString);

try
{
    MyConnection.Open();
    SqlCommand myCommand = new SqlCommand(sql, MyConnection);
    SqlDataReader DbReader = myCommand.ExecuteReader();

    while (DbReader.Read())
    {
        // the following code gets the date, splits at the . into year, month and day
        // then convert to ints these figures and make a date object with them
        was = DbReader[0].ToString();
        string[] now = was.Split('.');

        DateTime dt = new DateTime(Int32.Parse(now[0]),
                                  Int32.Parse(now[1]),
                                  Int32.Parse(now[2]));

        XDate xDate = new XDate(dt);

        // set the x and y values (doubles as required by pointpairlist)
        x = xDate.XLDate;

        // get count from database, convert object to string, convert that to int and finally to double y!
        y = Int32.Parse(DbReader[1].ToString());

        // set tooltip to display when cursor is hovered over chart element (point, line or bar)
        toolhint = "Date: " + xDate.ToString("dd/MM/yyyy") + 
                   "\nNumber of Entries: " + (int)y + "\nLog Type: " + whichLog;

        list.Add(x, y, toolhint);
    }
}
catch
{
    //Handle any exception, most likely caused by an incorrect SQL
query
    MessageBox.Show("Could not display graph. If advanced query has
been used check it is in the correct format.", "Error",
    MessageBoxButtons.OK, MessageBoxIcon.Error);
}
finally
{
    //ensure database connection is not left open
    MyConnection.Close();
}

myPane.XAxis.Type = AxisType.Date;
z1.IsShowPointValues = true;

//construction of requested chart type
if (chartType == "Scatter Chart")
{
    LineItem myItem = myPane.AddCurve("", list,
    System.Drawing.Color.CornflowerBlue);

    // Don't display the line (This makes a scatter plot)
    myItem.Line.IsVisible = false;
    // Fill the symbol interior with color
    //colour items based on value
    myItem.Symbol.Fill = new Fill(colors);
    myItem.Symbol.Fill.Type = FillType.GradientByY;
    myItem.Symbol.Fill.RangeMin = 0;
    myItem.Symbol.Fill.RangeMax = Int32.Parse(redLevel);
}
else if (chartType == "Line Chart")
{
    LineItem myItem;
    if (reset == false)
    {
        myItem = myPane.AddCurve(whichLog + " " + filter, list,
        System.Drawing.Color.Orange);
    }
    else
    {
        myItem = myPane.AddCurve(whichLog + " " + filter, list,
        System.Drawing.Color.Blue);
    }

    //colour items based on value
    myItem.Symbol.Fill = new Fill(colors);
    myItem.Symbol.Fill.Type = FillType.GradientByY;
    myItem.Symbol.Fill.RangeMin = 0;
    myItem.Symbol.Fill.RangeMax = Int32.Parse(redLevel);
}
else
{
   BarItem myItem = myPane.AddBar("", list,
    System.Drawing.Color.CornflowerBlue);
//colour items based on value
myItem.Bar.Fill = new Fill(colors);
myItem.Bar.Fill.Type = FillType.GradientByY;
myItem.Bar.Fill.RangeMin = 0;
myItem.Bar.Fill.RangeMax = Int32.Parse(redLevel);
}

//clear out any default blank space around ends of x axis
myPane.XAxis.Scale.MinGrace = 0;
myPane.XAxis.Scale.MaxGrace = 0;

// Calculate the axis ranges
z1.AxisChange();

//reset the scale in case user has zoomed in before
z1.RestoreScale(myPane);
}

public void fillGrid()
{

    //This method fills the datatable with the log data requested

    SqlCommand command;

    string sql = "";

    if (whichLog == "FileLog")
    {
        sql = "SELECT convert(varchar,Hub.event_time,103) + ' ' +
        convert(varchar,Hub.event_time,8) AS "Date and Time",
        Hub.machine_name AS "Machine Name",
        Hub.user_account AS "User Account",
        FileLog.event AS "Event",
        FileLog.file_path AS "File Path" FROM Hub, FileLog WHERE
        event_time BETWEEN "" + dateFrom + " AND " + dateTo + " AND Hub.id =
        FileLog.id " + filter + " ORDER BY event_time;";
    }
    else if (whichLog == "FileProcess")
    {
        sql = "SELECT convert(varchar,Hub.event_time,103) + ' ' +
        convert(varchar,Hub.event_time,8) AS "Date and Time",
        Hub.machine_name AS "Machine Name",
        Hub.user_account AS "User Account",
        Hub.log_type AS "Log Type",
        ProcessLog.event AS "Process Event",
        ProcessLog.process_name AS "Process Name",
        FileLog.event AS "File Event",
        FileLog.file_path AS "File Path" FROM Hub FULL OUTER JOIN
        ProcessLog ON Hub.id = ProcessLog.id FULL OUTER JOIN FileLog ON Hub.id =
        FileLog.id WHERE Hub.log_type = 'FileLog' OR Hub.log_type = 'ProcessLog'
        AND Hub.event_time BETWEEN "" + dateFrom + " AND " + dateTo + " ORDER BY Hub.event_time;";
    }
    else if (whichLog == "ApplicationLog")
    {
        sql = "SELECT convert(varchar,Hub.event_time,103) + ' ' +
        convert(varchar,Hub.event_time,8) AS "Date and Time",
        Hub.machine_name AS "Machine Name",
        Hub.user_account AS "User Account",
        ApplicationLog.event_id AS "Event ID",
        ApplicationLog.event_data AS "Event Data" FROM Hub, ApplicationLog WHERE
        event_time BETWEEN "" + dateFrom + " AND " + dateTo + " AND Hub.id = ApplicationLog.id " +
        filter + " ORDER BY Hub.event_time;";
    }
    else if (whichLog == "SystemLog")
    {
        sql = "SELECT convert(varchar,Hub.event_time,103) + ' ' +
        convert(varchar,Hub.event_time,8) AS "Date and Time",
        Hub.machine_name AS "Machine Name",
        Hub.user_account AS "User Account",
        ApplicationLog.event_id AS "Event ID",
        ApplicationLog.event_data AS "Event Data" FROM Hub, ApplicationLog WHERE
        event_time BETWEEN "" + dateFrom + " AND " + dateTo + " AND Hub.id = ApplicationLog.id " +
        filter + " ORDER BY Hub.event_time;";
    }
}
"Machine Name", Hub.user_account AS "User Account", SystemLog.event_id AS "Event ID", SystemLog.event_data AS "Event Data" FROM Hub, SystemLog WHERE event_time BETWEEN "' + dateFrom + '" AND "' + dateTo + '" AND Hub.id = SystemLog.id " + filter + " ORDER BY event_time;"
)
    else if (whichLog == "SecurityLog")
    {
    }
    else if (whichLog == "ProcessLog")
    {
    }
}
else
{
    sql = "SELECT convert(varchar,Hub.event_time,103) + ' ' + convert(varchar,Hub.event_time,8) AS "Date and Time", Hub.machine_name AS "Machine Name", Hub.user_account AS "User Account", Hub.log_type AS "Log Type" FROM Hub WHERE event_time BETWEEN "' + dateFrom + '" AND "' + dateTo + '" + filter + " ORDER BY event_time;"
}

using (MyConnection = new SqlConnection(connectionString))
{
    try
    {
        command = new SqlCommand(sql, MyConnection);
        SqlDataAdapter adapter = new SqlDataAdapter();
        MyConnection.Open();
        adapter.SelectCommand = command;
        adapter.Fill(dt);
        MyConnection.Close();

        //inform user if their query returned 0 results //display useful message via a messagebox
        if (dt.Rows.Count == 0)
        {
            MessageBox.Show("There were no results for your query.\nTry making your filter options less specific.", "Warning", MessageBoxButtons.OK, MessageBoxIcon.Warning);
        }
    }
    catch
    {
// handle exception, display message to user
MessageBox.Show("Could not display results. If advanced query has been used check it is in the correct format.", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
}