Electronics and Computer Science

Faculty of Physical and Applied Sciences

University of Southampton

Hannah Alice Short

01.05.18

A Graphical Programming E-Learning Platform

Employing Gamification and Software Visualisation

Project supervisor: Dr Su White

Second examiner: Dr Paul Chappell

A project report submitted for the award of

BSc Computer Science

# Abstract

Despite the growth in computing course enrolments over the past decade, negative attitudes persist towards adopting computer programming as a skill. Although it is a competence that carries multiple benefits, such as improving logical and technical abilities, many believe it is too complex and time-consuming to learn and generally exerts little appeal. It is necessary that an innovative method of integrating this skill into wider society is found as industry becomes ever more dependent on individuals possessing software development knowledge.

This project is concerned with planning, designing and implementing an e-Learning platform directed at a demographic with little or no prior programming experience. The application will focus on ensuring it is accommodating and flexible by providing tasks that can be completed by users in short daily sprints. Furthermore, the system will be centred around the concepts of software visualisation, the process of using visual tools to clearly depict program steps, and gamification, the approach of applying game elements in non-game environments with the intent to boost user engagement.

This report exhibits the described platform and starts by presenting reviews of both literature and existing platforms as well as an analysis of data gathered from potential users. These backgrounds have been used to inspire and aid the development of the application. Each stage undertaken is documented thoroughly including the planning, design, testing and evaluation of the platform.

**Contents**

[Abstract 2](#_Toc500718788)

[1 Project Goals 4](#_Toc500718789)

[2 Background and Report of Literature Search 5](#_Toc500718790)

[Literature Review 5](#_Toc500718791)

[Causes of Negligence towards a Computing Education 5](#_Toc500718792)

[Online Platform Engagement and MOOCs 5](#_Toc500718793)

[Learning Management Systems 6](#_Toc500718794)

[Application of Gamification Strategies 6](#_Toc500718795)

[Software Visualisation 7](#_Toc500718796)

[Graphical Programming 7](#_Toc500718797)

[Existing Systems 8](#_Toc500718798)

[Requirements Survey 8](#_Toc500718799)

[3 Design and Planning 9](#_Toc500718800)

[Requirements 9](#_Toc500718801)

[UML Diagrams with Use Cases 9](#_Toc500718802)

[UI Design 9](#_Toc500718803)

[Risk Assessment 9](#_Toc500718804)

[Gantt Charts 10](#_Toc500718805)

[4 Report on Technical Progress 10](#_Toc500718806)

[5 Plan of Remaining Work 11](#_Toc500718807)

[References 11](#_Toc500718808)

[Appendices 14](#_Toc500718809)

[Appendix 1: Survey Design 14](#_Toc500718810)

[Appendix 2: Requirements Elicitation 15](#_Toc500718811)

[Appendix 3: User Interface Design 15](#_Toc500718812)

[Appendix 4: Risk Assessment 16](#_Toc500718813)

[Appendix 5: Skills Audit 19](#_Toc500718814)

[Appendix 6: Initial Gantt Chart 20](#_Toc500718815)

[Appendix 7: Revised Gantt Chart (From December) 21](#_Toc500718816)

# Acknowledgments

I would like to thank my supervisor Dr. Su White for all the support and advice she has given throughout the course of the project. Additionally, I would like to thank my second examiner, Dr. Paul Chappell for his input.

Further thanks go to the participants of my requirements survey and the subset who participated in my user testing. Their input was useful when making key decisions in my project.

# Project Goals

The overall goal of this project is to create an engaging e-Learning platform that provides a basic knowledge of programming to beginners or simply individuals who are looking to reinforce their exiting knowledge. The sub-goals of this project are as follows:

1. **Ensure the platform accentuates software visualisation**

Graphical representations will be used to illustrate concepts with the aim of minimising theoretical information administered to the user

1. **Software clearly conveys fundamental programming and algorithmic concepts**

The platform will endeavour to deliver content in two domains. Firstly, programming concepts including formatting, naming convention and decomposition as well as algorithmic concepts such as looping, conditions, data structures and searching and sorting algorithms. This is so well-rounded knowledge can be provided and ensure good practise is taught.

1. **Deploy gamification strategies to drive participant motivation**

Another factor that will be incorporated to boost user engagement will be gamification techniques such as rewards and milestones. These extrinsic values are likely to increase motivation

1. **Provide suitable guidance before allowing users to attempt tasks**

It is important to allow users to have freedom to answer questions and try techniques for themselves after having followed tutorials, so they can consolidate their knowledge and understanding. The sufficient amount of guidance given will be determined depending on the difficulty of the concept being taught

# Background and Report of Literature Search

Although computer programming is widely recognised as a difficult competence to attain, it is one that is proven to be invaluable (Fesakis and Serafeim, 2009) and is accompanied by an ever-growing demand. The problem faced in the computing domain is discovering a way in which to advertise programming as an attractive skill to learn while teaching it credibly and conveying it in a simple manner.

## Literature Review

### Computing Education: Negative Perspectives

Despite the growing need for computer science and software engineering skills in many of today’s industries, a negative stereotype is still projected onto programming education. This stereotype forms the basis of why non-programmers feel unable to interact with code (Charters et al., 2014) and thus expand their logical skill set.

Rogerson and Scott (2010) have investigated students whom have reported fearing learning to program and have concluded the main reasons behind this are the complex thinking process needed, the inability to adopt a reflective manner when working and the lack of personalisation in this sort of educational environment. For this reason, introductory programming courses have lead learners to face difficulties and deficiencies resulting in high dropout rates (Robins, Rountree and Rountree, 2003).

Another factor that can be considered to contribute to these negative attitudes is the prominent gender imbalance in the technology sector. The proportion of women occupying technical computing jobs is low and appears unlikely to change due to cultural stereotypes (Abbate, 2012).

### Online Platform Engagement and MOOCs

Although the emergence of numerous online classroom environments over the last decade means it is now accessible and convenient to take up academic courses and expand skill portfolios, there is a fear that this has led to distanced learning and such environments lack interaction (McBrien, Jones and Cheng, 2009). Despite this concern, there have been results showing the positive correlation between learning on the web and engagement levels. Students enrolling on online courses are more likely to possess a higher level of practical competence and stronger personal growth. (Chen, Lambert and Guidry, 2009).

Massive Open Online Courses (MOOCs) enable people from a variety of backgrounds to freely build upon their education at their own convenience. It is thought that the reason behind MOOC user engagement rating so highly is due to the student’s freedom to only pursue areas that reflect upon their personal interests and goals. Also, it has been noted that overall engagement is higher yet still when peers enrol together due to the increased social factor (Barak, Watted and Haick, 2016)

### Learning Management Systems

As well as focusing on the content that is presented on e-Learning sites, the learning design and sequencing of activities should be reviewed. This aspect is covered by learning management systems, these are virtual environments that can include assignment management, discussion forums and sharing of files. Speculation exists that they are not widely used among student populations due to alternative social software tools being favoured (Dalsgaard, 2006). Researchers deem LMSs to be the initial focal point when creating online platforms and courses as they can act as a basis in the delivery of instructions and participant outcomes. (Cavus, Uzunboylu and Ibrahim, 2007)

### Application of Gamification Strategies

Gamification can be defined as ‘the use of game design elements in non-game contexts’ (Deterding et al., 2011, p.12). By developing an active learning environment for a student, their aspiration to be a successful will be increased from new found engagement (Dicheva et al., 2014).

Pedreira et al. (2015) have evaluated a number of applications incorporating gamification and have identified that the following gamification elements should be implemented into such an application to make it effective: point-based systems, dashboards, levels and rewards. The increased use of gamification in e-Learning for higher education will not only raise levels of student satisfaction and motivation but also allow for a personalised learning environment to be created (Urh et al., 2015)

Christopher Su (2016) compares the classroom environment to that of an escape room in the sense that a student is forced to recall information, communicate ideas and work under pressure. This notion contributes to the structural overlap of games and education, the basis of gamification. He claims that when creating a platform utilising gamification, you should set initial content before game design methods are laid over the top.

Gamification guru Yu-Kai Chou (2014) has developed a framework, branded Octalysis. He argues that gamification should not be based on game design elements but rather the following eight core drives:

1. Meaning
2. Accomplishment
3. Empowerment
4. Ownership
5. Social Influence
6. Scarcity
7. Unpredictability
8. Avoidance.

He allocated each of these to an edge in the octagonal framework (shown below) and explains these can be divided into two groups in either one of two ways. Firstly, excluding meaning and avoidance, the octagon can be split vertically with drives representing extrinsic tendencies on the left and intrinsic tendencies on the right:

|  |  |
| --- | --- |
| **Extrinsic Tendencies** – Executing Drives for a Purpose & Goal | **Intrinsic Tendencies** – Executing drives for enjoyment and to exercise creativity |
| Accomplishment | Empowerment |
| Ownership | Social Influence |
| Scarcity | Unpredictability |

The other way they can be grouped is by excluding ownership and social influence and splitting the octagon horizontally with ‘white hat’ gamification techniques on top and ‘black hat’ gamification techniques on the bottom. White hat techniques are positive and give a sense of growth, improvement, creativity and control while ‘black hat’ techniques are not so sustainable, a user will just execute an action to see what the outcome is without the belief it can deliver achievement:

|  |  |
| --- | --- |
| **Top Core Drives – White Hat Gamification** | **Bottom Core Drives – Black Hat Gamification** |
| Accomplishment | Scarcity |
| Meaning | Avoidance |
| Empowerment | Unpredictability |

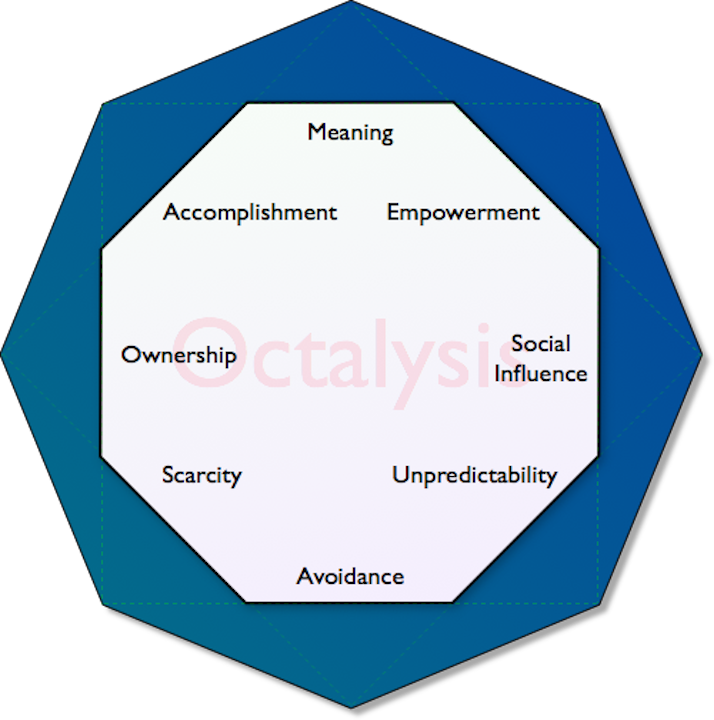


Figure 1: Yu-Kai Chou’s Gamification Framework: Octalysis[[1]](#footnote-1)

### Software Visualisation

A comprehensive description of software visualisation can be given as ‘the utilisation of graphical representations in form of text, pictures and animations (Olsson, Mozellus and Collin, 2015, p. 442). Software visualisation can accelerate programming pedagogy, enabling learners to obtain knowledge without the obligation to process copious amounts of theoretical information. Pacione, Roper and Wood (2004) have highlighted the need for visual programming in object-oriented languages due to the complexity of the interactions made in the code.

Olsson, Mozellus and Collin (2015) conducted a study in which they selected two techniques that were considered difficult for novice programmers to digest: for loops in an imperative language and the concepts of instantiation, inheritance and method calls in an object-oriented language. First year undergrads were divided, and each allocated to a technique, they were then divided again into control and experimental groups. The control groups were given a theoretical lecture explaining their concept whilst the lectures given to the experimental groups were reinforced with software visualisation. The results from this study found that 68% of the experimental group and 39% of the control group found that how they understood the dynamics of the technique were helped while 85% of the experimental group and 62% of the control group maintained a consistent understanding of where the execution of the code was focussed. These results show the scale of the impact software visualisation can have in conveying seemingly complex ideas.

Petre and Blackwell (1999) concluded that mental imagery plays a key role in program design, expert programmers make use of mental images in to produce a simulation of program behaviour. This was also true when it came to software design, programmers would envisage structures and layouts in their mind before attempting to give a description or recording notes relating to what they were designing.

### Graphical Programming

By using pictorial and graphical representations, a substantial level of information can be communicated, however this comes with the responsibility of recognising the way in which a taught concept, such as the execution of a program, will be perceived (Petre, 1995). Educating an individual via the means of graphical representations can offer an attractive alternative to simple text-based channel teaching methods.

Carlisle et al. (2005) designed RAPTOR, ‘the Rapid Algorithmic Prototyping Tool for Ordered Reasoning’ as they believed that when students were required to understand the flow of an algorithm, their attention was diverted instead to the complex syntax of the language. Thus, compromising the development of their problem-solving skills and hence showing the benefits of displaying a program graphically to direct focus onto solely the functionality of algorithms.

## Existing Systems

I have explored a variety of already available e-Learning platforms and addressed their features to gauge what is most effective in this type of an environment.

|  |  |  |
| --- | --- | --- |
| **Name** | **URL** | **Description** |
| Khan Academy | www.khanacademy.org | Khan Academy is an organisation that provides short lecture videos to supplement material taught in schools and colleges |
| **Advantages:** Videos provide step-by-step workings to illustrate concepts  **Disadvantages:** Lack of interactive elements – user’s progress/knowledge is not tested | | |
| W3Schools | www.w3schools.com | W3Schools is a site displaying openly available content to teach an array of programming languages and web technologies |
| Advantages: Tutorials are greatly detailed and are high in clarity. Many skills are available to learn.  Disadvantages: Lack of interactive elements – there is only readable content | | |
| Code Combat | codecombat.com | A web-based game requiring players to write code to be able level up |
| Advantages: Uses gamification techniques to promote programming as an appealing skill to attain. Highly interactive.  Disadvantages: Solely aimed at a child demographic | | |
| Duolingo | www.duolingo.com |  |
| Advantages: A platform that could be easily mirrored for programming languages  Disadvantages: A platform for leaning modern foreign languages rather than programming languages | | |
| Code Academy | www.codeacademy.com |  |
| Advantages:  Disadvantages: | | |

## Requirements Survey

I created an online survey to elicit opinions and attitudes towards learning to program as well as suggestions for the proposed application. My survey is composed of ten questions including Likert scale response questions and long answer response questions for suggestions. The full survey with questions and answers can be seen in Appendix 1. The design of this survey was influenced by John Creswell (2013), a professor specialising in educational research design.

From the 20 participants I gathered data from, 12 said they viewed programming to be ‘complicated’, ‘complex’ or ‘difficult’, although many of these also noted it to be ‘a useful skill to have’, ‘becoming ever-more relevant nowadays’, ‘the future of everything’ as well as ‘a good skill to have for your CV’.

# Planning

## Gantt Charts

## User Requirements

## Risk Assessment

It is important to note that my design and planning work still remains in its initial stage, this is due to choosing to undertake my requirements survey over the Christmas period which is likely to influence both the design and structure of my system. After I have collected the data, I will be able to add to and refine my existing diagrams and wireframes as well as develop more realistic use cases.

## Requirements

I have drawn up some initial requirements that the system will need to fulfil in order to be successful, once I have gathered responses from potential users in my survey, I will alter and add to these so they become more realistic and are representative of what users would like to see in such a platform.

## UML Diagrams with Use Cases

I have created two prospective use cases for my system along with two simple use case diagrams to accompany them. The first one is based on a user opening the application and having the choice of options on the home screen and the second is a user on the application looking to increase his/her points by completing the ‘while loops in Python’ skill.

## UI Design

I have preliminarily designed two of the main screens involved in the platform as basic wireframes in Balsamiq (Appendix 3), the first I have designed is how the home screen will be expected to look, this is where users will be able to select options, minimally view their progress as well as access tutorials and corresponding quizzes to level up on the platform. The second is how a tutorial screen will be expected to look, program code will be stepped through and a user can opt to go back to a previous step or skip to the next depending on how they find the difficulty of the content.

## Risk Assessment

I have produced a risk assessment (Appendix 4) to outline potential risks that could occur over the duration of my project. For each, I have estimated using a numbered scale the probability they will occur and the severity of the risk, from this I have calculated the product representing to what extent they would affect the success of the project if they were to occur. Mitigation and management methods have also been noted for each so there are recognised means of prevention as well as plans in place in the event of the risk arising.

## Gantt Charts

To estimate the timings of tasks included within my project and to track the progress being made, I have used Gantt charts. Before developing the project brief, I made an initial Gantt chart (Appendix 6), highlighting all the tasks that needed to be achieved across the scope of the project and any intermediate deadlines. Upon writing my progress report, I have created a second Gantt chart (Appendix 7) where I have adjusted the timings of these tasks and displayed the progress that has already been made.

# Design

Unfortunately, I have not made as much progress as I would have liked to have made up to this point in the project schedule. I believe the main cause of this is taking other modules this semester that are particularly heavy with coursework and not accounting for this completely when initially allocating time to the project. Next semester, the project will be taking up two module spaces and the additional modules I have chosen include little coursework in their assessment so I should in theory have much more time to allocate to the remaining work.

Thus far I have completed the majority of my literature review, attempting to read from all disciplines that my project is likely to cover although as the process progresses, it is likely that I will read some additional literature to gain a more concrete background in areas such as graphical programming.

After this, I began looking into and reviewing existing systems in order to evaluate what worked well and what could be changed for improvement in similar e-Learning platforms as well as producing my requirements survey in order to gather perspectives from individuals who have no or little experience in the computing domain.

I underestimated the time it would take to create my survey and complete an application for ethical approval and therefore this process has been pushed backwards with my data collection taking place over the Christmas break, which can be seen in my revised Gantt chart (Appendix 7).

I have created some initial simple UML diagrams and UI wireframes to form a basis for the design of my system although these will be enhanced and updated once my data has been analysed.

# Plan of Remaining Work

I have opted to work over the Christmas period although the tasks I will be working on will be completed at a much less intensive rate. From my Gantt chart detailing the work I am to do from December until the deadline of the project (Appendix 7), it is shown that I plan to conduct my survey and data collection as well as finish the last few bits of the design and planning section so that upon returning, I am able to start the development phase of the project.

I have also opted to work over the exam period but again this will be leisurely due to having to accommodate for revision. I have made this decision as I was not content with the idea of dropping the project for a period of almost a month. My development stage will start during this time whereby I will begin developing the interfaces of my system. Hopefully the end of this stage of implementation will tie in with the end of the exam period and I can then start working on the back-end features such as handling question data and developing a point scoring system. I will be developing my platform in Java as this is the language I am most familiar with and will be using SQL for keeping a store of application questions and responses. My software will be implemented using the MVC framework in order to have clean separation between my interfaces, data store, backend and logical components.

I plan to start the testing and evaluation phases a few weeks before the Easter break in order to dedicate the four weeks we have off to writing up my final report so it is ready for submission upon returning.

# References

Abbate, J. (2012) *Recoding Gender: Women’s Changing Participation in Computing*, Cambridge, MIT Press

Charters, P. Lee, M. J. Ko, A. J. Loksa, D. (2014) Challenging stereotypes and changing attitudes: the effect of a brief programming encounter on adults' attitudes toward programming, *Proceedings of the 45th ACM technical symposium on Computer science education – SIGCSE ’14*, pp.653-658, Available from: https://dl.acm.org/citation.cfm?id=2538938 [Accessed 8th December 2017]

Chou, Y. (2014, February 28) Gamification to improve our world: Yu-kai Chou at TEDxLausanne [Video file]. Retrieved from http://yukaichou.com/chou-musings/tedx-gamification-change-world/

Creswell, J. (2013) *Research Design: Qualitative, Quantitative, and Mixed Methods Approaches*, SAGE Publications Inc.

Deterding, S. Khaled, R. Nacke, L. Dixon, D. (2011) Gamification: toward a definition, *Chi 2011*, pp. 12-15, Available from: http://gamification-research.org/wp-content/uploads/2011/04/02-Deterding-Khaled-Nacke-Dixon.pdf [Accessed 6th December 2017]

Dicheva, D. Irwin, K. Dichev, C. Talasila, S (2014) A course gamification platform supporting student motivation and engagement, *2014 International Conference on Web and Open Access to Learning, ICWOAL 2014*, Available from: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=7009214 [Accessed 9th December 2017]

Fesakis, G. Serafeim, K. (2009) Influence of the familiarization with "scratch" on future teachers' opinions and attitudes about programming and ICT in education, *Proceedings of the 14th annual ACM SIGCSE conference on Innovation and technology in computer science education – ITiCSE ’09*, Vol. 41, Issue: 3, pp.258-262, Available from: https://dl.acm.org/citation.cfm?id=1562957 [Accessed 8th December 2017]

Olsson, M. Mozellus, P. Collin, J. (2015) Visualisation & Gamification of e-Learning & Programming Education, *The Electronic Journal of e-Learning*, Vol. 13, Issue: 6, pp. 441-454, Available from www.ejel.org [Accessed 21st September 2017]

Pacione, M. Roper, M. Wood, M. (2004) A novel software visualisation model to support software comprehension, *Proceedings - Working Conference on Reverse Engineering, WCRE*, pp. 70-79, Available from: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=1374307 [Accessed 21st September 2017]

Pedreira, O. Garcia, F. Brisaboa, N, Piattini, M. (2014) Gamification in software engineering – A systematic mapping, *Information and Software Technology*, Vol. 57, pp. 157-168, Available from: https://ac.els-cdn.com/S0950584914001980/ [Accessed 14th October 2017]

Petre, M. (1995) Why looking isn't always seeing: readership skills and graphical programming, Communications of the ACM, Vol. 38, Issue: 6, pp. 33-44, Available from: http://delivery.acm.org/10.1145/210000/203251/ [Accessed 8th December 2017]

Petre, M. Blackwell, A. F. (1999) Mental imagery in program design and visual programming, *International Journal of Human-Computer Studies*, Vol. 51, pp. 7-30, Available from: https://ac.els-cdn.com [Accessed 8th December 2017]

Robins, A. Rountree, J. Rountree, N. (2003) Learning and Teaching Programming: A Review and Discussion, *Computer Science Education*, Vol. 13, Issue: 2, pp. 137-172, Available from: https://doi.org/10.1076/csed.13.2.137.14200 [Accessed 8th December 2017]

Rogerson, C. Scott, E. (2010) The Fear Factor : How It Affects Students Learning to Program in a Tertiary Environment, *Journal of Information Technology Education*, Vol. 9, pp.147-171, Available from: https://www.learntechlib.org/p/111361/ [Accessed 6th December 2017]

See, C. (2017, April 24) Gamification in Higher Education - TEDxCUHK [Video file]. Retrieved from https://www.youtube.com/watch?v=d8s3kZz1yQ4

Urh, M. Vukovic, G. Jereb, E. Pintar, R. (2015) The Model for Introduction of Gamification into E-learning in Higher Education, *Procedia - Social and Behavioral Sciences*, Vol. 197, pp. 388-397, Available from: https://ac.els-cdn.com/S1877042815041555/ [Accessed 7th December 2017]

# Appendices

## Appendix 1: Survey Questions and Responses

**Section 1: Opinions Relating to Programming Education**

1. What are your initial thoughts on/how do you view the notion of ‘programming’?

* Challenging, Complicated, Difficult, Mathematical
* Writing a language which is understood by the computer
* An interesting prospect perhaps more of a hobby than for work purposes
* Complex sequences of numbers and letters to programme software/hardware
* I have no idea what that is
* The creation of software to complete tasks and the connection of these programs to one another into larger and more complex systems.
* Useful skill to have but can be either very interesting or incredibly boring
* Making stuff work on the computer by writing words
* I have a relatively underdeveloped understanding of the topic but being around programmers in my family and friends has given me a realistic scope of the topic. It’s largely laborious work that requires the ability to have an overview of what you want done before you start.
* Think it’s complicated
* Complex and difficult
* Very abstract and nerdy although enough people seem to be able to do it that I don’t think it’s THAT difficult. Becoming ever-more relevant nowadays but not for everyone.
* The future of everything pretty much
* Fun and challenging
* Very complex and hard to understand.
* Complex takes a lot of practice
* It’s very complexed. However, can be very useful in modern society.
* It is too complicated for a mere mortal like me to comprehend
* I think it’s a good skill to have for your CV it’s one of the only ‘hard skills’ that is accessible to everyone. Downside is that programming covers such a broad range that it’s hard to keep enthusiasm. For instance, being good at HTML doesn’t make you any better at Ruby or Python. Programming is too broad.
* Coding takes a lot of time to implement can be very difficult

1. Have you ever considered the prospect of learning to program? If your answer is no, why?
2. How complex do you consider programming to be?
3. Approximately how much time would you imagine it takes to acquire a good basic knowledge of programming? (In the case of regular learning)
4. How do you believe programming can benefit you? Please select all that apply

**Section 2: Analysis of Proposed Application**

1. If learning to program was a task you could carry out in small flexible sprints at your own leisure would you feel more inclined to engage with programming?
2. How much easier do you find visual learning as opposed to learning through theory?
3. How much more likely would you be to use an app if it incorporated gamification? Gamification is the application of game-design elements and game principles in non-game contexts. (Wikipedia). Such elements include prizes and earning points in competition with other players
4. How aware are you of e-Learning platforms that exist? Examples include DuoLingo TedEd and Khan Academy
5. If you had the opportunity what features would you add to an e-Learning platform designed to aid you in learning basic programming skills?

* To show how little pieces of programming can be put together to form a larger application or game.
* An online instructor to guide through the learning process
* Perhaps a ‘sandbox’ mode to allow people to use all aspects of programming.
* Add context to everything that was learnt so you could get a sense of making progress
* Without using such a platform, I could not be sure.
* None
* Smartphone usage is vital
* Present it in simple steps and start without any previous knowledge required or ask how much previous knowledge you have before it starts.
* Different areas to learn different languages purely like how Duolingo has different areas for different languages. Ability to order the language selection page by front end/back end development for apps for games on certain devices etc.
* Not sure
* Not sure
* Make it relevant to real-life i.e. I could go into a job interview and probably do something useful with programming.
* Explaining the different types of programming languages and their uses
* Lots of pictures
* Easy to start and to return to at a later time. Interactive exercises. Uses videos rather than long sections of text. |
* Levels of difficulty clear step by step points
* Nothing
* Talking animals
* Video Lessons, Commentary, Platform for collaboration with other users
* Tasks that aid understanding challenges recaps at the end of each topic

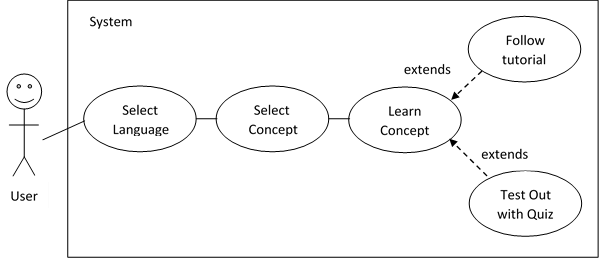
## Appendix 2: Requirements Elicitation

1. Users must be able to view their statistics and progress
2. Users must be able to easily forward and rewind during tutorials
3. Users must be given unlimited time to complete test quizzes
4. Users must be able to configure settings such as colour schemes and resetting progress
5. The tutorial content should maximise the user’s confidence when testing out of a skill
6. User data must be stored if they choose to exit the application
7. Users must be able to test straight out of a skill if they feel they can do so
8. Users must receive points from completing questions and assessments
9. Users must be able to improve their level based on the number of points they’ve earned
10. The graphics used should give the user a clear depiction of code execution

## Appendix 3: Use Case Diagrams

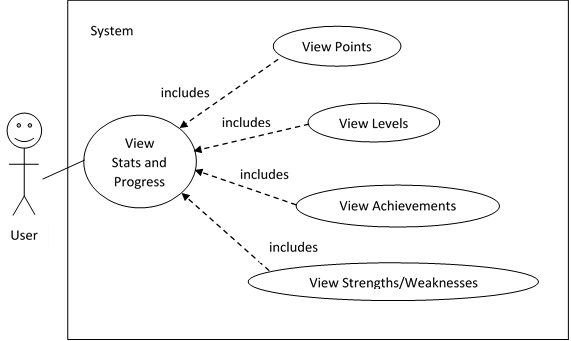
User wants to learn the concept of inheritance in Java:

* User opens home screen of platform
* User selects language as Java
* User selects skill as object orientation
* User views object orientation skill page
* User accesses tutorial on page
* User skips to inheritance section of tutorial
* User watches a graphical visualisation of inheritance with objects



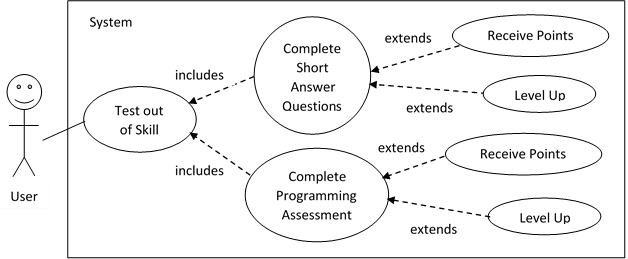
User wants to see the number of points they need to level up

* From home screen, user selects ‘View Stats’ button
* Stats for user are loaded and their current points are shown
* From progress bar, user can see how much more progress they need to make
* User starts to look for quizzes they can take to get to the next level



User wants to complete ‘While loops in Python’ in order to reach the next level

* After completing relevant tutorial, user selects ‘take quiz’ option
* User completes a set of short response questions
* User completes a short programming assessment
* User clicks finish
* Points from quiz are calculated
* If user has gained enough, they will receive a message saying they have levelled up
* Stats page will be updated to show new level

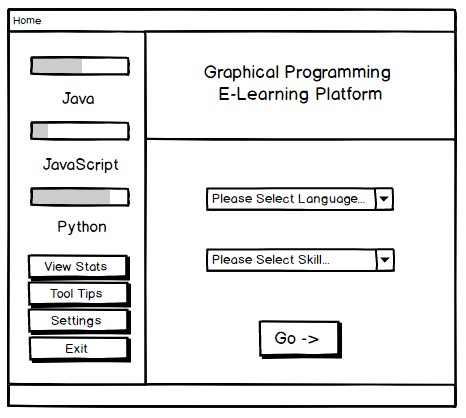


## Appendix 4: Sequence Diagram

## Appendix 5: Class Diagrams

## Appendix 6: User Interface Design

### 6.1 Home Screen



### 6.2 Tutorial Screen

## Appendix 7: Risk Assessment

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Risk** | **Probability (1-5)** | **Severity (1-5)** | **Exposure (P\*S)** | **Mitigation and Management Methods** |
| Hardware Failure | 1 | 3 | **3** | I will be working predominantly on my laptop and desktop for the duration of this project. I will ensure they are protected and will not be running any unnecessary software.  In the case of hardware failing, I will keep backups of all my project work using cloud-based solutions and external drives if required |
| Software Failure | 1 | 2 | **2** | I will be using software that I know is reliable such as IntelliJ for Java implementation and Balsamiq for drawing up wireframes.  Should any of the software I am utilising fail, I will endeavour to use an alternative or use the same software on a different machine. |
| Data and Documentation Loss | 1 | 5 | **5** | All the work I produce will be stored either in the cloud or on a drive. My code and question store will all be hosted in a GitHub repository; my designs will be stored on cloud-based applications like Balsamiq and my documentation will be regularly saved on my laptop and desktop as well as backed up on an external drive which will be kept in an alternative location to the main hardware I am using.   In the unlikely event of losing such data, I will attempt to access previous versions and if this is not possible, adjust my plan to accommodate for re-doing the affected piece of work |
| Minimal Participation in User Background and User Feedback Stages | 4 | 3 | **12** | The survey that I will be using to reinforce my background research will be shared on social media and through word of mouth to reach a wide audience and elicit the required number of participants. Regarding the feedback stage, I will take a smaller proportion of the initial participants meaning if somebody is not available, I can ask the next available candidate.  However, if I find that I have a lack of participation, I will personally approach individuals to make certain they are aware of the study. Failing this, I will have a smaller sample of users in my project. |
| Sickness | 2 | 3 | **6** | Throughout the duration of the project, I will attempt to have a good daily routine and be mentally prepared for the forthcoming work.  If I am marginally sick, I will try to carry on with the project and recognise that it will take longer to complete tasks. In extreme circumstances, I will contact my supervisor, tutor and ECS senior tutor to make them aware of my situation. |
| Supervisor Sickness | 2 | 2 | **4** | This is a risk that is out of my control but to ensure that exposure is as small as it can be, I will make sure I approach my supervisor when needed as early as possible. Alternatively, I can contact the COMP3200 module leader for project queries. |
| Failing to Meet Project Deadline | 2 | 5 | **10** | I will plan my time as efficiently as I can across the project period. I will also over-allocate time in the case of something going wrong so I have ample time to rectify this.  If I find that I am struggling to complete all the work I had set out to do coming up to the deadline, I will reduce the number of features in my application in order to have enough time to complete the report, this can then be evaluated in the report. |
| Struggling with Technical Aspects | 2 | 3 | **6** | I have conducted a skills audit to see where my technical weaknesses lie, I will allocate more time to these aspects as it is likely I will have to learn skills alongside carrying out the work  If I find a technical aspect such as programming in a particular language too difficult, I will ask a peer on the course, seek support from the programming help desk or look for an alternative means of completing the task, i.e. using a different language |
| Struggling with Report Writing | 3 | 3 | **9** | Likewise, with technical aspects, report writing has been addressed in my skills audit, I have noted that it is not one of my stronger skills and I will therefore allocate a lot of time to writing the report.   If I am struggling a great deal, I will ask peers for advice in structuring the report and ask people if they will proof-read what I have written to check for spelling and grammar errors |

## Appendix 8: Skills Audit



## Appendix 9: Initial Gantt Chart



## Appendix 10: Revised Gantt Chart (From December)

Make a table to go here pls, ta

## Appendix 11: Algorithm Code

## Appendix 12: Unit Testing

## Appendix 13: User Testing

## Appendix 14: UI Testing

## Appendix 15: Scenario Testing

## Appendix 16: Regression Testing

## Appendix 17: Project Brief

Name: Hannah Short

Student ID: 27758567

Supervisor: Dr Su White

**A Graphical Programming E-Learning Platform Employing Gamification and Software Visualisation**

**Problem:**

Many of those with no background in the computing field can look unfavourably upon the practise of learning to program and misunderstand the definition of programming leading them to believe it only caters for a limited demographic. If the pedagogy of programming education is to be improved and accelerated, whether within schools or wider society, it is paramount that language features and fundamental logic and algorithmic concepts are taught in a practical environment rather than via traditional theoretical methods. Also, that the incentive to learn how to proficiently code is more meaningful.

**Goals:**

The main focus of this project is to create a platform with an emphasis on software visualisation that is able to clearly display underlying programming and algorithmic concepts using graphical representations in a game-based environment. The platform should guide beginners seamlessly through example code executions as well as educate on basic language syntax before they are able to complete exercises assessing their understanding of visuals. Gamification principles will be used in order to boost engagement and give users a more convincing incentive to participate. The ability to compete against friends as well as earn points and rewards will make users feel there exists an alternative purpose to just simply having self-taught a skill.

**Scope:**

* Quantity of content that is to be included - there will need to be enough in order to clearly demonstrate the capabilities of the system but time constraints will have to be taken into account in order to know what can feasibly be achieved over the project course
* Effective evaluation methods will need to be put into practise to assess as to whether or not the software is proficient in providing a beginner with a good basic programming understanding while also making it an enjoyable experience for the user
* Design and interfaces of the system will have to be carefully considered as to whether they will make the platform appeal to a wider or more specific audience – i.e. anyone that is able to start learning or more primarily in classrooms
* Project is original and innovative – the aim is not to recreate programming tuition tools that are readily available

## Appendix 18: Design Archive

1. Octalysis Framework: http://yukaichou.com/books [↑](#footnote-ref-1)