2021

Codeo - Learning support research: use of automated feedback within undergraduate programming education, a case study

Bellas, Andrew

Bellas, A. (2021) 'Codeo Learning support research: use of automated feedback within undergraduate programming education', The Plymouth Student Scientist, 14(1), pp. 145-165. http://hdl.handle.net/10026.1/17334

The Plymouth Student Scientist
University of Plymouth

*The Plymouth Student Scientist, 2021, **14**, (1), 145-165*

Codeo – Learning support research: use of automated feedback within undergraduate programming education, a case study

Andrew Devin Bellas, BSc (Hons), AfCIIS

Project Advisor: Dr Shirley Atkinson, School of Engineering, Computing and Mathematics (SECaM), University of Plymouth, Drake Circus, Plymouth, PL4 8AA

Abstract

This paper describes the research process of a case study in the domain of pedagogical computer science, specifically in the context of how first year undergraduate students learn programming skills, with the purpose of investigating novel mechanisms to aide in engaging undergraduate students when learning to code, as well as add to an existing body of knowledge on computing pedagogy. The methodology employed in this endeavour was a threefold approach; stage one of the studies engaged undergraduate students in a series of data gathering exercises, such as a survey and a semi-structured interview. This allowed for the development of a proof of concept website prototype to engage students in programming exercises, before they would be automatically marked by a series of bespoke unit tests for each question. Finally, the third stage of research was gathering data on specific undergraduate experience with this research tool, where participants were asked to engage with the programming exercises, before then filling in another survey which asked questions prudent to the effectiveness of the research tool. The results of this case study show a generally positive response to interaction with the research tool, with participants highlighting that they believed the tool was useful, that feedback provided by the automated marking process was helpful, and that they would recommend such a learning system to their peers. Conclusions drawn from the case study suggest profound benefits to the use of tools such as the one implemented within this case study, as well as benefits to students seeing their peers as sources of knowledge in a cooperative manner.

Keywords: Codeo, learning support research, undergraduate, automated feedback, computing pedagogy, research tool, pedagogical computer science, pedagogy, code kata, code dojo

Introduction

Learning to program within the context of higher education has historically been challenging, as can be seen by high student failure and dropout rates from entry level programming courses (Butler and Morgan, 2007). Higher education traditionally follows the approach of structuring education in the form of lectures and practical sessions, where lectures convey knowledge and theory, and practical sessions seek to encourage students to apply this knowledge. However, there is evidence to suggest that, of students within a given cohort, there is inherently a subset that will not respond to this approach in a manner that benefits their academic growth (Bellaby, McDonald and Patterson, n.d.).

Among other factors impacting a student's ability to learn is feedback. Feedback is an important part of the learning process, as it allows students to consolidate their current level of understanding, as well as understand how to improve their current ability (Why is feedback important? - University of Reading, n.d.). A consideration for a means for programming education is the use of code dojos. Code dojos are social meetings where programmers cooperate in order to solve a specific problem (Coding Dojo, n.d.). These will be explored within this paper and are a key facet incorporated within this research.

Research Background

What is Computer Science Pedagogy?

Pedagogic computer science is the academic study and approach of practices within how the teaching of computing or computer science is conducted, and delves into methods of learning, theory of learning, and assessment, in order to create a model of good practice for conveying knowledge to students (Flint, 2018).

Research Project Definition

It is important to consider how students learn in the context of computer science education. This research project proposes a different way for undergraduate students to learn computer programming and posits an exploration of techniques in the context of pedagogy, in order to ascertain data regarding the relevancy of certain methods of education and their merit in benefiting the student experience. It is the hope of this study that the results obtained can be used to generate a discussion on alternative teaching paradigms for computer science education.

On Research Ethics

Data Protection

The Data Protection Act of 2018 dictates how personal information can be used. Research data also comes under the Data Protection Act, and as this research project stores personal data of participants, both in terms of the application, as well as subsequent interviews and surveys, it is an important legal consideration to follow the Data Protection guidelines (Data protection, n.d.).

This research project conforms to the Data Protection Act of 2018 as participant information is used only for the specified purpose of gaining data to analyse as a result of the study. This data is used in a manner that allows only its necessary processing, and has a set date for destruction. The data is also securely handled, as every approach to mitigate potential unauthorised access has been taken, as it is stored behind the authentication system of a University of Plymouth account.

The GDPR or General Data Protection Regulations is a European regulation regarding the handling and privacy of personal data. The GDPR states that data should be used for a defined, explicit purpose, that it be handled with adequate security, processed in a manner that is limited only to the express purpose the data is being processed, kept up to date where applicable, and kept for only as long as is necessary for the defined purpose of handling the data (The principles, n.d.).
The GDPR, in terms of research storage of personal information, specifies that personal data pertaining to participants such as names should be pseudonymised, which is defined by the GDPR as "the processing of personal data in such a manner that the personal data can no longer be attributed to a specific data subject without the use of additional information, provided that such additional information is kept separately and is subject to technical and organisational measures to ensure that the personal data are not attributed to an identified or identifiable natural person" (What is personal data?, 2020).

The University of Plymouth's Research Ethics Policy's section on human ethics in research specifies that human participant use must comply with four core tenets of research integrity (Pellowe, Łucznik and Martin, 2018):

- "Autonomous, informed consent", where a participant needs to have appropriate understanding beforehand about the nature of the study so they can make a well-evaluated decision on whether they would like to participate.
- "Openness and honesty", or the need to be transparent on the nature of the research, as well as the usage of resultant data.
- "Protection from harm", the burden of responsibility is placed on the researcher to keep participants free from physical or mental harm.
- "Confidentiality and data protection", participant personal data should be kept confidential and of an appropriate level of security.

Informed Consent
Within the research, participants were given a briefing before any research activity, such as surveys or semi structured interviews. Participants had to explicitly agree to consent to the study before being able to proceed, and in the case of surveys, the consent clause was written and kept. Participants were also informed not only of the nature of the research, but also how their results would be used. There was no coercion within the research process, as participants were never compensated for their time, as it was a voluntary procedure, nor was there any promise of recompense at any point before a participant took part within the study or after the study.

Research Honesty
At every point, transparency with participants was employed, as participants were informed of the nature of the research, as well as what the results would be used for, in the consent clause of surveys as well as within the briefing before a focus group.

Participant Safety
As per the University of Plymouth research ethics policy, researchers have a duty of care over their participants. Research should not allow participants to suffer mental or physical harm. Within the focus group session, participants, upon cessation of recording, were asked if they were uncomfortable with the resultant recording or

session, and should any grievances with the process be raised, they were to be addressed accordingly.

Data Management
As discussed within the legal implications of the research project, usage and processing of sensitive information within the research project has been taken very seriously. Where participant(s) have taken part in a focus group, participant data is 'pseudonymised', where a participant is codified as a string of letters, and this pseudonymisation process has been compliant with GDPR personally identifying data pseudonymisation techniques (Pseudonymization according to the GDPR [definitions and examples] – Data Privacy Manager, 2020). At no point have participant's sensitive data been publicised, nor is there any intention to do so. The data collected pertaining to participants will be summarily destroyed upon research completion.

Ethical Clearance Guidelines
This research project has met all of the above criteria as set by the ethics policy, as participants were given a briefing before any research activity such as surveys or semi structured interviews. Participants had to explicitly agree to consent to the study before being able to proceed.
Additionally, in regards to this specific project, the ethical clearance employed was for participants who were specifically students at the University of Plymouth or partner colleges who were currently enrolled. This ethical requirement has been stringently followed and indeed met throughout the project's entirety.

Literature Review
As this research aims to demonstrate how the process of teaching programming can be supported within undergraduate study a literature review is necessary to illustrate the purpose of the study and inform the results.
This literature review attempts to explore the current body of research surrounding the pedagogy of programming, the role of feedback in education, the application of code dojos in learning, and indeed current trends within programming teaching support.

On Pedagogical Computer Science
It is widely believed that feedback in the context of higher education has benefits for student performance, regardless of discipline of study (González-Marcos, Alba-Elías and Ordieres-Meré, 2017). However, there are several hurdles to providing good quality feedback within higher education. In certain cases, both lecturer and student alike display dissatisfaction with certain facets of feedback, as students and educators can perceive issues with feedback incongruously, and this disconnect can lead to issues in student performance (Henderson, Ryan and Phillips, 2019). Nevertheless, where feedback is implemented in an astute, well-evaluated fashion, such as where students are assessed in a 'learner-centered' manner, as opposed to a traditional manner, feedback is seen by students to be more beneficial, and indeed can be seen to aid self-regulated learning and student autonomy (Pereira, Flores, Veiga Simão and Barros, 2016).

It is also important to consider the role of the student, and the means with which they use the feedback they are given. Though feedback may be delivered in a timely manner and describe how to improve performance of a given student, it is important that a given student is also open and willing to accept the feedback in order to use it (Jonsson, 2012). As such, some focus must be applied on how to better encourage student usage of feedback to enhance their learning regulation.

A pressing problem with feedback in higher education is the vagueness with which it is defined as a learning tool, as can be seen in its multiple interpretations within different contexts. For example, some may visualise feedback as 'information provided by an agent regarding aspects of one's performance or understanding' (Hattie and Timperley, 2007), or rather an evaluation of a student's ability, and a final outcome of a student's actions. Though it may also be viewed as a pipeline of improvement and development, whereby the ultimate goal is to assist in guiding a student towards a specific learning success threshold from the current level of performance that they display, and that in order to achieve this, students must employ a degree of self-evaluation of themselves, and understand the desired outcome expected of them (Sadler, 1989).

Also, important to consider is the role of programming education as a means to prepare students for life within industry. It can be suggested that, through current educational paradigms, students simply do not learn the 'real world' programming skills that are required for work as a professional. (Parsons, Wood and Hayden, 2015).

On 'Code Dojos'

Code dojos are social programming sessions, where programmers collaborate on a given problem. (Coding Dojo, n.d.). In the context of higher education programming classes, code dojos can be used as an alternative means of facilitating the learning of programming as a means to introduce novice programmers to the concept. Code dojos employed as a pedagogical tool within undergraduate study can be seen as a beneficial force in reduction of absence and dropout rates within entry level programming courses, as well as heightened grade attainment (Gomes Rocha, Feraz Sabino and Rodriguez, 2018).

Specific employment of code dojos for higher education have historically been challenging, and course or discipline-agnostic, and the creation of dojos for course specific development is largely constrained by length of course. Added to this, students often value activities that gain course credit over those that are optional, meaning student turnout can be sparse when trying to organise dojos within universities (Heinonen, Hirvikoski, Luukkainen and Vihavainen, 2013).

Code dojos can also be used for grasping other concepts within software engineering, such as TDD (Test-Driven Development), and arguably, dojos offer a particularly alternative teaching method for learning to program, in that collaboration and social aspects of problem solving are prized above all else, which arguably leads to a more inclusive environment for learning, where students feel compelled to work together and see other students as sources of knowledge within the process of solving code problems. Code dojos also enable the use of a more applied learning

approach through trial and error among peers solving problems, which reinforces concepts such as pair programming and learning through practice (Batista Da Luz, Gustavo Serra Seca Neto and Vida Noronha, 2013).

Furthering the idea of the collaborative and communicative environments that a code dojo fosters, it can be seen that these aspects encouraged by dojo based learning encourage positive learning outcomes and experiences, as students see their peers as sources of knowledge, fostered by varying levels of familiarity and experience with programming throughout a group. Dojos also engage students in activities that encourage them to focus on problem solving abilities. Despite this however, programming socially is not without its drawbacks, as some students have adverse reactions to the social pressure of programming within view of peers, which can be a detriment to the learning process of some students (da R. Rodrigues et al., 2017).

On Feedback Automation

In the context of computer science education, automated feedback systems can be a useful educational tool worth exploring, as the limited success of students learning introductory level programming can be seen as reflecting the challenging nature of the subject matter. (Butler and Morgan, 2007) Among some of the causes of failure within programming education, are the lack of understanding due to methods used by lecturers within programming classes, a student's ability to think conceptually about solving a problem, and indeed the feedback available to students in relation to the specific programming task (Adu-Manu Sarpong, Kingsley Arthur and Yaw Owusu Amoako, 2013).

There are several established ways of providing automated feedback for programming exercises; namely, using automated unit test cases to assess code output, and provide a failure or success event as well as messages depending on the output of the code written by the user. Manual feedback is also used in some programming education tools, in the form of support forums. These two approaches both have drawbacks, such as test cases failing to provide ample information upon failure for a novice programmer to understand exactly where an issue within their submitted code has occurred, leading to lack of employable feedback. Manual evaluation can also fail in the context of user support forums provided by some services, as they are not only subject to human error in regards to solutions provided to a given problem, but also subject to variability in timeliness of response (Singh, Gulwani and Solar-Lezama, 2013).

Automated feedback also has practical implications for academics, such as alleviating the burden on educators, as it allows for less time to be spent on reviewing programming tasks. However, even many years after the advent of test case based automated marking and feedback for programming assignments, the issue of automated feedback systems still remains a largely unsolved problem within education (Fenwick, 2015), due to unavoidable problems within automated test feedback, such as students 'gaming' the system and merely submitting code solutions that hard output the value a test requires to pass.
Automated programming assessment can also be seen as beneficial when combined with inter-student communication, as in one such case, this approach was demonstrated to yield greater pass rates than a more traditional educational

approach. Despite this, averages of grades across the cohort did not change, potentially suggesting such a change in assessment and teaching, though beneficial in terms of aggregate student pass rates within a cohort, does not prove beneficial in idiosyncratic situations relating to specific student needs for individualised feedback (Kaila et al., 2015).

Review Conclusion

This review's purpose was to attempt to explore not only the background of this domain of research, but also the limitations of tools used within studies to automatically assess student code over a span of some years, as well as the educational use of programming dojos and general pedagogical practices in terms of how feedback can impact a student. From the literature reviewed, though some positive outcomes have been recorded from the use of automated feedback systems, it can clearly be seen that there are still fundamental flaws within this field, and tools designed to assist in computer aided learning and assessment still suffer from drawbacks which prevent the full potential of their application in educational environments. As can be seen from (Kaila et al., 2015), automated feedback can be coupled with the use of communication systems between students, in order to better the outcome of the tool's application in terms of student pass rates. Despite this, it is clear that for use within education, automated feedback and assessment systems are not yet ready to be widely deployed for practical usage, but do generate data that can be worked with and interpreted for research studies into feedback as a pedagogical tool for students.

From the literature reviewed, it can be seen that code dojos have had some success in the academic environment as tools for students to learn programming, as in one such study the use of code dojos cut not only absences and dropout rates of programming courses, but also improved overall student grades attained on the course. Code dojos also provide a stepping stone into learning other aspects of software development as a whole, such as team based practices and TDD (test-driven development), as reported in (Batista Da Luz, Gustavo Serra Seca Neto and Vida Noronha, 2013). The highly valued social aspect of a code dojo can also lead to undesirable outcomes with certain students however, as it can be seen that social anxieties while among fellow students when programming can lead to detriment in what a student can learn and take away from the dojo experience. Nevertheless, from the literature reviewed, it can be clearly seen that there is considerable merit in the application of code dojos to higher education settings for introductory level courses in programming.

Methodology

Methodology Overview

This research project followed a specific structure and operated within phases that mapped to the sprints of the project. A user-oriented design approach was taken to gathering requirements for the software tool for the research, before being developed and used to gain data for the study for analysis.

Participants

Within the study, participants were taken from a cohort of undergraduate first year computer science students enrolled at the University of Plymouth. Opportunity

sampling was used in order to gain the sample population, and a participant population of 24 undergraduate students *(n=24)* was found for the first phase of research. Of this population, one participant *(n=1)* agreed to partake in a semi-structured interview case study, and in the final stage of the study, whereby the software tool was used for gaining data, one participant *(n=1)* from the original sample and one first year student from a University of Plymouth partner institution partook.

Method of Study and Ontology

The research followed a constructivist ontological approach, and within the first phase of the study, where results were gathered for the development of the software research tool, participants were asked to complete a digital survey in order to gain requirements for the research software tool. Of those within the survey sample, 22 participants agreed to take part in the study, having indicated their approval via the ethics clause which was a compulsory start question *(n=22)*. One participant *(n=1)* refused the ethics clause of the survey, and was subsequently withdrawn from the study, and one participant *(n=1)* simply chose to not answer or interact with the survey at all, as shown in Figure 1.

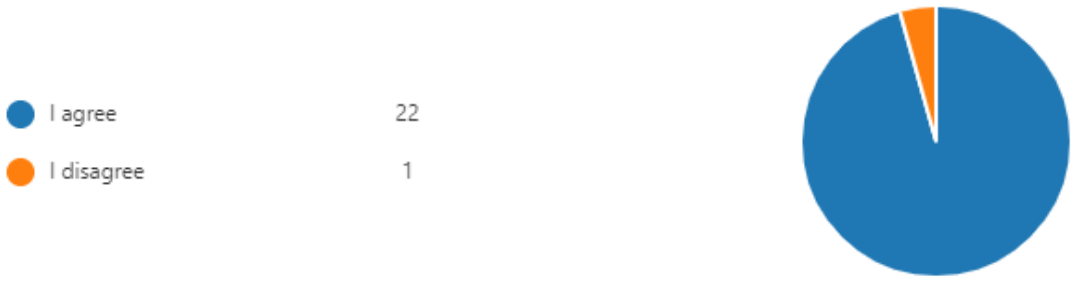

Figure 1: Ethical Disclaimer Agreement.

Of those that took part in the survey, only 13.63% agreed that they were familiar with code dojos, with a further 77.27% unfamiliar, and finally 9.09% unsure, as shown in Figure 2. The purpose of this question was to assert whether the dojo approach was widespread, or whether it was obscure.

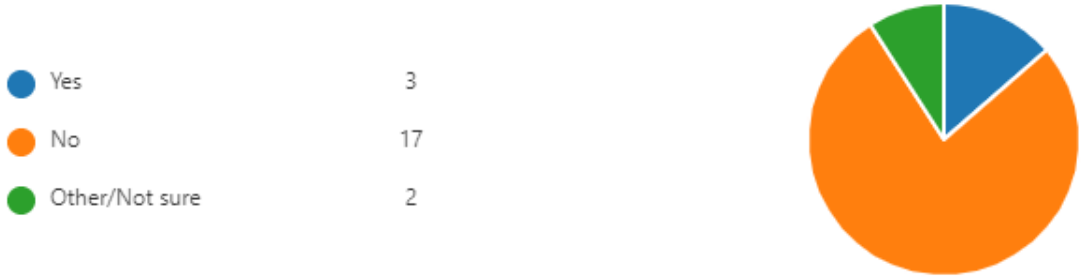

Figure 2: Familiarity with Code Dojos.

Even fewer participants agreed they knew of code katas (13.63%), when compared to code dojos. 81.81% of participants had not heard of code katas, and finally 9.09% were unsure. For the suggestion of an automated feedback system, a vast majority of participants approved of the idea, with 50% stating they strongly agree that it

would be a useful feature, and 45.5% agreeing. 4.5% were neutral to the idea, and no participants disagreed with the proposition, as can be seen in Figure 3.

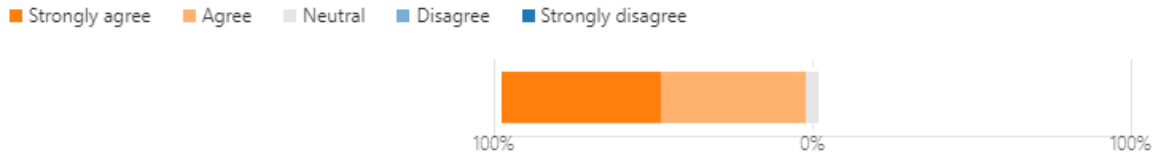

Figure 3: Approval of the suggestion of an automated feedback system.

For the suggestion of a points system, there was also a vast positive response, with 18.2% of participants strongly agreeing, and a majority agreeing (50%). 13.6% had neutral feelings towards the feature, and 18.2% disagreed, as illustrated in Figure 4.

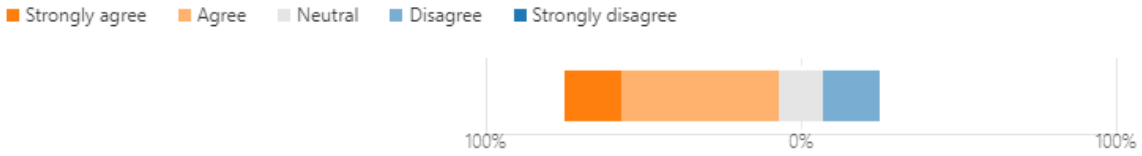

Figure 4: Approval for a points or scoring system for motivation to complete code problems.

The idea of a progress checking page yielded mainly positive responses, with 27.3% strongly agreeing, 54.5% agreeing, 13.6% having no otherwise strong feelings about the feature, and 4.5% disagreeing, as shown in Figure 5.

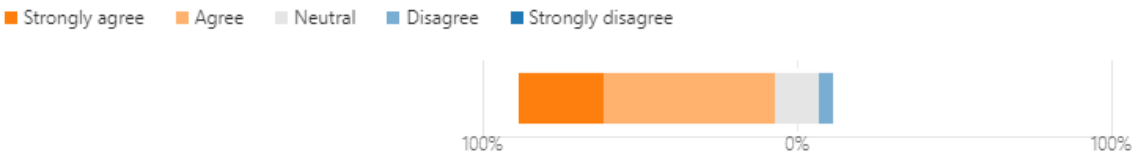

Figure 5: Approval for a progress checking page for completed code problems.

Interestingly, an overwhelming majority of participants did not agree that having a social score comparison feature was a good idea, with the majority disagree vote totalling 40.9%. 18.2% did not have any strong feelings towards the feature, 22.7% of participants agreed it was a useful feature, and 18.2% strongly agreeing it was a useful feature, as shown in Figure 6.

Figure 6: Approval for a social score comparison feature.

The suggestion of a progress bar to intuitively show how much progress a user has made was overwhelmingly agreed with, at 54.5%, with a further 27.3% strongly agreeing that it was a desirable feature. 9.1% were neutral, and 9.1% disagreed, as shown in Figure 7.

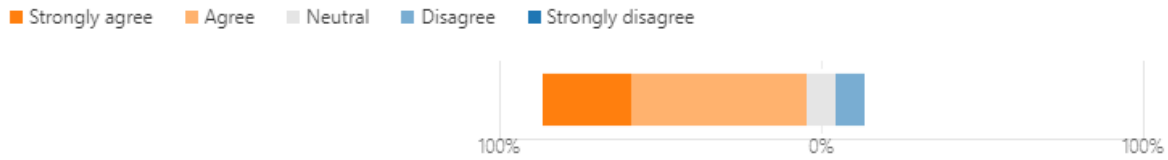

Figure 7: Approval for a progress tracking bar for code problem completion.

To further consolidate the findings of the initial survey, a case study interview was held, in order to ascertain more depth within the results of the survey. The interview consisted of a semi-structured style, where a set list of questions were asked of the participant, but elaboration on these questions was provided and encouraged organically to gain greater depth of information. This format of interview was used as it provided a means for gaining qualitative data in relation to the project with only a small number of participants. Unplanned questions were then asked in order to further gain knowledge on participant expectations. The interview was recorded after first briefing the participant on the nature of the session as well as what the results would be used for, before gaining consent to proceed and record. After the session, the participant was debriefed and asked if they were still happy to allow the use of the recording and the data provided for the purposes of this research. Upon agreement to this, the audio was used to enrich understanding of the survey results. A lexical taxonomy was then created of both the qualitative data obtained from the case study session, as well as from the survey qualitative results.

The research tool was then developed to an MVP (Minimum Viable Product) level for gathering data, before results gathering began. Participants were then asked to complete a follow up survey detailing their experience with the software. These results were then taken and analysed for use in the study to determine to what level the outcome conformed to the original research proposition.

Research Management

In managing all aspects of the research, an agile approach was taken. This involved frequently planning sprints, and the use of a Kanban project backlog for tracking progress and prioritising specific tasks and project requirements.

Research Software Tool Development

While this project was based on research, a software application was employed in order to gain data for the study. Therefore, it is prudent to document the use of this tool and the development process thereof, as this allowed for the collection of data pertaining to participants using the application, which could then be compared to participants involved in other facets of the study.

Research Tool Architecture

The research employed empirical software engineering in order to gain data for the study. The software application itself used a database for persistence of user data, an API for controlling data processes and behaviours between the client and

database, and a front end for displaying data to a participant. Evaluation of programming was performed client side, via a jQuery call to evaluate JavaScript code, and a JavaScript test file on the server was then called to test the input, before the result of the test file was displayed in the on-page console.

Design Process

Architecture

The following are the various diagrams used throughout the development of the research software tool, in order to illustrate the design process (Figures 8, 9, 10 and 11).

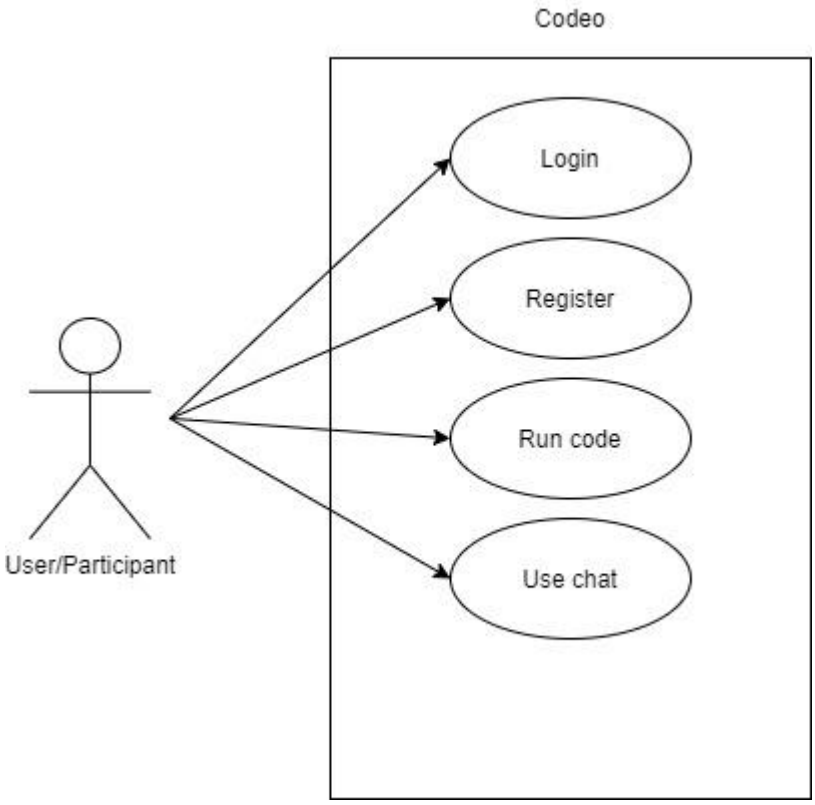

Figure 8: UML Diagram of the Research Application

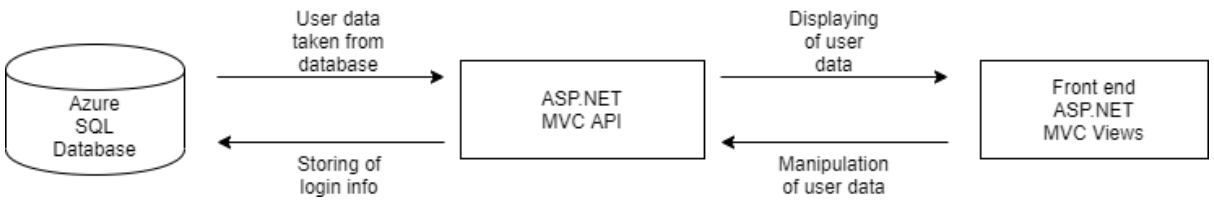

Figure 9: Research tool system architecture.

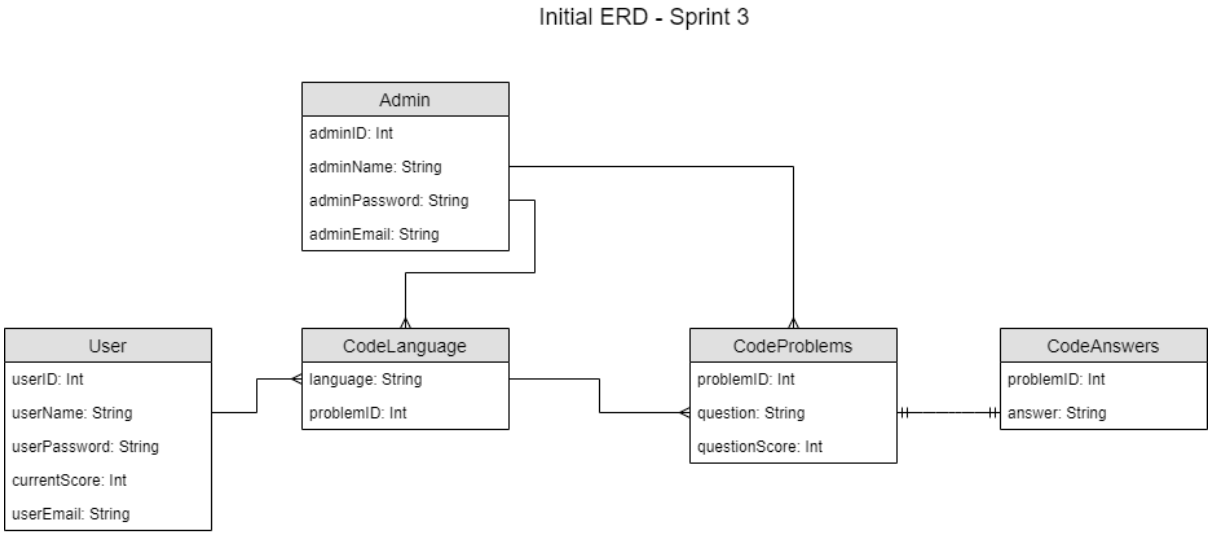

Figure 10: Initial research tool entity relationship diagram.

Technologies Implemented Within Research Tool

jQuery
jQuery is a Javascript library (jQuery Introduction, 2020) which allows, among other things, the execution of javascript code via an eval() function. jQuery was used in order to evaluate JavaScript code in order to execute in the browser, to allow a participant to undertake the code challenges presented, as well as dynamically allow for participants to toggle the level of difficulty they would like in a code problem.

ASP.NET MVC
ASP.NET MVC is a web framework that facilitates the MVC design pattern (ASP.NET MVC Pattern | .NET, 2020), and was chosen for use to effectively provide a model view controller (MVC) implementation of a website, with a code first database implementation using ADO.net database objects, an API, and a front end. ASP.NET MVC was also chosen for the other compatible technologies within its ecosystem, which were to be used in other aspects of the project, such as SignalR 2 and the Identity authentication framework. ASP.NET MVC also allows for easily running scripts, which could be used to evaluate code input.

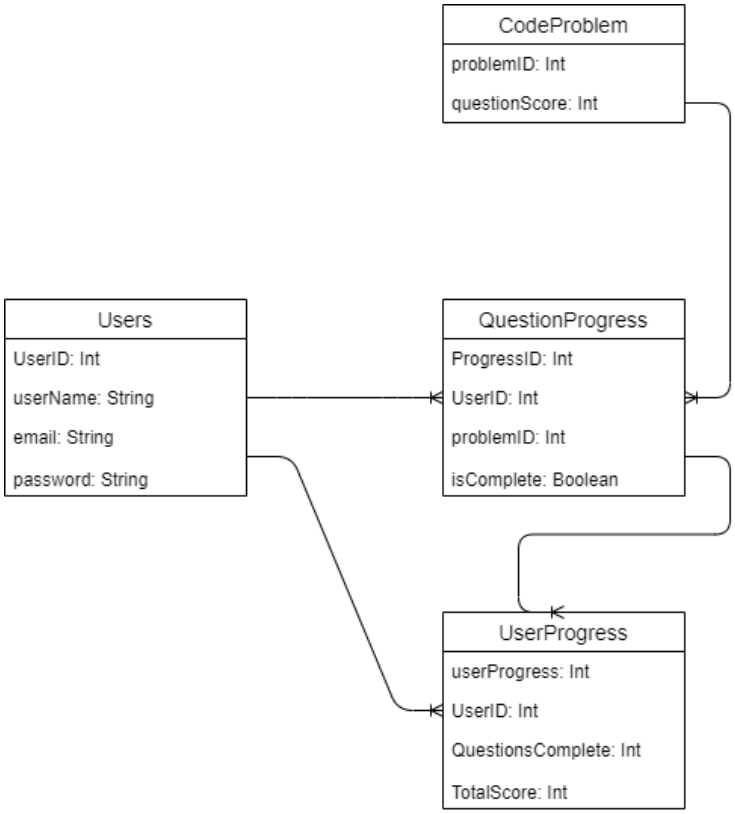

Figure 11: Finalised database entity relationship diagram for research tool.

Database Storage (Microsoft Azure SQL Database)
Microsoft Azure is a cloud service that provides a host of facilities (What is Azure – Microsoft Cloud Services | Microsoft Azure, 2020). A Microsoft Azure server was used to host an SQL database, which stored participant data for authentication on the application.

SignalR (Communicative Element)
SignalR 2 is a real-time technology which allows for instant responses in client-server interactions (Real-time ASP.NET with SignalR | .NET, 2020). This proved particularly useful in the development of the research tool, as it facilitated communication between users and as such allowed for a further level of support in solving code problems. SignalR was also used to check connections to the hub, so that users in the chat room were able to see the number of other users currently there with them.

Bootstrap 3
Bootstrap is a CSS framework, used for developing websites that employ responsive design (Bootstrap 3 Tutorial, 2020). Bootstrap 3 was used in order to style the site and make it responsive and appealing to a participant.

AceJS
Ace.js is a code editor that can be embedded in a website page, which allows for rich text formatting and syntax highlighting (Ace - The High Performance Code Editor for the Web, 2020). Ace was used to format the editor as it aided the user

experience/HCI aspect of the code experience, allowing a participant to see the appearance of a ‘normal’ code editor.

ASP.NET MVC Identity Framework
The identity framework is an authentication system for securing user accounts within ASP.NET (Introduction to ASP.Net Identity 2.0, 2020). The identity framework was used in order to secure sensitive participant data when signing up to the website in order to authenticate their participation in the study. The identity framework hashes and securely stores passwords, as an attempt to mitigate any type of malicious password acquisition.

Research Analysis Results

The results gathered within the study detail the response to automated feedback provided by two participants *(n=2)*. The male female split within the participant population was 50/50. Participants were asked to engage with the software, and then answered a series of questions. All participants consented to participating in the survey.

The results gathered indicate that participants found the software useful, with a resounding positive response, illustrating that it was both comprehensive and easy to understand. Participants also agreed that the automated feedback provided was useful, with insights such as “feedback helped me identify syntax issues” and “it let me know what I needed to fix to get code running”.
Of the participants, one agreed that the research tool improved their JavaScript knowledge, and another dictated that there was not enough time within the study to see a noticeable benefit to their JavaScript skill.
Both participants agreed that the real-time aspect of the automated feedback system was beneficial, as it both improved the quality of solutions implemented and allowed for guidance in solving the problem.

All participants perceived the chat feature would be useful as a means to see other students as sources of knowledge, and furthering this the idea of seeing peers as cooperative agents was well received, both in terms of the idea of seeing fellow students as helpers, as well as seeing fellow students as competitors.
There was divide in the use of the chat feature in some regards, however, as one participant felt it may be a distraction, while another felt it would benefit teamwork skills.

Both participants agreed that they would recommend the software to their peers as a means for learning to code. Closing comments from participants dictated that for better facilitation of the software as a learning aid, it would require further questions, as well as ‘hint’ or ‘help’ features.

Table 1: Qualitative analysis of participant feedback.

Question	**Codified Positive Response** (%)	**Codified Mixed Response** (%)	**Codified Negative Response** (%)
“Was the feedback this software provided useful?”	100	0	0
“To what extent was the feedback this software provided useful?”	50	50	0
“This software used JavaScript as a means of programming; Has this software enhanced your knowledge of JavaScript Programming?”	50	50	0
“Was the feedback system given useful in order to solve code problems?”	100	0	0
“If the feedback system was useful, what specific parts of the feedback system did you find useful?”	100	0	0
Do you perceive the chat feature to be useful?”	50	50	0
“Do you feel it is important to see your colleagues/peers as cooperative agents?”	100	0	0
“Do you feel the chat/social aspect feature would be useful?”	50	50	0

"Would you recommend this learning approach to your peers?"	100	0	0
"Are there any further comments you would like to add regarding the efficacy of this research tool?"	50	0	50

Discussion

Research Objectives

The author of this paper was driven by a deep interest in academia and the research of pedagogical practices within computer science, and the betterment of current teaching paradigms relating to the study of programming in higher education.
In terms of the research objectives, the ultimate goal of this research was to ascertain whether the use of automated feedback and a dojo environment was beneficial as a learning tool for students studying computer programming. In this regard, the study has been partially successful, as results were gathered through the use of the software and analysed to find how students perceived their experience. In future, however, were the research to be followed up, in order to gain a fuller picture of student experience with these concepts, additional students would be sampled in order to gain a more reliable idea of the success of the research software and its impact as a learning aid.

Development Objectives

The development objectives of the research application were, for a minimum viable product, to allow the collection of data for the research study. This would have entailed the creation of a means of interpreting a programming language input, and a planned 15 code problems, as well as automated solution checks for these problems, which would guide a user with meaningful feedback. The MVP also required the creation of a social feature. However, as a critical evaluation, there were many other features that could have been implemented as suggested by participants in the initial survey, in order to enrich the participant experience and potentially gain further qualitative data as a result.

Project Management

Project management was conducted using agile and through the use of a Kanban board to track progress. The management process started off well, with initial sprints and their respective tasks being completed in a timely manner. However, as tasks began to overrun, they snowballed and became increasingly difficult to manage. Measures were taken to finish tasks even when they overran, however this was not always possible.

Technology Evaluation

The technologies used were largely suitable for the research project, as they allowed the (limited) implementation of automated feedback tests on code problem inputs. However, if this tool was to be reconstructed, node.js would most likely have been used, due to easier integration with Javascript unit testing frameworks like Chai or Mocha. In terms of database technology, Microsoft Azure was less than optimal. Due to the COVID-19 pandemic, there were outages of facilities in mainland Europe, and as such there was some hinderance to the project, though this was eventually overcome.

Personal Reflection

The author's own approach to development also has to be considered in terms of critical evaluation in order to give an all-encompassing view of the successes and limitations of the project. All submissions were made in a timely manner, and through the use of Trello, tasks were accurately documented. The process for the creation of the research followed a consolidated, pre-planned structure which was split into phases and mapped to sprints. The phases of research, initial participant gathering, development of the research tool, and study follow up, were well planned, and were kept consistent from the beginning to the end of the project.

A great portion of time throughout the project was spent reading into the background of the field of pedagogy. This was perhaps one of the greatest aspects of the project process, as the literature discovered was fascinating, and allowed the gaining of both a greater breadth of knowledge in the field, and enthusiasm for pedagogical research. COVID-19 severely impacted the process with which the research project was conducted, making the task increasingly difficult due to restrictions. Adjusting to the alternative way of working took some time, and as such was a severe limiting factor of the author's work process.

In terms of technology, there was a steep learning curve with certain aspects of the research tool development. Of note was the approach to code-first database development, and the use of Microsoft Azure, which proved to be unreliable due to outages and issues relating to data migration, hindering efforts that were expected to be more timely, such as the development of a progress tracker.

Conclusions:

To conclude, the aims of the study were to gather data about the alternative ways to allow for programming learning within higher educational contexts. In this regard, the project has been partially successful, as it has garnered data, albeit in case study format, of individual undergraduate students and their experience with a more applied approach to learning with automated feedback.

As such, the project has both facilitated and demonstrated the deployment of a software application and its utility in supporting the programming learning efforts of first year undergraduate students through automated feedback. It has also systematically delved further into the wider domain of critique of the pedagogical status quo of programming education as a means to add to the existing body of literature and further discussion.

The key findings from this study indicate that this approach to undergraduate programming education are indeed helpful to learning, as indicated in the data, where automation of feedback had achieved a clearly positive response from participants in regards to learning to code.

Limitations and Future Work:
Due to lack of participants, a dojo was near impossible to facilitate, and as such the research focused on the aspects that encompass a dojo, rather than the practice itself, such as social programming, as a means to adapt to the conditions presented, and a further focus on automated feedback, hence its relevance in the literature review. Despite this, certain aspects incorporated from the code dojo philosophy appeared to prove greatly beneficial to participants, however future work may be required in order to prove the efficacy of these findings.

Though the project was mired with hurdles throughout the research process, namely the ongoing COVID-19 pandemic, and its impact on gaining participants for a further, larger scale study, future developments are required. As such, future work with a wider participant pool is required to further reinforce the findings of this case study.

Acknowledgements
I would like to extend my deepest thanks and gratitude to Dr Shirley Atkinson, who has supported and mentored me throughout the entire project, as well as nurtured my interest and passion for academia. I couldn't have wished for a better supervisor. I would also like to thank my friends and family, who have always supported me both in my academic pursuits and in life.

References

Butler, M. and Morgan, M., 2007. *Learning Challenges Faced By Novice Programming Students Studying High Level And Low Feedback Concepts*. [ebook] Monash: Monash University, pp.1 - 9. Available at: <https://ascilite.org/conferences/singapore07/procs/butler.pdf> [Accessed 16 May 2020].

Bellaby, G., McDonald, C. and Patterson, A., n.d. *Why Lecture?*. [ebook] Derby: University of Derby, pp.1 - 4. Available at: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.472.4887&rep=rep1&type=pdf> [Accessed 13 May 2020].

Reading.ac.uk. n.d. *Why Is Feedback Important? - University Of Reading*. [online] Available at: <https://www.reading.ac.uk/internal/engageinfeedback/Whyisfeedbackimportant/> [Accessed 19 May 2020].

Codingdojo.org. n.d. *Coding Dojo*. [online] Available at: <https://codingdojo.org/WhatIsCodingDojo/> [Accessed 1 May 2020].

Flint, A., 2018. *Higher Education Teachers As Pedagogic Researchers | Higher Education Academy*. [online] Heacademy.ac.uk. Available at: <https://www.heacademy.ac.uk/blog/higher-education-teachers-pedagogic-researchers> [Accessed 28 April 2020].

GOV.UK. n.d., *Data Protection*. [online] Available at: <https://www.gov.uk/data-protection> [Accessed 9 May 2020].

Ico.org.uk. n.d. *The Principles*. [online] Available at: <https://ico.org.uk/for-organisations/guide-to-data-protection/guide-to-the-general-data-protection-regulation-gdpr/principles> [Accessed 13 May 2020].

Pellowe, A., Łucznik, K. and Martin, J., 2018. *Research Ethics Policy*. 1st ed. [ebook] Plymouth: University of Plymouth, pp.6 - 11. Available at: <https://www.plymouth.ac.uk/uploads/production/document/path/12/12337/General_Research_Ethics_Policy__final_draft_V1.0_.pdf> [Accessed 20 May 2020].

Data Privacy Manager. 2020. *Pseudonymization According To The GDPR [Definitions And Examples] – Data Privacy Manager*. [online] Available at: <https://dataprivacymanager.net/pseudonymization-according-to-the-gdpr/> [Accessed 14 May 2020].

González-Marcos, A., Alba-Elías, F. and Ordieres-Meré, J., 2017. An Online Assessment and Feedback Approach in Project Management Learning. *Proceedings of the 9th International Conference on Computer Supported Education*,.

Henderson, M., Ryan, T. and Phillips, M., 2019. The challenges of feedback in higher education. *Assessment & Evaluation in Higher Education*, 44(8), pp.1237-1252.

Pereira, D., Flores, M., Veiga Simão, A. and Barros, A., 2016. *Effectiveness And Relevance Of Feedback In Higher Education: A Study Of Undergraduate Students*. [ebook] Studies in Educational Evaluation, pp.1 - 5. Available at: <https://repositorium.sdum.uminho.pt/bitstream/1822/47058/1/text.pdf> [Accessed 18 May 2020].

Jonsson, A., 2012. Facilitating productive use of feedback in higher education. *Active Learning in Higher Education*, 14(1), pp.63-76.

Hattie, J. and Timperley, H., 2007. The Power of Feedback. *Review of Educational Research*, [online] 77(1), pp.81-112. Available at: <https://pdfs.semanticscholar.org/887f/1315f3b6a84468c5efaed053b9d393861f8a.pdf?_ga=2.52320553.1830765311.1590187710-2116762472.1587821283> [Accessed 18 May 2020].

Sadler, D., 1989. Formative assessment and the design of instructional systems. *Instructional Science*, [online] 18(2), pp.119-144. Available at: <http://michiganassessmentconsortium.org/wp-content/uploads/Formative-Assessment-and-Design-of-Instructional-Systems.pdf> [Accessed 7 May 2020].

Parsons, D., Wood, K. and Hayden, P., 2015. What are we Doing When We Assess Programming?. In: *h Australasian Computing Education Conference (ACE 2015)*. [online] Sydney: Proceedings of the 17th Australasian Computing Education Conference (ACE 2015), pp.1 - 9. Available at: <https://crpit.scem.westernsydney.edu.au/confpapers/CRPITV160Parsons.pdf> [Accessed 18 May 2020].

Gomes Rocha, F., Feraz Sabino, R. and Rodriguez, G., 2018. Using Dojo as a Pedagogical Practice to Introduce Undergraduate Students to Programming. In: *2018 XIII Latin American Conference on Learning Technologies (LACLO)*. [online] 2018 XIII Latin American Conference on Learning Technologies (LACLO), pp.1 - 4. Available at: <https://ieeexplore-ieee-org.plymouth.idm.oclc.org/stamp/stamp.jsp?tp=&arnumber=8783549&tag=1> [Accessed 20 May 2020].

Heinonen, K., Hirvikoski, K., Luukkainen, M. and Vihavainen, A., 2013. Learning agile software engineering practices using coding dojo. *Proceedings of the 13th annual ACM SIGITE conference on Information technology education - SIGITE '13*,.

Batista Da Luz, R., Gustavo Serra Seca Neto, A. and Vida Noronha, R., 2013. Teaching TDD, the Coding Dojo Style. In: *International Conference on Advanced Learning Technologies*. [online] Parana: 2013 IEEE 13th International Conference on Advanced Learning Technologies, pp.1 - 5. Available at: <https://ieeexplore-ieee-org.plymouth.idm.oclc.org/stamp/stamp.jsp?tp=&arnumber=6601956&tag=1> [Accessed 16 May 2020].

da R. Rodrigues, P., Franz, L., Cheiran, J., da Silva, J. and Bordin, A., 2017. Coding Dojo as a transforming practice in collaborative learning of programming. *Proceedings of the 31st Brazilian Symposium on Software Engineering - SBES'17*,.

Singh, R., Gulwani, S. and Solar-Lezama, A., 2013. Automated feedback generation for introductory programming assignments. *Proceedings of the 34th ACM SIGPLAN conference on Programming language design and implementation - PLDI '13*,.

Fenwick, J., 2015. *Considerations In Automated Marking*. [ebook] Sydney: Proceedings of the 17th Australasian Computing Education Conference (ACE 2015), pp.1 - 8. Available at: <https://crpit.scem.westernsydney.edu.au/confpapers/CRPITV160Fenwick.pdf> [Accessed 17 May 2020].

Kaila, E., Rajala, T., Laakso, M., Lindén, R., Kurvinen, E., Karavirta, V. and Salakoski, T., 2015. *Comparing Student Performance Between Traditional And Technologically Enhanced Programming Course*. [ebook] Sydney: Australasian Computing Education Conference (ACE 2015), pp.1 - 8. Available at: <https://crpit.scem.westernsydney.edu.au/confpapers/CRPITV160Kaila.pdf> [Accessed 17 May 2020].

W3schools.com. 2020. *Jquery Introduction*. [online] Available at: <https://www.w3schools.com/jquery/jquery_intro.asp> [Accessed 16 May 2020].

Microsoft. 2020. *ASP.NET MVC Pattern | .NET*. [online] Available at: <https://dotnet.microsoft.com/apps/aspnet/mvc> [Accessed 13 May 2020].

Azure.microsoft.com. 2020. *What Is Azure – Microsoft Cloud Services | Microsoft Azure*. [online] Available at: <https://azure.microsoft.com/en-gb/overview/what-is-azure> [Accessed 12 May 2020].

Microsoft. 2020. *Real-Time ASP.NET With Signalr | .NET*. [online] Available at: <https://dotnet.microsoft.com/apps/aspnet/signalr> [Accessed 28 May 2020].

W3schools.com. 2020. *Bootstrap 3 Tutorial*. [online] Available at: <https://www.w3schools.com/bootstrap/> [Accessed 22 May 2020].

Ace.c9.io. 2020. *Ace - The High Performance Code Editor For The Web*. [online] Available at: <https://ace.c9.io/> [Accessed 14 May 2020].

C-sharpcorner.com. 2020. *Introduction To ASP.Net Identity 2.0*. [online] Available at: <https://www.c-sharpcorner.com/uploadfile/16101a/introduction-to-asp-net-identity-2-0/> [Accessed 19 May 2020].

SurveyGizmo. 2020. *What Is SPSS And How Does It Benefit Survey Data Analysis?*. [online] Available at: <https://www.surveygizmo.com/resources/blog/what-is-spss/> [Accessed 23 May 2020].

R-project.org. 2020. *R: What Is R?*. [online] Available at: <https://www.r-project.org/about.html> [Accessed 22 May 2020].

Ico.org.uk. 2020. *What Is Personal Data?*. [online] Available at: <https://ico.org.uk/for-organisations/guide-to-data-protection/guide-to-the-general-data-protection-regulation-gdpr/what-is-personal-data/what-is-personal-data/> [Accessed 12 May 2020].

Adu-Manu Sarpong, K., Kingsley Arthur, J. and Yaw Owusu Amoako, P., 2013. *Causes Of Failure Of Students In Computer Programming Courses: The Teacher – Learner Perspective*. [ebook] International Journal of Computer Applications, pp.1 - 6. Available at: <https://research.ijcaonline.org/volume77/number12/pxc3891311.pdf> [Accessed 20 May 2020].