EFFECTIVE FEEDBACK IN PROGRAMMING ASSIGNMENTS TO STUDENTS: A VISUAL STUDIO 2015 EXTENSION TO ALLOW BETTER FEEDBACK AND SUBMISSION TO STUDENTS.

By
Luke Mosley (100254405)
INDEPENDENT STUDIES 6CC995
In the years 2012 - 2016
Abstract

Programming assignment submission and feedback at the University of Derby for computer science students, or any students who undertake programming based assignments is currently an average system at best. Students do not always like the system for programming assignments and neither do the staff.

Coursework Manager is a new solution to this problem that would allow students to easily submit work from within the Visual Studio IDE as a plugin, and then view feedback of the work once marked, as the old system did. This would be the grade and the paragraph feedback, however with the added addition of then also allowing students to see line by line feedback of work submitted to Coursework Manager within the Visual Studio IDE.

The main aim for Coursework Manager is to simply make programming assignment management, submission, marking and feedback better for both students and staff overall compared to the current system that is in place and used within the University of Derby (Course Resources/Blackboard).
# Table of Contents

Abstract ......................................................................................................................... 1  

1. Introduction ............................................................................................................... 5  
   1.1. Project aim ...................................................................................................... 5  
   1.2. Project objectives .......................................................................................... 5  
   1.3. Project rationale ........................................................................................... 5  
   1.4. Hypothesis .................................................................................................... 6  

2. Literature/Competing product review ..................................................................... 7  
   2.1. Introduction .................................................................................................. 7  
   2.2. Effective feedback ......................................................................................... 7  
      2.2.1. Why is feedback important ..................................................................... 7  
      2.2.2. Automated marking/feedback systems: ................................................. 8  
      2.2.3. Currently used “Code” feedback methods (in other software) ............ 9  
   2.3. Existing submission and feedback systems for university assignment submissions..... 11  
      2.3.1. Current system in use at the university .................................................. 11  
      2.3.2. Commonly used submission systems (LMS)........................................ 11  
      2.3.3. TurnItIn in detail (turnitinuk.com, n.d.) ................................................. 13  
   2.4. IDE Integration for programming assignment feedback .................................... 14  
   2.5. Conclusions and summary ............................................................................ 14  
      2.5.1. Key issues ............................................................................................... 15  

3. Methodology ........................................................................................................... 16  
   3.1. Introduction .................................................................................................. 16  
   3.2. Strategy ........................................................................................................ 16  
      3.2.1. Why a Visual Studio extension? .............................................................. 16  
      3.2.2. The existing extension (Add-In) and web service .................................. 17  
      3.2.3. Finding a solution with the Visual Studio 2015 SDK/API ...................... 18  
      3.2.4. Proposed solution one (Visual Studio Sidebar “IWpfTextViewMargin”) .... 18  
      3.2.5. Proposed solution two (Visual Studio “Error List ‘Message’”) ............. 19  
      3.2.6. Proposed data storage for line by line feedback: Both proposed solutions .... 20  
   3.3. Development process ....................................................................................... 20  
   3.4. Development decisions ................................................................................... 21  
   3.5. Limitations .................................................................................................... 21  
      3.5.1 Visual Studio SDK Limitations ................................................................. 21
3.5.2 Coursework Manager (Project) Limitations ................................................. 21
3.6. Data gathering .................................................................................................. 22
  3.6.1. Student feedback survey on the "Visual Studio Line by Line Feedback and Submission Plugin" (Coursework Manager) .............................................................. 22
3.7. Data analysis .................................................................................................... 22
3.8. Conclusions ...................................................................................................... 23
4. Findings and analysis ......................................................................................... 24
  4.1. Introduction ..................................................................................................... 24
  4.2. Findings and basic analysis ........................................................................... 24
    4.2.1. Student feedback survey on the "Visual Studio Line by Line Feedback and Submission Plugin" (Coursework Manager) Results ......................................................... 24
  4.3. Further analysis ............................................................................................. 31
    4.3.1. Student feedback survey on the "Visual Studio Line by Line Feedback and Submission Plugin" (Coursework Manager) Further Analysis ..................................................... 31
5. Discussion .......................................................................................................... 35
  5.1. Introduction ................................................................................................... 35
  5.2. Discussion ...................................................................................................... 35
    5.2.1. The perspective (Student Survey) ............................................................. 35
    5.2.2. Further Work/Improvements Questions ................................................ 36
6. Conclusions and recommendations ................................................................. 39
  6.1. Conclusion ..................................................................................................... 39
  6.2. Recommendations ....................................................................................... 39
    6.2.1. Window/Form management in the Visual Studio Extension .................... 39
  6.3. Future Work/Research .................................................................................. 40
    6.3.1. Student Web Front end for assignment management and possibly assignment submission .................................................................................................................. 40
    6.3.2. Addition of a formal method of flagging up results for later discussion ...... 40
    6.3.3. Highlighting or noting line by line feedback by severity ......................... 40
7. References .......................................................................................................... 41
8. Appendices ......................................................................................................... 47
  8.1. Appendix 1: ................................................................................................. 47
  8.2 Appendix 2: .................................................................................................. 48
  8.3. Appendix 3: ................................................................................................. 49
  8.4. Appendix 4: ................................................................................................. 49
  8.5. Appendix 5: ................................................................................................. 49
8.6. Appendix 6: ................................................................. 50
8.7. Appendix 7: ................................................................. 50
8.8. Appendix 8: ................................................................. 51
8.9. Appendix 9: ................................................................. 53
1. Introduction

1.1. Project aim

The aim of the project is to design and develop a submission and feedback system. This should allow an effective, concise and more efficient feedback about programming assignments to university students at the University of Derby.

1.2. Project objectives

1. A literature review will be undertaken to find out about feedback, why it is useful and the most effective type of general feedback. The review will also be used to look into the current problems with the currently used feedback system, and why students may or may not find the current system useful or fit for purpose.

2. The literature review will also be used to look into any pre-existing solutions that exist. Comparisons will be made to see how the existing products fit in to the project, or if the new system is of any benefit.

3. A new system will be designed and developed using a partly created existing submission system (Coursework Manager), to allow students, to submit work easier. This system will also allow students to receive more valuable feedback in Programming based assignments.

4. The system will be tested to check it meets all the aims above and will measure the effectiveness of the design and develop phase of the project.

5. Feedback from students on the current system will be gathered to create an opinion and results on the potential issues of the current feedback system.

6. Feedback from students on the new system will be gathered to create data usable to see if the aims above have been met.

1.3. Project rationale

The rationale of the project is to enable programming students who hand in programming assignments to get line by line feedback on their programming assignments. This is of more value and more objective than a paragraph summary of feedback explaining issues within the code, which is the current system in place at the University of Derby.
1.4. Hypothesis

The hypothesis for this paper is that feedback in programming assignments is generally suboptimal and usually a paragraph summary of the whole project/solution. This is because of the effective use of the feedback given, is also, questionable as it is difficult to get effective and concise feedback in programming assignments from a summary paragraph box, which is the system the university currently uses. This paper hopes to prove that better and more concise feedback in programming assignments is required. It should also hopefully prove that the feedback given is more useful to students than the current system that is in place.
2. Literature/Competing product review

2.1. Introduction

In this section, the paper shall look at and discuss existing submission and feedback products and systems that are commercially available. It shall also discuss effective feedback and why this is important within any environment, but specifically an assignment feedback and programming assignment way. This section shall go into greater detail of the reasons why the hypothesis was made and point out any issues that may have not been foreseen in the programming feedback of assignments.

2.2. Effective feedback

The art of giving feedback is an important tool used to aid further personal development (Dignen, 2014). Feedback is a way of being able to self-reflect on the work you have done and allows for you to use this information to improve further and learn from your mistakes (reading.ac.uk, n.d.). It is also important that the feedback given has the correct data within it, so that, you could analyse and act on the constructive feedback to further improve or learn from your work (fullcirclefeedback.com.au, 2012; Dignen, 2014).

2.2.1. Why is feedback important

*Feedback in general:*

Feedback in general is used in a lot of real world environments and is something we are used to receiving at different stages of our life (opm.gov, n.d.; Dignen, 2014; fullcirclefeedback.com.au, 2012). For example, feedback is used in work environments, educational environments and also in day to day life so it is something we come to know and expect (Dignen, 2014; fullcirclefeedback.com.au, 2012). Giving or receiving feedback is not just something expected in an educational environment, or even a programming based assignment way. Because of this, feedback comes in many different forms such as verbal and written feedback (Race, 2001).

*Why is any form of feedback important?*

A definition of this is “Feedback is receiving other people’s ideas about what we think we have learnt.” (studymore.org.uk, n.d.). Feedback is designed to be a loop that lets you take on-board criticism, think about your work and then learn from it, so you can learn for the future (studymore.org.uk, n.d.). Feedback in the workplace is different however: the main
goal of work place based feedback is to see or set goals for people to achieve (fullcirclefeedback.com.au, 2012; opm.gov, n.d.). This is not too dissimilar to an educational setting as the purpose is near enough the same, which is to explain and learn from your mistakes to further develop yourself in the future (Dignen, 2014; Race, 2001). This is the main core of why a more granular feedback system is required for programming assignments. Without the correct information or data for exactly where you went wrong, objective self-reflection becomes difficult (Race, 2001; Dignen, 2014).

Feedback for students and why it is important:

“Feedback is one of the most powerful influences on learning and achievement, but this impact can be either positive or negative” (Hattie & Timperley, 2007). Feedback in education is important as it is an underpinning of learning (Hattie & Timperley, 2007; Race, 2001) simply because to learn you need to make mistakes and learn from them (Hattie & Timperley, 2007; Race, 2001). However only if the feedback is concise and useful/constructive can they understand the feedback and use it to further improve in the future (Hattie & Timperley, 2007; Race, 2001). This is where the paragraph feedback given for current programming assignments falls down, as it is not objective or detailed enough to show a student exactly where “X” method is incorrect or why the class “Y” is not needed. This matches with feedback needing to be objective and Goal-Referenced to be the most effective form of feedback (Wiggins, 2012). A line by line feedback solution would give more detail and allow the student to get more out of the feedback given, to further self-reflect and improve in the future (Hattie & Timperley, 2007; Race, 2001). This goes back to the core of feedback and learning and how they work very closely together to motivate and engage the student to learn (Ip, 2005). Another point to make is that feedback is also less effective the longer it takes to get feedback back on an assignment (Ip, 2005). Meaning that this may be a consideration later down the line that the design of the project may need to take into account.

2.2.2. Automated marking/feedback systems:

This is an area with a lot of research; this paper is not designed to implement an automated marking system. However what can we learn from current implementation of these systems and the feedback they give to students?
“Providing feedback on programming assignments manually is a tedious, error prone, and time-consuming task” (Gulwani, et al., 2014). Many of these systems in place are designed for answering specific programming questions which all expect a specific answer (Gulwani, et al., 2014; Nutbrown, 2012). This would not be suitable for a lot of the University of Derby’s programming assignments due to their open ended nature (Gulwani, et al., 2014; Nutbrown, 2012) and would only work effectively if everyone had or implemented the same program (Gulwani, et al., 2014; Nutbrown, 2012). So even automated marking and feedback systems are limited in the amount of feedback they can give to students (Gulwani, et al., 2014; Nutbrown, 2012). However in specific cases where the assignment being set is a question with one specific answer (e.g. implement a C# “Stack” class) this would be possible to be auto marked and feedback automatically given as there would only be a few ways of logically doing this assignment correctly (Gulwani, et al., 2014; Nutbrown, 2012). However for an open ended assignment such as “Design a web application” how each student decides to do this and what tool they use is open ended which is why this is definitely not a one size fits all solution (Gulwani, et al., 2014; Nutbrown, 2012).

Automated feedback does work depending on the style and programming assignment set (Gulwani, et al., 2014; Nutbrown, 2012). However the feedback could be more granular and detailed which is the outcome this paper wants. It is not a suitable solution though for every assignment.

The “Automated/Assisted Marking & Feedback” paper (Nutbrown, 2012) gives us an insight into how students react to more granular feedback (Nutbrown, 2012). Section 4.3.2 of the (Nutbrown, 2012) paper suggests that students do prefer more granular feedback as it allows them to “identify possible areas of improvement to their work from the feedback.” (Nutbrown, 2012). This is what this project is all about and gives the paper a good basis that students do want more valuable feedback (Nutbrown, 2012) available to them, rather than just a one paragraph summary of their submitted work with a grade (Nutbrown, 2012).

2.2.3. Currently used “Code” feedback methods (in other software)
Within the software development field as a whole, developers are now use to seeing live feedback in a line by line fashion within IDE’s (tools) such as Visual Studio (msdn.microsoft.com, 2015). In a real world working development environment this is also common place, speaking from personal development work experience.
“Live programming allows programmers to edit the code of a running program, and immediately see the effect of the code changes.” (Burckhardt, et al., 2013). Techniques like this are becoming more and more common in the software development space. Line by line feedback in the SVN and IDE space is becoming the norm (Burckhardt, et al., 2013). The feedback system the project plans to implement will not be live as it is intended to allow lecturers to give feedback on assignments. However it will still bring the benefit of giving more effective feedback to programming assignments, compared to a block of text and improving the overall student experience.

The current system in use does not handle code assignments well for effective feedback in a line by line and file by file setup (Appendix 1). The project would allow feedback that is more detailed line by line. This would also make it easier for staff to mark and give more meaningful direct feedback and improve the overall student experience.

In real world working environments, as a software developer, feedback like this is common practice, for example when a company uses source control management (subversion.apache.org, n.d.). Tools like SVN automatically give line by line feedback on what has changed since a previous check-in (tortoisesvn.net (Docs), 2015). During the check-in process, this kind of feedback in invaluable in a working environment as it allows the developer to see a potential mistake before it occurs. It also allows the developer to critically evaluate and check their work before submitting (checking-in) more effectively, because of the line by line feedback it gives the user at this stage of the process (tortoisesvn.net (Docs), 2015).

Reusing this theory and way of giving feedback line by line will make students more employable, because it will give them the skills on how to understand programming feedback. Feedback that is given to them similar to SVN (Line by Line) like in the real world work environment. Understanding feedback given like this in detail is important. This should increase feedback usefulness and the student experience overall in programming assignments if a line by line feedback method is used.
2.3. Existing submission and feedback systems for university assignment submissions

2.3.1. Current system in use at the university

The system currently used within the university is “Blackboard” (blackboard.com, 2015). However students know it internally as “Course Resources” which is part of the wider “UDo” portal branding (University Of Derby, n.d.; University Of Derby, n.d.). For most assignment submissions students are used to using “TurnItIn” (turnitin, 2015). TurnItIn is what the university uses as standard for assignment submissions (Appendix 4). “The University of Derby uses TurnitinUK, as a tool for online submission (eSubmission), online feedback and to match text in student submissions to other sources.” (derby.ac.uk/tel, n.d.).

TurnItIn works really well for written assignments (such as text based written papers) as this is what it is designed primarily for. However it is impossible to use for programming based assignments due to its explicit file types and file size requirements (Appendix 2) (turnitin, n.d.; derby.ac.uk, 2013). Zip files are also allowed to be uploaded and submitted to TurnItIn (turnitin, n.d.). But they must adhere to the explicit file types discussed above (Appendix 3) (turnitin, n.d.).

This inherently causes issues straight away as submitting, for example, a C# programming assignment. As TurnItIn would reject “.cs” (MSDN, 2008) and “.csproj” (MSDN, 2008) files as it does not accept unknown file types (turnitin, n.d.).

The current system, which is in place for programming based assignment submissions, is to use the alternative submission point (Not TurnItIn) built in to Blackboard. This allows students to upload a zip file of their program/solution to the Blackboard submission point (Appendix 5). “If you were hoping to use another file type, larger files or a multiple file submission, then you might be able to use the Course Resources Assignment tool instead.” (derby.ac.uk, 2013). As appendix 5 shows, once feedback is received after marking, the feedback comes up as a paragraph about the attached file on the right hand side. This is the current system students use when handing in programming based assignments through the in place submission and feedback system (derby.ac.uk, n.d.).

2.3.2. Commonly used submission systems (LMS)

A Learning Management System (from here on referred to as a LMS) is a usually web based application which automates and allows easier tracking of education modules and files as
well as online assignment submissions and self-service content publications (Ellis, 2009). A
LMS may not always be used in an educational setting but instead used in a business or
organisational setting to allow staff to self-train and certify on internal products or processes
(Ellis, 2009). However for this paper it shall be concentrating on the academic setting of a
LMS implementation and how it is used to aid and allow effective submission and feedback
of assignments (Ellis, 2009).

For this part of the report, the paper shall discuss the most commonly used LMS’s used by
academic institutions. It shall also discuss how these systems also work on a submission and
feedback of assignments level. This section shall also briefly discuss what each LMS is and
can do. For the list of most commonly used LMS’s the data found in Appendix 6 was used
(capterra.com, 2015). The top three LMS’s from this dataset are discussed further below
(capterra.com, 2015).

1. Edmodo: (edmodo.com, n.d.)
“Edmodo is an online networking application for teachers and students”
(edmodoteacherhub.wikispaces.com, 2011). Edmodo is more aimed at the primary and
secondary school education setting (Also known in the US as “K-12”) (edmodo.com, 2015).

Because of this Edmodo assignment submission system is very rudimentary because of the
market it is aiming for (edmodoteacherhub.wikispaces.com, 2011). An assignment once
created allows for free text to be entered or files uploaded (up to a 100mb limit) (Cauley,
2014; support.edmodo.com, 2016). However issues occur as you “cannot remove or attach
any new files to the assignment.” (Cauley, 2014) So resubmitting a programming based
assignment would be difficult. Also the Feedback given by this system would not be useful in
a programming assignment sense due to the feedback only working with certain file types to
allow annotations/feedback (Cauley, 2014). Effective feedback to students does not seem to
be this LMS’s strong point. However as it is aimed at a younger audience and simpler
feedback is more common for younger students (edmodoteacherhub.wikispaces.com, 2011).

2. Moodle: (moodle.org, n.d.)
“Moodle is a learning platform designed to provide educators, administrators and learners
with a single robust, secure and integrated system to create personalised learning
environments.” (docs.moodle.org, 2016).
The submission and feedback within Moodle when not using TurnItIn does support a form of programming assignment submission (docs.moodle.org, 2015). However it does involve using a 3rd party plugin which is a plagiarism detector is called “JPlag” (jplag.ipd.kit.edu, n.d.; docs.moodle.org, 2015). JPlag and the Moodle combination supports zip file solution uploads (plagiat.htw-berlin.de, 2012; Nutbrown, 2012; screenr.com, n.d.).

Moodle’s submission process also supports a very similar feedback system, to the system the university use (Blackboard) as discussed in section 2.3.1 and also from Edmodo. Moodle is also a feedback in a paragraph based solution (When using .zip files). The tutor writes the feedback after marking the submission on their local machine (docs.moodle.org, 2016). Line by line feedback is available but only in written assignments such as a pdf document (docs.moodle.org, 2016). Moodle seems to support programming assignment submissions well. However the feedback that is given is still only paragraph based feedback to students (docs.moodle.org, 2016).

3. Blackboard: (blackboard.com, n.d.)

Blackboard is a learning management system used to allow online resource management and assignment submissions (blackboard.com, n.d.).

The submission and feedback system for programming assignments has already been discussed in section 2.3.1 as this is the system in place at the university. Blackboard works with a simple zip file upload system however it does not have a plugin to check for plagiarism (blackboard.com, 2011). Blackboard does not also support the raw file types on their own such as “.cs” which is why .zip files are used (help.blackboard.com, n.d.). This is the only real difference as both accept zip file uploads that the student has to zip up and submit themselves. Paragraph based feedback as already discussed is also provided by blackboard to students.

2.3.3. TurnItIn in detail (turnitinuk.com, n.d.)

TurnItIn has been discussed and touched on in section 2.3.1 above however what exactly is TurnItIn?

“TurnItIn is the global leader in evaluating student work.” (turnitinuk.com, n.d.). TurnItIn allows submissions to be checked online for originality and plagiarism issues, allows online grading and allows tutors to provide useful feedback to students (turnitinuk.com, n.d.).
However the common thing here is that most UK institutions use TurnItIn for assignment submissions as standard. “Number of students under license (more than 24 million)” (turnitinuk.com, n.d.). This figure is for worldwide usage however UK uptake is also very high (turnitinuk.com, n.d.). The TurnItIn FAQ says that TurnItIn can be used across all disciplines for assignment submission however, as discussed above, this is incorrect for programming based assignments (turnitinuk.com, n.d.).

TurnItIn can integrate with LMS’s however the ones that are supported is limited (turnitinuk.com, n.d.). Out of the three discussed above in section 2.3.2 only Moodle and Blackboard are supported by TurnItIn officially (turnitinuk.com, n.d.). This makes the choice of LMS’s to use in an academic setting between these two main LMS’s, if using TurnItIn, is a major requirement of the academic institution (turnitinuk.com, n.d.).

TurnItIn cannot be used for programming based assignment submissions for the reasons discussed in section 2.3.1. TurnItIn’s feedback system does allow line by line or comment feedback but only on written assignments due to its file type limitations (turnitin, n.d.). So TurnItIn alone is not suitable for programming based assignment submission and feedback due to its limitations.

2.4. IDE Integration for programming assignment feedback

No exact replica or similar product exists from the extensive research undertaken, which closely integrates with an IDE, that would give better feedback and submission to students. The closest tool is JPlag which the paper touched on in the Moodle section. However this is different as it integrates with Moodle with a Web UI and is only really a plagiarism checking tool (jplag.ipd.kit.edu, n.d.; docs.moodle.org, 2015).

This puts this project in the unusual position that there are no real exactly matching existing products or competitors which do a similar task in any IDE.

2.5. Conclusions and summary

This project does seem feasible after the literature review undertaken, as there are improvements to be made for a more detailed feedback for programming assignments at the University of Derby (Dignen, 2014; Gulwani, et al., 2014; Hattie & Timperley, 2007; derby.ac.uk, 2013). Giving effective feedback is important in an educational setting which is why improving the feedback for programming assignments is important and feasible.
The paper “Automated/Assisted Marking & Feedback” (Nutbrown, 2012) does also prove that there is a need and desire from students on programming based assignments for better and more useful feedback (Nutbrown, 2012).

2.5.1. Key issues

*The use of TurnItIn within the university:*

TurnItIn is the preferred submission and feedback tool for most academic institutions due to its plagiarism checker (turnitin, n.d.; derby.ac.uk, 2013). However, due to the issues discussed in the review, it is not possible to use it for programming based assignments (turnitin, n.d.; derby.ac.uk, 2013). Because of this, the project does need to keep in mind that TurnItIn could implement a code/programming based submission system and the university would be likely to choose this, over any other system, due to its close integration with the current system (turnitin, n.d.; derby.ac.uk, 2013). This is not an issue at this moment in time, but is a possible future issue with the system and this project down the line.

*Giving effective feedback:*

The whole point of this project and paper is to improve the feedback given to students on programming based assignments at the University of Derby. The project needs to keep in mind that this could potentially be an issue if the feedback method is implemented incorrectly or students don’t agree and actually prefer to use the old system over the potential new feedback and submission system.

*Existing products/solutions to giving programming feedback to students:*

No existing line by line feedback Visual Studio plugin tool exists within the academic assignment feedback sector. All existing solutions such as the Moodle and plugin combination use a web UI front end to give the feedback to the students (jplag.ipd.kit.edu, n.d.; docs.moodle.org, 2015). The project needs to keep in mind that there may be a reason why no one has implemented a plug-in to an IDE yet for this solution, or simply no one has decided or tried to approach the problem this way before. Either way, it is, a potential issue and something that may be an issue down the line.
3. Methodology

3.1. Introduction

The paper so far has explored the current ways in which feedback is given to students in programming assignments at the University of Derby and the problems with it. In this section a solution shall be proposed, explored and designed to hopefully resolve the hypothesis.

3.2. Strategy

This project is based on an inherited submission system Wayne Rippin wrote and implemented in Visual Studio 2013 ("Assignment Manager"), which is an Extension and a WCF service and website solution. Wayne had started to port the Extension and service to Visual Studio 2015 and to use WebAPI/MVC. However the port was not fully complete and a part of this project has been to port across and improve on the missing logic from the 2013 version. The project has improved on and added to the 2015 version of the Extension and now WebAPI and MVC site to get it into a working and concise state. The new project is called “Coursework Manager”.

The project will use my own version of the Agile methodology with Scrum during the design and development process of the solution and project (scrummethodology.com, n.d.). The reason the project is following a Scrum-like methodology is due to me being the sole team member. This is why the project is mostly just using the Kanban Board element of the scrum process to allow easy planning of tasks and task tracking (kanbantool.com, 2016). This process is used heavily in business for project and development management and this is one of the main reasons the project is using this planning and tracking tool and methodology (kanbantool.com, 2016).

The actual implementation of scrum and Kanban the project will be using is TargetProcess (targetprocess.com, n.d.). This was chosen due to previous experience using it as well as TargetProcess’s easy to use and access web front end to the Kanban board for planning and easy task management (Appendix 7).

3.2.1. Why a Visual Studio extension?

A Visual Studio Extension and Web solution was chosen for this project partly because of an existing system already been somewhat developed and in place. However the main reason
behind a Visual Studio Extension being chosen as a basis for this project, is the fact that the University of Derby use Visual Studio extensively in teaching and for assignments whether these be C#, C++ or any other language that Visual Studio supports. A lot of assignments rely on Visual Studio solutions so this is why a Visual Studio Extension was chosen to be the best solution to the original submission problem. Visual studio is also natively installed on all lab machines that computing students use so this is another consideration and reason on why a Visual Studio Extension solution was decided upon.

3.2.2. The existing extension (Add-In) and web service

The existing Extension (Add-In) base solution used for this project, was a part-ported 2015 Visual Studio Extension solution and MVC/WebAPI service. The project has improved on this and ported across most of the missing logic from the original Visual Studio 2013 submission system Add-In/Extension as already briefly discussed in section 3.2 above.

The current project solution of the system at a high level consists of the diagram outline below:

This high level diagram of the system shall not change during this project. This is because any changes made will be at a lower level, such as additions or changes to the Database, WebAPI, MVC Web Application or the Visual Studio Extension itself.
All calls between the Visual Studio Extension and any data access will all be done through the WebAPI service exposed to the Extension. All calls to this service have to be authenticated in a similar way the Web UI is accessed, meaning there is less chance of someone guessing the service link and maliciously adding in data as they would not be authenticated in the correct way.

3.2.3. Finding a solution with the Visual Studio 2015 SDK/API

“The Visual Studio SDK helps you extend Visual Studio features or integrate new features into Visual Studio” (msdn.microsoft.com, n.d.). The SDK is designed so you can extend and create Extensions for the Visual Studio IDE. This is what the existing solution discussed above uses and is currently implemented.

The SDK documentation is at times a little confusing and examples of how to use certain parts of the SDK are very scarce. The official examples, which exist, for the SDK are very basic examples that show only a small feature at a time (github.com, n.d.). Compared to asp.net MVC examples which the web is full of, because of this doing certain things within the SDK can be very difficult and very confusing. Documentation and help available in the wider development community is basically nonexistent, as the Visual Studio SDK is a very niche SDK to use. This is the reason why two solutions have been proposed below.

3.2.4. Proposed solution one (Visual Studio Sidebar “IWpfTextViewMargin”)

The solution proposed here would solve the feedback issue by allowing feedback on a line by line basis. A “Sidebar” would be used to add functionality similar to how a breakpoint works. The only difference would be the logic behind it: as in staff members would be able to add comments similar to how you would add a breakpoint to that line. A comment box would then pop up and allow them to add their comment/feedback. To edit an existing comment they would simply click on an existing comment icon. These comments would be stored within the database for each student submission to say what comment goes with what file and on what line.

The student would then be able to see these once the work had been marked. They would do this by pressing the solution download option, which would then download and show icons on the sidebar for that assignment next to the lines where comments and feedback have been left by the tutor. An icon would be displayed allowing them to see the feedback.
The way to implement the solution would be to use an “IWpfTextViewMargin” with a “IWpfTextViewMarginProvider” as explained in the SDK documentation section “Extending Margins and Scrollbars” (msdn.microsoft.com, n.d.). However, examples or sufficient documentation of how to use this to do what is described above is non-existent. This documentation section is the only point which explains how to use the sidebar in any detail whatsoever. This makes this implementation method more difficult to accomplish however the project will try to solve the hypothesis and problem of the project this way first.

3.2.5. Proposed solution two (Visual Studio “Error List ‘Message’”)

The solution proposed here would also solve the feedback issue by allowing feedback on a line by line basis. The “Error List Window” (msdn.microsoft.com, n.d.) would be used to add “Messages” that are feedback/comments on a line in a code file. Once a message is clicked on, it would take the student to the position in the file that the feedback/comment was left. This would give the same functionality as the proposed solution above however this is a different solution to the same problem.

For a staff member to leave feedback/comment on a line, they would right click in the file, in the open solution to be marked, and press a menu option that would then allow them to add a feedback/comment on that line. Once added this would then appear in the “Error List Window” (msdn.microsoft.com, n.d.) as a message. The difference for a member of staff would be that on a message click for them, they would be taken to the position as a student would. However they also would get the Add/Edit feedback form up which would allow them to modify the feedback/comment left or delete it, if it is added in the wrong place or in error.

The way to implement this would be to use the “ErrorListProvider” (msdn.microsoft.com, n.d.) which allows access to the built in “Error List” so that you can then interact with the “Error List” such as add messages to it etc. With custom click logic events for on click of a “Task” which is what the “message” in the “Error List” is known as (msdn.microsoft.com, n.d.). There are a few examples of this method and the documentation of this part of the Visual Studio SDK is more substantial compared to solution one above. This makes this implementation easier to implement and get working consistently and reliably, to accomplish the hypothesis and problem of the project. The paper however, will try the first solution first before then moving to this solution if it turns out solution one is not possible.
3.2.6. Proposed data storage for line by line feedback: Both proposed solutions

The proposed data storage solution for line by line feedback, in both solutions above is to store the comments in a XML file. This would then be downloaded or uploaded to the service on a save or read of a grade/feedback being left. To accomplish this an XML data model will be used to create the XML file structure. Two new service end points will also be added to allow the download and upload of the new XML file to allow staff and students to submit and see feedback. The XML file will use the “XmlSerializer” class in the .net framework to handle the creation and reading of the XML feedback file (msdn.microsoft.com, n.d.).

3.3. Development process

The main development process for the project is based on the Agile methodology as discussed in section 3.2. The functionality of the project will be designed and then developed in one of the proposed solutions above. The solution chosen will depend on numerous factors which are discussed further in section 3.4 below.

For the project TargetProcess was used to help manage time and tasks that were required. The Kanban board in TargetProcess was used to visually see and manage each story and task. The use of TargetProcess allowed for easy project and time management of the development and paper stage of the project. An example screenshot of the TargetProcess Kanban board used can be seen in (Appendix 7).

The solution will rely on two testing approaches, live development testing and test cases (testing after the development).

For the live development testing, testing was carried out as the functionality was added. This was a good approach for the project and this did pick up numerous bugs with the solution. These bugs, during development, were found and fixed within the development stage because of the use of live testing.

The second testing approach was testing the finished solution. Test cases were written and the test plan executed. The test cases and statuses can be seen in (Appendix 8). This test approach also picked up on some bugs that have since been fixed. The use of both of these test approaches should help to make the finished solution bug-free and error-free.

SVN has also been used for source control management throughout the process of the design and develop stage of the solution.
3.4. Development decisions

Solution two was the solution that was chosen to be implemented in the end. This was due to the lack of documentation or useful examples for the “IWpfTextViewMargin” to be able to use it in the way that was originally planned. The reason for the lack of information too, could also be due to the way I was possibly planning to use it out of the scope it was originally intended and designed for. The main issue was there seemed to be no way, once created, to be able to input information into the sidebar on a file by file basis easily.

Solution two helped deliver the same result buy simply using the “message” option in the “Error List” window. This allows the feedback to be easily seen and also on click of the message/feedback, take you to exactly in the file where the feedback was left. This solution fits the projects aim and objectives well, so in the end this is one of the main reasons why it was chosen to be the option to go with. The proposed XML file solution also worked well during implementation. So this proposed solution was chosen to be used in the actual end solution to address the hypothesis.

3.5. Limitations

3.5.1 Visual Studio SDK Limitations

There are inherently limitations to the Visual Studio SDK as it is a niche area of a Microsoft SDK. As ASP.net for example is more widely used so has a vast array of documentation to support it. The Visual Studio SDK documentation on the other hand, at times, can be a little confusing. This is where the SDK limitations are most felt, as you have to try and adapt quickly if at first the interface or class you are trying to use does not work as expected, or as the documentation says it should. You then have to work around it or find an alternative. This is why originally the paper proposed two solutions to the hypothesis and aim of the project. This was in case the first proposal was not possible due to SDK limitations. It turned out this decision was a good one as the original proposal was indeed not possible in the context that the paper was trying to use it in.

3.5.2 Coursework Manager (Project) Limitations

Coursework Managers right click context menu to add feedback (for staff) in a file only works with "Code files"

The current version of the Coursework Manager Extension only allows you to add feedback to “Code” files such as those ending in “.cs” and “.config”. Currently it does not support
“.cshtml” files for example. This is due to the use of the “IDM_VS_CTX_CODEWIN” menu parent (msdn.microsoft.com, n.d.). However after much searching there does not seem to be a matching parent context for a “Text Editor” type. So because of this apparent SDK limitation, currently the menu item only appears for code files. This should not be a major issue as most programming feedback would generally be left in the “.cs” files anyway.

3.6. Data gathering

Data will be gathered for the project through a student survey.

Google forms will be used to create and gather the results from the survey (google.co.uk, n.d.). This also has the advantage of being able to send students a link to fill out the survey. This should hopefully make survey participation higher than an offline paper based survey.

The survey in section 1 will ask questions about the current feedback system that is in place for programming assignments at the University of Derby. Questions about the new/proposed system with screenshots explaining how the new system works will then be asked in the 2nd section of the survey.

The student survey will be anonymous and no participation data will be being collected bar the questions being asked within the survey.

3.6.1. Student feedback survey on the "Visual Studio Line by Line Feedback and Submission Plugin" (Coursework Manager)

The full question plan and survey link can be found for the student survey in (Appendix 9). This survey will be asked to students at the University of Derby, who are on a course that has or has had programming based assignments. The survey will also be a random selection from all years of study.

This will then generate data so the hypothesis can be proved or disproved to be correct or incorrect for line by line feedback.

3.7. Data analysis

The results from the student survey from Google forms will give raw survey data, and also do some basic analysis on each question by creating some result graphs. These base results given will be the basis for any data results and analysis further done on the results.
3.8. Conclusions

The chosen solution (solution two) fits the projects aim and objectives well, as now line by line feedback can be received by students and staff can now add line by line feedback for students to see. Another advantage with this is the existing functionality of showing the grade and paragraph feedback is still in there. So the line by line feedback does not take anything away, but instead adds another level of feedback granularity for students, as now the students can see exactly where they have gone wrong on the line in the file. Examples of the system in use can be seen in the survey plan for the student survey (Appendix 9).

The survey discussed above in section 3.6 and 3.7 will then be able to prove or disprove the hypothesis further. The design and development stage is now done. However the users (students) are the ones who know if the system will improve their feedback experience in programming assignments overall. Which is why the next stage now is to get their feedback and opinions to discuss further.
4. Findings and analysis

4.1. Introduction

This section of the paper will show the findings from the surveys discussed in sections 3.6. It will also discuss the analysis that has been done on the results and also show these analysed results and findings.

4.2. Findings and basic analysis

Google forms gave basic results, analysis and graphs for each question that was asked in each survey. Those results for the raw questions for each survey can be found below.

4.2.1. Student feedback survey on the "Visual Studio Line by Line Feedback and Submission Plugin" (Coursework Manager) Results

The student survey received 18 participants in total and the results of their opinions can be found below:

Section 1. Questions about the current in place submission and feedback system, for programming assignments, at the University of Derby (Course Resources)

Q1. Is the current system in place (Course Resources) for programming assignments easy to use/understand?

(18 responses)
Q2. Is the paragraph feedback given currently (Course Resources), useful and easy to understand for programming assignments?

(18 responses)

Q3a. How clear is programming assignment feedback with the current system (Course Resources) in your opinion?

(18 responses)

Q3b. For the question above: Can you identify improvements needed or areas that work well with the currently in place feedback system (Course Resources)?

A further breakdown of feedback, perhaps a way to show us good and bad points of our code.

The feedback needs to be granular. The paragraph feedback often given does not give you the detail required to understand what you did wrong.

Any feedback regarding any issues with exact positions in code would be beneficial.

Looks old fashioned, clean simple look would help this

You should be able to give feedback within the programming as it would be helpful for people who are not confident at programming so they know exactly what is being meant.
It’s hard to get specific feedback. If you have errors in your code, you want to know where exactly, if you have a lot of classes, it will be hard to pinpoint specific errors in one paragraph. The nice thing about feedback with written essays, is the person marking the work can highlight work and leave comments, this would be nice for programming assignments too.

Actual feedback on individual code would be good!

Q4a. How formal do you feel the current feedback system (Course Resources) is in terms of flagging up results to further discuss?

(18 responses)

Q4b. For the question above: Can you suggest any improvements?

We currently email lectures to discuss grades, sometimes this can take a while and isn't straight forward.

More specific feedback would be nice in order to understand what the student did wrong.

actual positions within source code which have been highlighted within a short feedback summary which needs to be improved could be beneficial

I can’t, no
Q5. Would your experience of a programming based assignment feedback, be improved, if feedback was delivered code line by code line in the IDE (Visual Studio)?

![Pie chart showing 100% response]

Q6. How would you rate your overall experience with the current submission and feedback system (Course Resources) for programming assignments?

(18 responses)

![Bar chart showing ratings]

Q7. Any suggestions for improvement on programming based assignment feedback with the current system (Course Resources)

More granular feedback or detailed feedback would be really useful!

Identification of where concepts within programming implementations met or exceeded expectations
Section 2. Questions about the new/proposed submission and feedback system, for programming assignments. (Coursework Manager)

Q1. Does the new/proposed system (Coursework Manager) look easy to use/understand?
(18 responses)

Q2. Is the new/proposed (Coursework manager) paragraph feedback and line by line feedback, useful and easy to understand for programming assignments?
(18 responses)
Q3a. How clear would programming assignment feedback be in your opinion with the new/proposed system (Coursework Manager)?

(18 responses)

Q3b. For the question above: Can you identify improvements needed or areas that work well with the new/proposed feedback and submission system (Coursework Manager)?

This method would be very clear in pinpointing areas of code that need improving.

if the marker of the assignment wishes to leave inline coding examples for areas of improvement i.e. this could have been done this way for simplified reasons, the coding could be highlighted in a different syntax colour

Nope - clean, simple and clear!

Q4a. Would having a formal method of flagging up results for discussion later be a venture to pursue?

(18 responses)
Q4b. For the question above: If yes, how strongly do you feel that this is a venture to pursue.
(17 responses)

Q5. Do you feel that your experience of programming based assignment feedback, will be improved by the code line by code line feedback available in the new/proposed solution (Coursework Manager)?

Q6. Would a traffic light based system be beneficial where lines are highlighted by either red, amber or green to specify feedback further?

Q7. Any suggestions for improvement on programming based assignment feedback with the new/proposed system (Coursework Manager)? (I.E. The traffic light system)

Anything that makes feedback easier to understand is an advantage. The paragraph feedback was really bad for some assignments!
If the traffic light feature is implemented don't be too ambitious with line marking. Only use the traffic lighting to accompany the comment/feedback marking system previously mentioned. It should be used for readability, so students can easily spot negative and positive critic linking to the feedback (Amber for general comment?)

Turnitin in general is a steaming pile of offal produced by donkeys. Any improvements would be good. By the way, how is one lecturer going to provide line by line feedback of 100+ prog 1/2 or more importantly graphics 1/2 assignments? I can't see anyone doing that.

The traffic light system, while very helpful, may be difficult to read for the colour-blind. May consider adding symbols or something for each colour?

4.3. Further analysis

Further analysis was done on the raw data from the findings section above. The analysis was done using Excel to like similar questions together to provide a graph of results. The results from this further analysis on the raw data can be found below for the student survey.

4.3.1. Student feedback survey on the "Visual Studio Line by Line Feedback and Submission Plugin" (Coursework Manager) Further Analysis

The analysis chosen to be done on certain questions here likes the results of the questions from section one with the results from question that are similar in section two.
CS1. Comparison of Q1 "Is the current system in place (Course Resources) for programming assignments easy to use/understand?" and "Does the new/proposed system (Coursework Manager) look easy to use/understand?" Questions.

Comparison of "Is the current system in place (Course Resources) for programming assignments easy to use/understand?" and "Does the new/proposed system (Coursework Manager) look easy to use/understand?" Questions.

Linear Question Scale (1-5)
1 being "Strongly Disagree" and 5 being "Strongly Agree"

- Blue: Is the current system in place (Course Resources) for programming assignments easy to use/understand?
- Green: Does the new/proposed system (Coursework Manager) look easy to use/understand?
CS2. Comparison of Q2 "Is the paragraph feedback given currently (Course Resources), useful and easy to understand for programming assignments?" and "Is the new/proposed (Coursework manager) paragraph feedback and line by line feedback, useful and easy to understand for programming assignments?" Questions.

Comparison of "Is the paragraph feedback given currently (Course Resources), useful and easy to understand for programming assignments?" and "Is the new/proposed (Coursework manager) paragraph feedback and line by line feedback, useful and easy to understand for programming assignments?"

Comparison of responses:
- **Currently (Course Resources):**
  - 1: Strongly Disagree (1 response)
  - 2: Disagree (1 response)
  - 3: Neutral (7 responses)
  - 4: Agree (3 responses)
  - 5: Strongly Agree (1 response)

- **Proposed (Coursework manager):**
  - 1: Strongly Disagree (6 responses)
  - 2: Disagree (0 responses)
  - 3: Neutral (1 response)
  - 4: Agree (8 responses)
  - 5: Strongly Agree (9 responses)

Linear Question Scale (1-5)
1 being "Strongly Disagree" and 5 being "Strongly Agree"
CS3. Comparison of Q3a "How clear is programming assignment feedback with the current system (Course Resources) in your opinion?" and "How clear would programming assignment feedback be in your opinion with the new/proposed system (Coursework Manager)?" Questions.

![Comparison of "How clear is programming assignment feedback with the current system (Course Resources) in your opinion?" and "How clear would programming assignment feedback be in your opinion with the new/proposed system (Coursework Manager)?" Questions.](image-url)

- How clear is programming assignment feedback with the current system (Course Resources) in your opinion?
- How clear would programming assignment feedback be in your opinion with the new/proposed system (Coursework Manager)?
5. Discussion

5.1. Introduction

This section of the paper will discuss the results and analysis from section 4 above. This section will also evaluate the results to see if the aims and hypothesis of the project has been met.

5.2. Discussion

5.2.1. The perspective (Student Survey)

The results of the student survey has given an interesting perspective to how the students see line by line feedback for assignments. This has also shown a very good indication of how students would react to this change and advancement if Coursework Manager was implemented at the university.

The comparison done within CS1 above in section 4.3.1 shows the students opinion on the current system in place and the proposed new system and their thought of the two side by side. The graph clearly shows that students are ok with the current system in place and it does its job however students are not 100% happy with it. In comparison the same question on the proposed system has a much higher response rate on the strongly agree side than the old system. This shows that students would find the new system easier to use and understand in comparison to the old system.

The comparison done within CS2 above in section 4.3.1 shows the students perspective of how useful and easy to understand they feel the feedback is for the current and new system. This question is not as clear cut as the question above as there is more of a spread of opinion on the current system. However even with the spread the overall opinion is still more towards the negative side of the scale for the current in place system, whereas for the proposed system with line by line feedback added the graph results speaks for themselves. The graph shows all students but one in the survey replied with the feedback and line by line feedback results for this question being on the positive side of the scale.

Comparison CS3 above in section 4.3.1 this time shows the comparison between how clear students feel the feedback is with the current and proposed system. This question and comparison for the current system had a big spread for results on the scale. However the bigger average of results fall within the negative side of the scale for the current system. The
proposed new system comparison question had all students, but one, in the survey replying with the result for this question being on the positive side of the scale. The additional part of this question for more detail also had a lot of participants saying that they would like a more granular and further breakdown of feedback in programming assignments (Section 4.2.1, Section 1 Q3b results). The counter question asking about the proposed system, then proves that this is what students want by people saying they agree that line by line feedback would make their overall feedback experience better (Section 4.2.1, Section 2 Q3b results).

Question 5 in Section 4.2.1, Section 1 & Section 2. Both these questions together show that students want and would benefit from more granular and line by line feedback as both of these questions had a 100% response of yes. These questions alone on their own show that students would like and want line by line feedback for programming assignments.

One thing however to consider here is something a student brought up in question 7 (Section 4.2.1, Section 2 Q7) the student/participant said:

“All improvements would be good. By the way, how is one lecturer going to provide line by line feedback of 100+ prog 1/2 or more importantly graphics 1/2 assignments? I can’t see anyone doing that.”

This may be true as a tutor would not have the time to give granular detail to all students. The way the feedback is used inevitably is decided upon by tutors themselves and the addition of line by line feedback is another tool in the teaching toolbox they can use to help students understand their mistakes and programming assignment feedback better.

The raw question asked and those further analysis in CS1, CS2 and CS3 prove together that the aims and hypothesis of this paper were correct in that students want and prefer more granular feedback in programming assignments. It also proves that the line by line integrated IDE solution proposed should meet this criteria well and in a student’s opinion improve their overall feedback experience in programming assignments at the university.

The results and analysis prove that students did feel that programming assignment feedback was substandard and that the effectiveness of paragraph feedback is not always as effective as it could be.

5.2.2. Further Work/Improvements Questions

When asking questions in the student survey, this opportunity was used to ask a few more questions. These were aimed at what students may want as future improvements of
Coursework Manager or what they maybe though was missing.

These questions were:

- Q4a/4b. Would having a formal method of flagging up results for discussion later be a venture to pursue?
- Q6. Would a traffic light based system be beneficial where lines are highlighted by either red, amber or green to specify feedback further?

These questions gave an interesting viewpoint of ideas and possible advancements students wanted in addition to the current iteration of Coursework manager.

Q4a/4b. Would having a formal method of flagging up results for discussion later be a venture to pursue?

The “Would having a formal method of flagging up results for discussion later be a venture to pursue?” question, gave an insight that students would definitely like this as an additional option. In either system as the results prove that 15/16 people said yes to this and would like this as a possible addition.

However the next question asked “For the question above: If yes, how strongly do you feel that this is a venture to pursue.” This gave an even bigger granular breakdown of just how much students want/think this is a good addition. The result being 11/16 students saying they agreed that this was a good venture to further pursue. This conclusion being made by saying that 1-3 is “Bad” and 4-5 being “Good” on the linear scale (Section 4.2.1, Section 2 Q4a/4b).

Q6. Would a traffic light based system be beneficial where lines are highlighted by either red, amber or green to specify feedback further?

The “Would a traffic light based system be beneficial where lines are highlighted by either red, amber or green to specify feedback further?” question gave a very interesting insight as all participants (16/16) said yes to this question. This proves that students do want even more granular feedback in programming assignments. This also shows students do actually care about their feedback and generally want to learn from any mistakes made. However the “Any suggestions for improvement on programming based assignment feedback with the new/proposed system (Coursework Manager)? (I.E. The traffic light system)” question did raise a very good point (Section 4.2.1, Section 2 Q6/7).
A survey participant said to this question:

“If the traffic light feature is implemented don’t be too ambitious with line marking. Only use the traffic lighting to accompany the comment/feedback marking system previously mentioned. It should be used for readability, so students can easily spot negative and positive critic linking to the feedback (Amber for general comment?)”

Another participant said to this question:

“The traffic light system, while very helpful, may be difficult to read for the colour-blind. May consider adding symbols or something for each colour?”

Now both these responses both raise very good points. As colour alone for those who may be colour blind may not be the perfect or ideal implementation of this proposed addition to coursework manager. The first comment also has a very good point of “It should be used for readability, so students can easily spot negative and positive critic linking to the feedback (Amber for general comment?)”. The use of the colour coding/icon system would allow students to easier see where they have gone wrong or right in feedback form.

These feedback comments show that the traffic light idea does need further research and work to make it possible/feasible. However the comments and the main question results do show this is indeed a very valid approach for further work and is something students want. This is even more granular and easier to understand feedback in programming assignments which proves that the hypothesis of the project was indeed correct.
6. Conclusions and recommendations

6.1. Conclusion

In conclusion, current feedback methods in place at the University of Derby for programming assignments are not 100% optimal. Paragraph feedback although useful in some aspects, is not alone the best approach for giving feedback to students in programming assignments. Students want more granular feedback as it would help them understand and learn from their mistakes easier. They then can see exactly where they have then gone wrong rather than having to guess with the paragraph feedback.

The line by line feedback solution proposed in this paper is a good match to meet all of the aims and objectives and the hypothesis of this paper. If implemented the proposed system from the students perspective feedback survey would please a lot of students, as they can then use the feedback to be more critical of their work and learn from mistakes further. This means that they can then see exactly where they need to improve or have gone wrong within their code without having to ask.

The proposed system may not be perfect as it only works best with C# solutions at this moment in time. C++ solutions have been tested in a very basic way and did work. In theory C++ solutions should work the same way, however without extensive testing like the C# solutions. Further testing would need to be carried out to test that C++ works as expected and to test other languages within Visual Studio. This would then definitely confirm if this is the case or not. However the main aim and purpose of this project has been proven that simply more granular and line by line feedback is what students want in programming based assignments at the University of Derby.

6.2. Recommendations

6.2.1. Window/Form management in the Visual Studio Extension

Further thought is required to be given to handling the windows better in the Extension in the future, as currently sometimes losing or accidently closing a window, can in some select circumstances break the logic flow. How best to improve this at this moment in time, with the limitations of the Visual Studio SDK is unknown. However it is something that does need to be further improved upon as it would improve the overall user experience of the
Extension. This may be something someone chooses to address while taking on further work on the Coursework Manager Solution.

6.3. Future Work/Research

6.3.1. Student Web Front end for assignment management and possibly assignment submission

Currently the Coursework management system only supports staff interactions on the web front end. Adding a student front end where they can see their grade/feedback, as well as their line by line feedback could be a good venture/avenue for further work/research/improvement within Coursework Manager.

6.3.2. Addition of a formal method of flagging up results for later discussion

The student survey question “Would having a formal method of flagging up results for discussion later be a venture to pursue?”, results have shown that this is something students want. This may be a good venture/avenue for further work/research/improvement for the Coursework Management system, as students have strongly suggested that this would be something they would be interested in.

6.3.3. Highlighting or noting line by line feedback by severity

The student survey question “Would a traffic light based system be beneficial where lines are highlighted by either red, amber or green to specify feedback further?”, results have shown that this is something students want. This could be another area for further work/research/improvement to the Coursework Management system, as students have strongly suggested this is something they want as an addition to Coursework Manager.
7. References

Available at: http://www.blackboard.com/cmspages/getfile.aspx?guid=ecee5885-b52b-4924-9ef8-70d461df3b1f
[Accessed 02 2016].

Available at: http://www.blackboard.com/learning-management-system/blackboard-learn.aspx
[Accessed 01 2016].

Available at: http://uki.blackboard.com/sites/international/globalmaster/
[Accessed 02 2016].

Available at: http://www.blackboard.com/about-us/what-we-do.aspx
[Accessed 02 2016].

Burckhardt, S. et al., 2013. *It's alive! continuous feedback in UI programming. SIGPLAN Not. 48, 6 (June 2013), 95-104..* [Online]
Available at: http://research.microsoft.com/pubs/189242/pldi097-burckhardt.pdf,
http://dx.doi.org/10.1145/2499370.2462170
[Accessed 10 2015].

Available at: http://www.capterra.com/learning-management-system-software/#infographic
[Accessed 02 2016].

Available at: http://cl.ly/UWzo/download/Edmodo.pdf
[Accessed 02 2016].

Available at:
http://www.derby.ac.uk/media/derbyacuk/contentassets/documents/learningenhancement/teluserguides/esub/More-about-how-Turnitin-is-used-at-the-UoD.pdf
[Accessed 02 2016].

derby.ac.uk, 2013. *Creating a submission point in TurnitinUK (What do I need to know?)*. [Online]
Available at:
[Accessed 02 2016].
Available at: http://www.derby.ac.uk/about/learning-enhancement/learning-
teaching/assessment-feedback/esubmission/  
[Accessed 02 2016].

Dignen, B., 2014. Five reasons why feedback may be the most important skill. [Online]  
Available at: http://www.cambridge.org/elt/blog/2014/03/five-reasons-feedback-most-important-skill/  
[Accessed 02 2016].

Available at: https://docs.moodle.org/30/en/Programming_Code_Plagiarism_Plugin#Introduction  
[Accessed 02 2016].
Available at: https://docs.moodle.org/30/en/About_Moodle  
[Accessed 02 2016].
docs.moodle.org, 2016. Assignment (Submission) types (V3.0). [Online]  
Available at: https://docs.moodle.org/30/en/Assignment_activity#AssignmentSubmission29_types  
[Accessed 02 2016].
Available at: https://docs.moodle.org/30/en/Assignment_activity#AssignmentSubmission29_types  
[Accessed 02 2016].
Available at: https://www.edmodo.com/about  
[Accessed 02 2016].
edmodo.com, n.d. Welcome to Edmodo. [Online]  
Available at: https://www.edmodo.com/  
[Accessed 02 2016].
edmodoteacherhub.wikispaces.com, 2011. What is edmodo?. [Online]  
Available at: https://edmodoteacherhub.wikispaces.com/file/view/Executive-Summary-Edmodo.pdf  
[Accessed 02 2016].
Available at: http://www.astd.org/~/media/Files/Publications/LMS_fieldguide_20091  
[Accessed 02 2016].
Available at: http://www.fullcirclefeedback.com.au/360-power-of/  
[Accessed 02 2016].
Available at: https://github.com/Microsoft/VSSDK-Extensibility-Samples  
[Accessed 02 2016].
google.co.uk, n.d. Google Forms. [Online]
Available at: https://www.google.co.uk/forms/about/
[Accessed 03 2016].

Available at: http://research.microsoft.com/en-us/um/people/sumitg/pubs/fse14.pdf,
http://dx.doi.org/10.1145/2635868.2635912
[Accessed 02 2016].

Available at: http://education.qld.gov.au/staff/development/performance/resources/reading/power-feedback.pdf
[Accessed 02 2016].

Available at: https://en-us.help.blackboard.com/Learn/9.1_2014_04/Student/040_In_Your_Course/030_File_Attachments
[Accessed 02 2016].

Available at: http://www.cdtl.nus.edu.sg/Ideas/iot407.htm
[Accessed 02 2016].

Available at: https://jplag.ipd.kit.edu/
[Accessed 02 2016].

kanbantool.com, 2016. [Online]
Available at: http://kanbantool.com/kanban-board
[Accessed 02 2016].

moodle.org, n.d. Moodle. [Online]
Available at: https://moodle.org/

Available at: https://msdn.microsoft.com/en-us/library/hcw1s69b.aspx
[Accessed 01 2016].

Available at: https://msdn.microsoft.com/en-GB/library/43f44291.aspx
[Accessed 01 2016].

Available at: https://msdn.microsoft.com/en-us/library/33df3b7a.aspx
[Accessed 03 2016].
Available at: https://msdn.microsoft.com/en-us/library/microsoft.visualstudio.shell.errorlistprovider.aspx
[Accessed 03 2016].

Available at: https://msdn.microsoft.com/en-us/library/dd885244.aspx?f=255&MSPPError=-2147217396#Anchor_2
[Accessed 02 2016].

msdn.microsoft.com, n.d. FAQ: Converting Add-ins to VSPackage Extensions. [Online]
Available at: https://msdn.microsoft.com/en-us/library/dn246938.aspx
[Accessed 03 2016].

Available at: https://msdn.microsoft.com/en-us/library/bb166441.aspx
[Accessed 02 2016].

Available at: https://msdn.microsoft.com/en-us/library/microsoft.visualstudio.shell.vsmenus.idm_vs_ctxt_codewin.aspx
[Accessed 03 2016].

msdn.microsoft.com, n.d. XmlSerializer Class (.NET Framework (current version)). [Online]
Available at: https://msdn.microsoft.com/en-us/library/system.xml.serialization.xmlserializer(v=vs.110).aspx
[Accessed 03 2016].

Available at: https://msdn.microsoft.com/en-us/library/ms173077(v=vs.90).aspx#Anchor_1
[Accessed 01 2016].

Available at: http://www.cs.nott.ac.uk/~psxsn6/citations/annual_review.pdf
[Accessed 02 2016].

Available at: https://www.opm.gov/policy-data-oversight/performance-management/performancemanagement-cycle/monitoring/feedback-is-critical-to-improving-performance/
[Accessed 02 2016].

Available at: http://plagiat.htw-berlin.de/software-en/c12-en-results/c12-02-jplag-en/
[Accessed 02 2016].

Available at: https://www.reading.ac.uk/web/FILES/EngageinFeedback/Race_using_feedback_to_help_st
udents_learn.pdf
[Accessed 02 2016].

reading.ac.uk, n.d. Why is feedback important?. [Online]
Available at: https://www.reading.ac.uk/internal/engageinfeedback/Whyisfeedbackimportant/efb-WhyIsFeedbackImportant.aspx
[Accessed 02 2016].

Available at: http://www.screenr.com/zfl8
[Accessed 02 2016].

Available at: http://scrummethodology.com/
[Accessed 02 2016].

Available at: http://www.skillsoft.com/
[Accessed 02 2016].

Available at: http://studymore.org.uk/feedback.htm
[Accessed 02 2016].

Available at: https://subversion.apache.org/
[Accessed 02 2015].

[Accessed 02 2016].

support.edmodo.com, 2016. Submit an Assignment. [Online]
Available at: https://support.edmodo.com/hc/en-us/articles/205010904
[Accessed 02 2016].

Available at: https://www.targetprocess.com/
[Accessed 02 2016].

tortoisesvn.net (Docs), 2015. Committing Your Changes To The Repository. [Online]
Available at: http://tortoisesvn.net/docs/release/TortoiseSVN_en/tsvn-dug-commit.html
[Accessed 10 2015].

tortoisesvn.net (Docs), 2015. Viewing Differences. [Online]
Available at: http://tortoisesvn.net/docs/release/TortoiseSVN_en/tsvn-dug-diff.html
[Accessed 10 2015].


University Of Derby, n.d. [Online] Available at: [https://courseresources.derby.ac.uk](https://courseresources.derby.ac.uk) [Accessed 01 2016].

University Of Derby, n.d. *UDO.* [Online] Available at: [https://udo.derby.ac.uk/](https://udo.derby.ac.uk/) [Accessed 01 2016].

8. Appendices

8.1. Appendix 1:

Example feedback given on one of my programming assignments using the current Blackboard submission system.
8.2 Appendix 2:

File Types and Size

Turnitin currently accepts the following file types for upload into an assignment:

- Microsoft Word™ (DOC and DOCX)*
- Corel WordPerfect®
- HTML
- Adobe PostScript®
- Plain text (TXT)
- Rich Text Format (RTF)
- Portable Document Format (PDF)
- OpenOffice (ODT)

The file size may not exceed 40MB. Files of larger size may be reduced in size by removal of non-text content or the instructor may be contacted to request multiple assignments to submit the document in sections.

**Note:** Text only files may not exceed 2 MB.

**Note:** PDF documents must contain text to be submitted. PDF files containing only images of text will be rejected during the upload attempt. To determine if a document contains actual text, copy and paste a section or all of the text into a plain-text editor such as Microsoft Notepad or Apple TextEdit. If no text is copied over, the selection is not actual text.

**Tip:** Users submitting scanned images of a document or an image saved as a PDF will need to use Optical Character Recognition (OCR) software to convert the image to a text document. Manual correction of the resulting document is highly recommended to fix any errors caused by the conversion software.

**Note:** Some document formats can contain multiple data types. This includes text, images, embedded information from another file, and formatting. Non-text information that is not saved directly within the document will not be included in a file upload. This includes references to a Microsoft Excel® spreadsheet included within a Microsoft Office Word document.

**Note:** Users whose files are saved in a file type that is not accepted by Turnitin will need to use a word processing program to save the file as one of the accepted types. Rich Text Format and Plain Text file types are nearly universally available in word processing software. Neither file type will support images or non-text data within the file. Plain text format does not support any formatting, and rich text format supports only limited formatting options.

**Tip:** When converting a file to a new file format, users should rename their file with a name other than that of the original file. This is suggested to prevent permanent loss of the original formatting or image content of a file due to it being overwritten.

Taken From: “Submitting a Paper” (turnitin, n.d.)
8.3. Appendix 3:

Zip File Uploads

Instructors are able to upload a zip file of papers to a Turnitin assignment. The zip file may be any size up to approximately 200MB and contain up to 1000 individual files. If the zip file exceeds either limit it will be rejected.

Instructors uploading zip files are **advised to ensure that no unacceptable file types are contained within the zip file** and to be careful of duplicate copies of the same file within the zip file. Turnitin will attempt to detect duplicate or invalid files and warn the user of any duplicate or unacceptable files.

Taken from: “Submitting Papers” (turnitin, n.d.)

8.4. Appendix 4:

![Assessment 1](image)

>> [View/Complete](link)

Example submission point from a student view on Blackboard for assignments (My student view).

8.5. Appendix 5:

![Screenshot of Blackboard programming assignment submission point with paragraph based feedback.](image)
8.6. Appendix 6:

(capterra.com, 2015)

8.7. Appendix 7:

Example of the Kanban board showing stories of what needs doing and the stages of the stories.
8.8. Appendix 8:

Test cases:

All test were pointing to the "Production" service/Web UI server unless otherwise stated.

<table>
<thead>
<tr>
<th>Test No</th>
<th>Description</th>
<th>P/F/Other</th>
<th>Reason</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Test of login connection to local test service (Dev service)</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Test of login connection to remote service on production server</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Check menu option works for Coursework manager</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Submission of assignments for students</td>
<td>F then P</td>
<td>There was a small bug on changing the state of submission for a student in the SQL on the service. This is now resolved.</td>
</tr>
<tr>
<td>5</td>
<td>Resubmission of assignments for students</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Assignment statuses for students</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Assignment statuses for staff</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Grade and paragraph feedback send for staff</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>To mark assignment open for staff</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Line by line feedback left by staff upload and send</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>Student management screen functionality (Load)</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Staff management screen functionality (Load)</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Student View Grade logic (Open Solution)</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>Student View Grade logic (Assignment grade, data and paragraph feedback shown)</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>Student View Grade logic (Line by line feedback loaded and works if there is line by line feedback. If not no line by line feedback is loaded)</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>Done button on &quot;Grade feedback screen&quot; for student closes and deletes solution.</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>Staff manage assignment screen shows correct data and statuses</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>Staff manage assignment screen grade button opens solution to be graded and the feedback screen.</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>Line</td>
<td>Description</td>
<td>Status</td>
<td></td>
</tr>
<tr>
<td>------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>--------</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>Line by line feedback menu option shows up in code files for assignments opened to be marked.</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>Line by line feedback menu option does not show up for students or in any solution that is not opened to be marked.</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>If line by line feedback exists. Click logic on feedback for staff goes to file location and loads up feedback form to edit/delete feedback.</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>If line by line feedback exists for a marked student solution. Click logic takes them to the file and location feedback was left in.</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>Service/web front end loads and login works as expected</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>Staff area loads on login of staff member to web UI (And staff menu appears)</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>Student &quot;No functionality yet&quot; area loads on login of student to Web UI</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>Staff Web UI Modules grid data is correct</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>27</td>
<td>Staff Web UI Modules grid add module adds module</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>Staff Web UI Modules grid Edit module edits the module</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>Staff Web UI Assignments grid data is correct</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>Staff Web UI Modules grid add Assignment works as expected</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>Staff Web UI Modules grid Edit assignment works as expected</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>Staff Web UI Staff management grid shows correct data</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>33</td>
<td>Staff Web UI Staff management add staff works as expected</td>
<td>F then P</td>
<td>This at first did not add a staff member however after some debugging the issue is now resolved. (SQL Query issue)</td>
</tr>
<tr>
<td>34</td>
<td>Staff Web UI Student management grid shows correct data</td>
<td>P</td>
<td>This at first did not add a student however after some debugging the issue is now resolved. (SQL Query issue)</td>
</tr>
<tr>
<td>35</td>
<td>Staff Web UI Student management add staff works as expected</td>
<td>F then P</td>
<td>This at first did not add a student however after some debugging the issue is now resolved. (SQL Query issue)</td>
</tr>
</tbody>
</table>
8.9. Appendix 9:

Student Feedback survey on the “Visual Studio Line by Line Feedback and Submission Plugin” (Coursework Manager) Question Plan.

This survey can be found at: http://goo.gl/forms/CiEuVABA5K

“Some short questions about the current, in place feedback system, and the new proposed system (Images will be shown through the survey of new and old).

Is it better, worse or indifferent to the feedback system currently in use, for programming based assignments at the University of Derby?”

Section 1. Questions about the current in place submission and feedback system, for programming assignments, at the University of Derby (Course Resources)

An example submission and feedback (Comments) for the current system in place for programming assignments (Course Resources).

Q1. Is the current system in place (Course Resources) for programming assignments easy to use/understand?

(A Linear scale, 1 being “Strongly Disagree”, 5 being “Strongly Agree”)

Q2. Is the paragraph feedback given currently (Course Resources), useful and easy to understand for programming assignments?

(A Linear scale, 1 being “Strongly Disagree”, 5 being “Strongly Agree”)

Page 53 of 57
Q3a. How clear is programming assignment feedback with the current system (Course Resources) in your opinion?
(A Linear scale, 1 being “Not Very Clear”, 5 being “Very Clear”)

Q3b. For the question above: Can you identify improvements needed or areas that work well with the currently in place feedback system (Course Resources)?
(A Free text field to allow the participant to say or identify any improvements they feel there may be.)

Q4a. How formal do you feel the current feedback system (Course Resources) is in terms of flagging up results to further discuss?
(A Linear scale, 1 being “Not Very Formal”, 5 being “Very Formal”)

Q4b. For the question above: Can you suggest any improvements?
(A Free text field to allow the participant to say or identify any improvements they feel there may be.)

Q5. Would your experience of a programming based assignment feedback, be improved, if feedback was delivered code line by code line in the IDE (Visual Studio)?
(A yes/no choice question)

Q6. How would you rate your overall experience with the current submission and feedback system (Course Resources) for programming assignments?
(A Linear scale, 1 being “Really Bad”, 5 being “Really Good”)

Q7. Any suggestions for improvement on programming based assignment feedback with the current system (Course Resources)
(A Free text field to allow the participant to say or identify any improvements they feel there may be with the current system.)

Section 2. Questions about the new/proposed submission and feedback system, for programming assignments. (Coursework Manager)

“Remember this is the new/proposed system (Coursework Manager). That is built in to Visual Studio (VS 2015 plugin) that now allows line by line feedback (As well as the existing feedback and grade methods).”
1: Example of the first screen of Coursework Manager. This is where you would submit or view grades/feedback for an assignment.

![Coursework Manager](image1)

2: This is the assignment grade and feedback screen in the plugin. (If you clicked View Grade for the Week 9 assignment above) It still shows the grade and paragraph feedback as the old system did. However it now opens the submitted solution in the background. As well as loading in line by line feedback, this is shown in the "Message" section of the error list (Bottom of the screen).

![Assignment Grade and Feedback](image2)

![Solution Explorer](image3)
3: This screenshot shows what happens if you click a feedback message in the error list. The file the feedback matches to will open and it will put the cursor in the position that the feedback was left from the tutor. (So you can see exactly where the feedback was left)

Q1. Does the new/proposed system (Coursework Manager) look easy to use/understand?
(A Linear scale, 1 being “Strongly Disagree”, 5 being “Strongly Agree”)

Q2. Is the new/proposed (Coursework manager) paragraph feedback and line by line feedback, useful and easy to understand for programming assignments?
(A Linear scale, 1 being “Strongly Disagree”, 5 being “Strongly Agree”)

Q3a. How clear would programming assignment feedback be in your opinion with the new/proposed system (Coursework Manager)?
(A Linear scale, 1 being “Not Very Clear”, 5 being “Very Clear”)

Q3b. For the question above: Can you identify improvements needed or areas that work well with the new/proposed feedback and submission system (Coursework Manager)?
(A Free text field to allow the participant to say or identify any improvements they feel there may be.)

Q4a. Would having a formal method of flagging up results for discussion later be a venture to pursue?
(A yes/no choice question)
Q4b. For the question above: If yes, how strongly do you feel that this is a venture to pursue?

(A Linear scale, 1 being “Strongly Disagree”, 5 being “Strongly Agree”)

Q5. Do you feel that your experience of programming based assignment feedback, will be improved by the code line by code line feedback available in the new/proposed solution (Coursework Manager)?

(A yes/no choice question)

Q6. Would a traffic light based system be beneficial where lines are highlighted by either red, amber or green to specify feedback further?

(A yes/no choice question)

Q7. Any suggestions for improvement on programming based assignment feedback with the new/proposed system (Coursework Manager)? (I.E. The traffic light system)

(A Free text field to allow the participant to say or identify any improvements they feel there may be.)