

Python Trivia Game

Joshua Hore

20020337

UXCFXK-30-3

Digital Systems Project

Abstract

My digital project will be a trivia game that randomly chooses from a set of questions to do with Python, to make learning it more engaging and fun for new learners. The techniques that I will be making use of include gamification, choosing from a set, and designing a suitable interface through Tkinter. I will also be using Python itself to design the game because of my existing experience with using it and its building libraries such as Tkinter that can help me to design the software.

My intention is to use repetition and reward to help new Python learners with being able to learn in a way that doesn't require reading sources and making notes which may not be engaging enough for new learners to get any enjoyment out of.

By researching websites and academic articles, I have learnt more about gamification, as well as how to create the algorithms that allow my game to work. In this report, I hope to explain how I made my game, how I documented it and hopefully, allow anyone who reads this report to learn how to do the same.

Below is a link to my project on Github.

<https://github.com/j2-hore/Python-Trivia-Game.git>

Acknowledgements

Benedict Gaster- My project supervisor who helped advise me on the project, pointed me in the direction of the right places to research and gave the lectures that helped me to understand the project more.

Nathaniel, Teresa and Toby Hore- My brother, mother and father, respectively.

Anne Ford- My grandmother, who was able to provide me a quiet environment to study, during the last weeks before my assignment was due.

Elias Pimenidis- The head of the Computer Science course that I took, of which this project was part of.

Mehmet Aydin and Barhka Javed- They ran my Advanced Software Development module, where I first tried using Tkinter while here at UWE.

Table of Contents

Abstract	1
Acknowledgements	2
Table of Contents	3
Table of Figures	4
Table of Tables	4
Introduction	5
Literature Review	7
Requirements	14
Methodology	16
Design	17
Implementation	21
Project Evaluation	27
Further Work and Conclusions	29
References / Bibliography	30

Table of Figures

Figure 1: A diagram of The process of Agile SCRUM methodology.	16
Figure 2: My Use-Case Diagram.....	17
Figure 3: My Statemachine Diagram.....	18
Figure 4: A wireframe of the game when it first runs	19
Figure 5: A wireframe of the game after a correct answer.....	19
Figure 6: A Wireframe of the game after an incorrect answer	19
Figure 7: A wireframe of the game when it first runs	20

Table of Tables

Table 1: Functional Requirements	14
Table 2: Non-Functional Requirements	15
Table 3: My Diagrams	18
Table 4: My Wireframes.....	20
Table 5: Testing Table	26

Introduction

My planned project is a game that teaches people how to code in Python by choosing and outputting questions to ask from a set. Players will see how high a score they can get within the time limit. The concept is based on Gamification. This is where game-like elements, such as interactivity are used to make tasks more engaging and more fun.

From what I know about gamification already, I already know that using elements from games can make tasks that under normal circumstances, may not be thrilling enough for some, be presented in a way that increases the appeal for others. Not everyone learns well by using Google and reading textbooks, as they can contain information that can be a lot to take in. Gamification can be used as a way around this.

In the context of this report, it is designed for people new to Python and is meant to help them learn it, but in a way that requires them to think about an answer from a selection, score points from it to make their knowledge feel rewarding and record their scores to challenge them to gain more points within the time.

My motivation for doing this was because of my own willingness to learn Python, but also the difficulties that I have had doing so. While I tried reading through multiple articles and making notes, I found that the information that I needed to take in was too much for me to handle at once and it was difficult to be aware of what the most important areas to learn were. Additionally Google will not be much help for me if I can't understand the software and decide how best to phrase my questions about Python. It did not help that the year that I started this course was during the pandemic year where I was having to balance my learning with my fears of how difficult life had become and it was much harder to ask questions to my tutors as I could only do it online.

Because of this, I want to make learning Python easier for the next line of people that take courses on it so that they do not have the same difficulties that I did. I felt that using the knowledge of Python that I did have to design something that would make learning Python easier would be the perfect conclusion to my three years of trying to learn it.

The different areas of gamification that I will be looking to implement are interactivity, meaning that someone who is playing the trivia game needs to be able to engage with it in order to answer the questions, a time limit to restrict how long people play the game for at once and a scoring system that rewards points for correct answers to motivate people to want to do better by scoring more points within the time limit.

I believe that these areas are enough to motivate users to score well and if more people use the game, then they may try to compete with each other to gain higher scores, making the chance of people continuing to use the game even greater.

Even if it takes multiple uses of the game for people to properly take in the information that it provides, if it is successful at delivering knowledge of Python in an entertaining way, people may take in more information than they realise as they may see what they were doing as a fun game rather than a form of studying.

Part of learning more about the field comes from my literature review, which you will see later in this report. From doing my own research, I have learnt about the usefulness of the software that I have chosen to design my game, more about what gamification is, what examples of gamification in the field that I am looking at exist already and how they relate to my own intentions.

This report includes, in order, a literature review where I talk in detail about my search into the field and resources that I have chosen to use, the functional requirements that I have set for myself, the approach to the methodology that I have taken, a summary of the design of my application, with screenshots and diagrams to illustrate how it works further, a guide to how I wrote and implemented my game with sample code and information on any errors that I ran into, an evaluation of my game including what went well and what I would do differently if I had the time and my final conclusion on the project including what I could do next.

Programming in general is normally not an easy area in computing to learn, but I want to see if gamification is truly the right way to go to make it easier.

Literature Review

Gamification is defined as being when game-like elements are used in non-game-based applications. Examples of these include mechanics, such as rules, rewards, levelling and dynamics, such as co-operation and competition, which relate to emotion.

The reason why companies may use gamification include increasing engagement and making learning tool usage easier. This is similar to why I have chosen to create this game, because from my own experience with trying to learn Python, I have found out that when learning it, there can be a lot of information to take in relating to the different terms and methods that people who use Python regularly may need to remember. While plenty of sources exist for researching Python and making notes and Python is still one of the easier programming languages to learn, it can still be challenging. I wish to use Gamification to see if there is a way of getting new programmers to learn it in ways that could be considered much more fun and engaging.¹

An additional benefit of Gamification that I have found from another source includes making use of addiction. While this sounds like an odd benefit, it makes sense that if people constantly want to play a game that is helping them to learn, then the game would be doing its job well. By using a reward system, such as scoring, players feel happy when they do well and will continuously want to do better. The more that they play the game, the more likely it is that they'll take in the information.

If more people use the game, then they may start to compete with each other to try to gain better scores. This would also give people more of a motivation to want to do well in the games. In order to score more points, they would have to commit the answers to memory so even if a player's main goal is not to learn, but to do better, they are still taking in valuable information.²

To try and understand gamification further, I have also used journal articles. The referenced article also mentions how gamification can be used pretty much anywhere, even for sites that we wouldn't expect to use it, such as PayPal for money transfers and eBay for shopping. The article also explains how gamification provides three main features. Usability, which relates to helping users to gain awareness of features at a good pace. Trust, which means that reward systems offer a form of interaction between a user and the software, which increases the bond that the user feels that they have with the company or software and finally, motivation. People who use

¹ Bi Worldwide (2022). *What is gamification?* [online] BI WORLDWIDE. Available at: <https://www.biworldwide.com/gamification/what-is-gamification/> [Accessed 31 January 2023].

² Mulkeen, D. (2021). *The Top 5 Benefits of Gamification in Learning.* [online] Learnlight. Available at: <https://www.learnlight.com/en/articles/5-benefits-of-gamification-in-learning/> [Accessed 19 March 2023].

gamification software will be able to learn more about what they're trying to learn and this in turn may give them new ideas about how to use this knowledge.³

Another scholarly article that I have read goes into further detail about why gamification is useful in fields such as higher-learning education. It mentions how positive feedback can be a large factor in motivating people to work and lacking motivation can put people off working if they feel that it stresses them too much. Python itself is something that students will need to learn if studying computing at a University level and not everyone who starts the course may understand it that well. My trivia game is meant to take this kind of information into account and use scoring and engagement to allow people less skilled with Python to have fun learning it, but still take in a good amount of information.⁴

Another benefit of the way that I have designed my application would be its ease of use and learning. As it uses Python, it reads in a manner that is very similar to English. Methods in Python are easy to remember because they use words that people who know English already understand. (e.g. the line `for statement A = True` is an example way to start a piece of repeating code called a loop in Python and it's easy to remember how to write because we can guess from the word usage and identifiable symbols such as the equals sign that the programmer wishes the loop to run for as long as statement A is true). The reason why this was done was because the designers intended it to be not as difficult to learn as other programming languages. This fact combined with my existing knowledge of it has made it easier for me to dedicate less of my time to learning the language from scratch. If my project is successful, then it will make learning Python even easier for new programmers and its ease of use and learning would make it a good language for new programmers to start off with.⁵

One journal article that I have looked at lists a few more of Python's benefits that I considered to be worth mentioning. It's designed for use with a number of operating systems, such as Windows and Linux. While Windows is the operating system that I prefer to use, even if I faced an issue where I had to use a different computer, I would still be able to get the program working. While my software is designed for Windows first and foremost, this does suggest that looking into releasing the game for multiple operating systems could be an option. A second point that the article mentions is that Python is open source. While, this does mean that it has not been made to

³ D. Basten, "Gamification," in IEEE Software, vol. 34, no. 5, pp. 76-81, 2017, doi: 10.1109/MS.2017.3571581.

⁴ Kaufmann, D.A. (2018). Reflection: Benefits of Gamification in Online Higher Education. Journal of Instructional Research, [online] 7, pp.125-132. Available at: <https://eric.ed.gov/?id=EJ1188367>

⁵ Arora, S. (2019). *Why Learn Python? Reasons and Benefits of Learning Python*. [online] Simplilearn.com. Available at: <https://www.simplilearn.com/why-learn-python-a-guide-to-unlock-your-python-career-article> [Accessed 18 March 2023].

be specialised towards what I want to use it for, it also means that if I want to use this software then the problem of cost when acquiring software to design this application is not an issue for me.⁶

Another scholar article that I read mentions that because of the open-source nature of Python, this means that a large amount of people have already used it and found ways to make it friendly towards people who may decide to use Python for a variety of reasons. Many Python libraries already exist thanks to this. This can include libraries to assist with internet compatibility, mobile compatibility and graphical user interfaces. My own application already makes use of a graphical user interface and the knowledge that I have gained from this article suggests that it wouldn't be too difficult if I wanted to find an additional server or library that would allow me to make my application easy to find on the internet or as a mobile app.⁷

One final advantage of Python would be its ease of debugging. It contains a catalogue of what it recognises as errors and if the code fails, it can immediately stop running and explain to the programmer where the code failed and which line caused the problem, such as missing a symbol or incorrect indentation. This makes it much easier to figure out where the main issue is if the code won't run and if the programmer can't figure it out, then they could search for the error using Google to see if anyone had a similar error and how they solved it. The fact that Python is easy to use combined with it being easy to debug helps to take some of the pressure off compared with certain other languages, even when working on a new project from scratch.⁸

I have also looked into the benefits of using Tkinter, a built-in Python library that can be used for designing forms and interfaces, which I have chosen for designing the interface for my game. While it helps that I have had experience with using it in previous projects, Geeks for Geeks has elaborated on the advantages further.

Firstly, since Tkinter is already a built-in Python library, it does not require any additional installations to run as long as I remember to add a line telling the code to import the Tkinter library. This makes it easier to run and work with Python. Tkinter also has a smaller library to another library called PyQt. While this does make Tkinter less advanced-looking and more dated compared to it, it's also easier to make sense of when compared with PyQt. This is useful because I do not wish to dedicate too much of my time solely on trying to learn the code and I should not

⁶ VIDUKA, D., KRAGULJAC, V. and LIČINA, B., 2021. A COMPARATIVE ANALYSIS OF THE BENEFITS OF PYTHON AND JAVA FOR BEGINNERS. *Quaestus*, (19), pp. 318-327.

⁷ Lin, J. W., 2012: Why Python Is the Next Wave in Earth Sciences Computing. *Bull. Amer. Meteor. Soc.*, 93, 1823–1824, <https://doi.org/10.1175/BAMS-D-12-00148.1>.

⁸ TechVidvan Team (2019). *Python Advantages and Disadvantages - Step in the right direction - TechVidvan*. [online] TechVidvan. Available at: <https://techvidvan.com/tutorials/python-advantages-and-disadvantages/> [Accessed 18 March 2023].

need to write code that is too complicated to understand, because when I considered how I could design my project, I realised that the interface could entirely be created through simple Tkinter widgets.⁹

I've also looked into what a Tkinter guidebook called the Tkinter GUI Application Development Cookbook says about Tkinter's benefits. While, it is part of an open-source software, it is tailored to collaborating with Graphical User Interfaces and despite being a single library, it contains a large amount of functions that can make Graphical User Interface's easier to be designed and run. This includes entering information, storing information, and designing apps that perform long-running actions without cutting too much into network resources.

This all suggests that Tkinter is an easy library to work with for designing interfaces and if the code is successful, then it will not require any additional servers to properly work unlike other Python applications. Because of all this, I believe that I have made good choices in the languages and libraries that I have chosen and have tried to make the most of using them to design the application that I have envisioned.¹⁰

There are also sources that I have looked at that have given real-world examples of Gamification. These include Nike+, which set challenges, rewards for completing them, and leader boards to make workouts more fun and motivate people to do better.

Another example would be Google Forms, which used processes such as creating reward badges or interactive submission systems to benefit not just the user, but entire groups of employees or students.

One final example would be HP's Utility Cup, which allowed people to compete based on their knowledge of cyber security. This shows that gamification has been used in ways that relate to computing knowledge before.¹¹

Additionally, I have looked up some tips to try to increase the appeal of my trivia game. Some of the tips that I found included making sure that the correct answer is not always in the same place, so that no-one thinks that they can win just by constantly pressing one button and making wrong answers plausible, keeping questions relevant, but not too difficult, with only a few questions that

⁹ GeeksforGeeks. (2020). *Python Tkinter - Entry Widget*. [online] Available at: <https://www.geeksforgeeks.org/python-tkinter-entry-widget/>. [Accessed 8 March 2023].

¹⁰ De Paz, A.R., 2018. *Tkinter GUI Application Development Cookbook: A practical solution to your GUI development problems with Python and Tkinter*. Packt Publishing Ltd.

¹¹ Information on these three examples come from Growth Engineering (2018). *What is the Definition of Gamification and What Does it Mean?* [online] Growth Engineering. Available at: <https://www.growthengineering.co.uk/definition-of-gamification/> [Accessed 31 January 2023].

really try to get people to think. After reading this, I have considered how I could generate the questions randomly, but I will still need to think about the difficulties of the questions that I ask.¹²

I have also looked into a source to see if there are examples of gamification I can list that exist to help people learn coding already. The referenced source lists twelve examples, but I will only be describing three of them and what I believe that I can learn from them.

Firstly, the article mentions Codecademy. It's a form of free-to-use Gamification designed for beginners to learning code. They contain 100 lessons for a range of coding software, including Python and since it is designed for beginners, it starts with the basics, allowing anyone, regardless of code skill to make use of it. Although my own trivia game will teach people Python in a different way, knowing about this other example has allowed me to consider additional points about my game. These include how easy I can make my game for people to read, since it is meant to be for people new to Python, and if my own game can be modified easily for learning a different programming language.

A second example is Treehouse. While it requires a paid subscription, unlike my own software, that I wish to be free to use to make it more likely that people will want to use it, I still consider it a useful software to talk about. It uses quizzes to help people learn code, similar to what I planned, but this software shows how such use of gamification can be expanded on. This includes offering points and badges for notable progress. As I mentioned in some of my non-programming-based examples, if people are rewarded for what they do, it makes them feel happy and want to do it more. While I do not plan for my final code to contain badges, it will contain a point system that I have added for the purpose of positive feedback and to try to motivate people to aim for higher scores.

A final example of programming gamification is Checkio. This software shares coding problems with others and encourage them to communicate with other people who use Checkio to work together to solve these problems. If I gave my own trivia code any kind of online functionality, then it would be a leader board so that people wanting the highest scores have to compete with others, which would be another form of motivation. However, after learning about this software, it is interesting to see that online gamification does not always have to mean that people have to compete with each other. They can also work together to learn Python together. If I ever design

¹² Second Street. (2023). *5 Tips for Building Awesome Trivia Quizzes*. [online] Available at: <https://uplandsoftware.com/secondstreet/resources/blog/building-trivia-quizzes/> [Accessed 31 Jan. 2023].

another form of gamification, I may consider a co-operative one to find out if people learn Python better by themselves or with others.¹³

I have also looked into examples of software for creating trivia games of different kinds. This is to see if there is anything that these kinds of software provide to the people that choose to use them, and if I can take any kind of inspiration from them.

Firstly, Kahoot. Already a favourite of people trying to design trivia questions for different subjects. Its main benefits to people include its versatility. People can participate in Kahoot quizzes from both their browser and from their phone. It's also varied in the kinds of questions that it can generate. Not all Kahoot questions need to have four answers tied to them. Some can have only two answers if the question is true or false. It also allows competition with up to nine other players and more if people pay for a membership. While I am not keen on having people who use my own software pay money for it or even for a membership. Mobile compatibility, question variety and competitive gameplay are all ideas that I may consider if I consider my own trivia game a success and want to expand its capabilities further. I also believe that it may be possible to easily adapt my game to suit different subjects, like Kahoot can do if I can construct a working algorithm for my own game successfully.

This game will be designed to help people learn Python, but I will also be using the Python programming language itself to code the game. The referenced source at the end of the following paragraph mentions a few of the benefits to programmers that Python can offer them. These include its versatile design. Python is not specialised towards any purpose in particular and consequently, there are a variety of purposes and fields that it can be used for. These include scientific computing, security testing, finance and game design. While it was not designed specifically with game-design in mind, I have used it for a number of different concepts already and this is still beneficial, as I can make use of some of my existing Python knowledge to see what I can make use of to work with designing this game.

Houseparty is another interesting example. The article mentions that it became popular during the recent Covid-19 pandemic, as it was a way for people to interact and play quizzes with each other even if they were in different homes. In addition to trivia questions. If the device that you are using has a camera, the app can use it to allow you to speak to other guests face to face, in a manner similar to Zoom or Microsoft Teams, which also became popular as a result of the Covid Pandemic. Designing an app that allows people to socialise could also be a good improvement

¹³ All information on coding gamification examples come from Zietek, T. (2016). *12 gamification platforms that help learn coding*. [online] Medium. Available at: https://medium.com/@tom_z_official/12-gamification-platforms-that-help-learn-coding-814aeb04341e [Accessed 4 April 2023].

that I could make to my own game at some point. Not just because of making it social-distancing friendly, but also, I want people to have fun trying to compete with each-other and wouldn't want them to feel alone when they're trying to learn Python. This is another way that I could make learning Python friendlier for beginners.

A final application that I wish to talk about is Trivia Maker. This application can also make multiple choice trivia questions, but it does not stop there. It can also design questions in other manners. One of the other styles of questions that it can ask is feud style, which appears to be an alternative approach where a subject is given, and the individual players gain points if they can select the same ones that are hidden from the audience. An example of how this could be used in Python is providing a problem and listing alternative solutions that people have to guess. The other style is grid style. This one seems to position four different questions relating to four different subjects and award different points based on which question was answered. I have already decided to provide a scoring system to reward people who get correct answers and scoring more points for harder questions could make people trying to learn Python feel even more proud of what they know already. While I do not plan on including either additional style in my game currently, I could see writing trivia game software in other styles to be an idea to consider, as this could provide further appeal to people trying to learn Python, depending on the way of learning Python that they consider best for them.¹⁴

¹⁴ Examples from all three trivia games come from Voloshina, A. (2020). *The 6 Best Software Apps for Running Your Own Trivia Night*. [online] Trivia Bliss. Available at: <https://triviabliss.com/trivia-night-software/> [Accessed 5 Apr. 2023].

Requirements

Functional Requirements

ID	Priority	Description
F1	MUST	Randomly select a question to do with Python and output it to the user alongside the matching answers.
F2	MUST	Award a point to the user if they get a correct answer and give them nothing if they get it wrong.
F3	MUST	Reset the score each time that a request is made to start the game from the beginning.
F4	MUST	Have a "Start" button which starts the timer and begins the question and answer answering algorithm when it runs.
F5	MUST	Have a "Stop" button that stops the timer and prevent any answers from being chosen until "Start" is pressed to continue the game.
F6	MUST	Give the user four different options for the answer.
F7	SHOULD	Use a timer to challenge users on how many points they can score.
F8	SHOULD	Have a "Reset" button that the user to start the game from the beginning regardless of if they're currently running the game or not.
F9	SHOULD	Allow the user to choose the length of the timer from 30 to 300 seconds (1/2 of a minute to 5 minutes exactly)
F10	COULD	Have separate databases for "Easy" and "Hard" questions and provide two start buttons relating to which set of questions the user would prefer.

Table 1: Functional Requirements

Non-Functional Requirements

ID	Priority	Description
N1	MUST	Provide an interface to the user that displays the question and all specified buttons in a single window.
N2	MUST	Be able to handle scores of at least three digits.
N3	MUST	When "Start" is pressed, output the first question and answers within three seconds.
N4	MUST	When the previous question has been answered, select each follow-up question and answer also within three seconds.
N5	MUST	Generate a follow-up question regardless of if the answer was right or wrong.
N6	MUST	Stop the timer if "Stop" is pressed, but have its value stay at its current value without going down.
N7	SHOULD	Start the timer at the number that the user requests and then decrease it to 0 at intervals of 1 second exactly.
N8	SHOULD	Be able to output at least 30 different questions to do with Python as well as appropriate answers.
N9	COULD	Remember the best scores and times and keep a leader board of them.
N10	WON'T	Be designed to run perfectly on MAC computers. This is to test if the game can run well enough on Windows first.

Table 2: Non-Functional Requirements

Methodology

I would like to credit the referenced presentation with teaching me about the concept of methodology. This term is another way to describe the approach that I plan to take when I design my game.¹⁵

For my approach to this project, I will take an approach to my methodology known as Agile SCRUM Methodology. (See diagram below) This is a methodology where development is done in short episodes called "Sprints." This can be seen through how every other week; I have had meetings with my project supervisor to explain how far I have come on my project and determine what actions I should do next.

This has helped to make sure that I am constantly in contact with my supervisor and receiving feedback to make sure that I am always on the right track or making sure that I know if my plans for the project need to be worked on at all.

The source mentions that taking approaches where the designers do not meet with the supervisors as often can lead to a lack of effective communication with them and projects needing to be greatly changed halfway through or even close to the end of development. Agile SCRUM methodology aims to reduce the likelihood of this, and I credit it as being helpful in always knowing what I should be doing next for my project.¹⁶

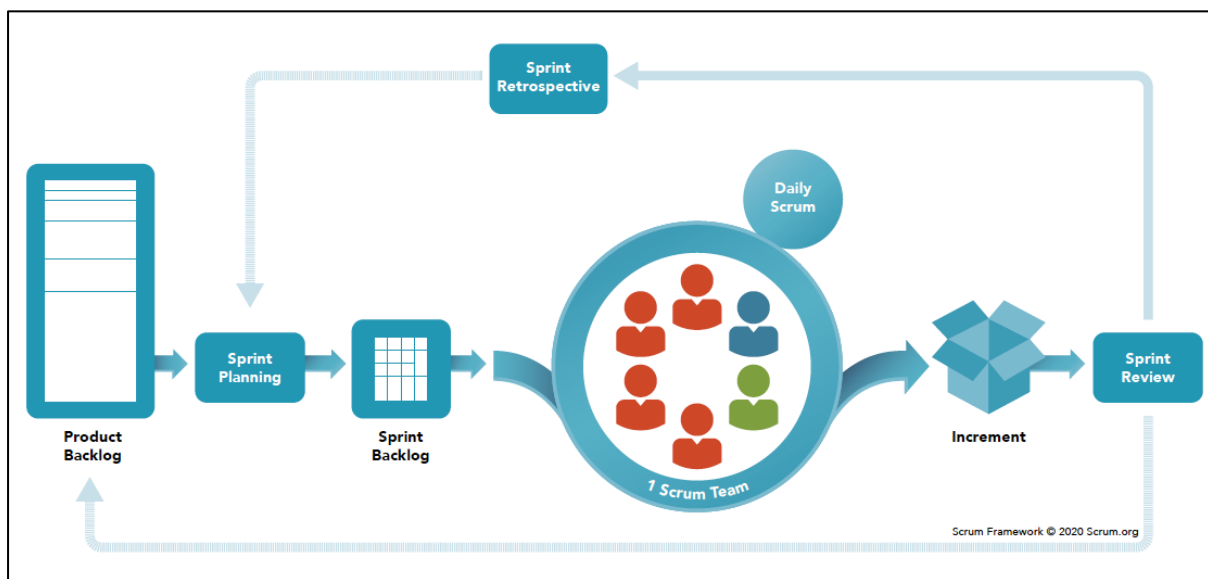


Figure 1: A diagram of The process of Agile SCRUM methodology.¹⁷

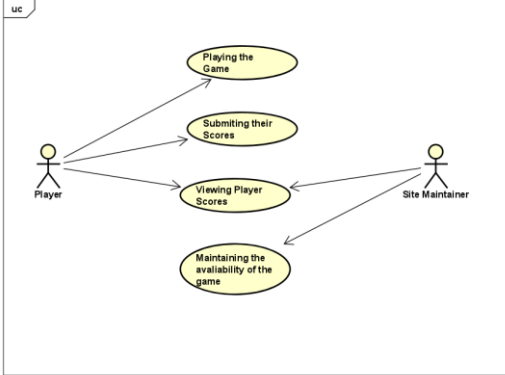
¹⁵ Simons, Chris. and Jayatilake, Dilshan. (2022) *Lifecycles for Large Scale Software Development* [PowerPoint presentation]. Available at: https://blackboard.uwe.ac.uk/uwenav/ultra/courses/351106_1/cl/outline (Accessed 20th March 2023)

¹⁶ Peek, S. (2013). *What Is Agile Scrum Methodology?* [online] *Business News Daily*. Available at: <https://www.businessnewsdaily.com/4987-what-is-agile-scrum-methodology.html> [Accessed 20 March 2023].

¹⁷ Scrum.org. (2023). *What is Scrum?* [online] Available at: <https://www.scrum.org/learning-series/what-is-scrum> [Accessed 21 March 2023].

Design¹⁸

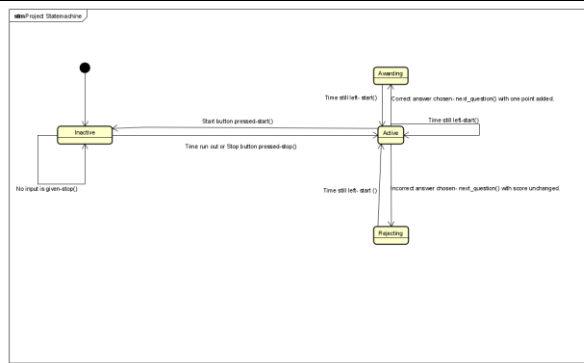
Architecture

Diagram	Cross-Reference	Screenshot	Description
Use-Case	Figure 2: My Use-Case Diagram	 <pre> graph LR Player((Player)) --> UC1([Playing the Game]) Player --> UC2([Submitting their Scores]) Player --> UC3([Viewing Player Scores]) SM((Site Maintainer)) --> UC3 SM --> UC4([Maintaining the availability of the game]) </pre>	<p>This use-case diagram shows the different people involved in the trivia game. Currently the only two people who interact with the game are the people who play the game, who will be able to submit their score to the leader board, should the game someday have a way for people to compete, as well as whoever maintains the site, should the game have a leader board, both will have access to it.</p>

¹⁸ I wish to credit Admin (2021). *Difference Between High Level Design and Low Level Design*. [online] BYJUS. Available at: <https://byjus.com/gate/difference-between-high-level-and-low-level-design/> [Accessed 16 April 2023].

State machine

Figure 3: My State machine Diagram

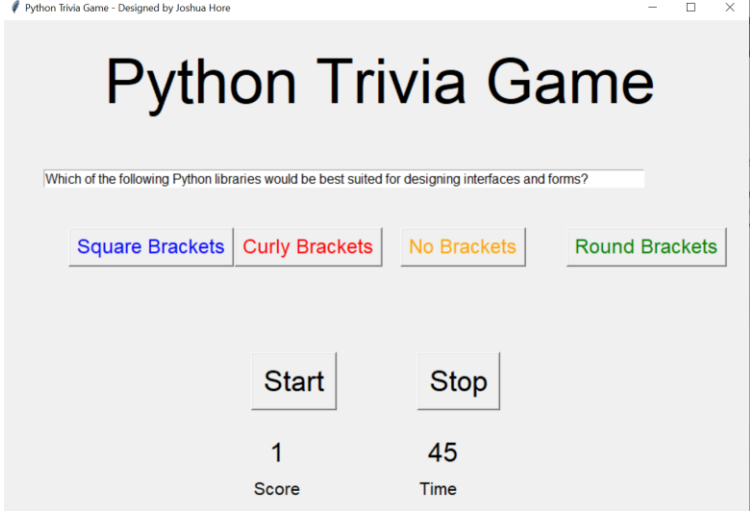


This diagram shows the states that the trivia game can be in. It won't be running when the game is first opened, but will do so, when the "Start" button is pressed. It will go back to being inactive if the timer reaches 0, or the "start" button is pressed.¹⁹

Table 3: My Diagrams

¹⁹ I wish to credit Visual Paradigm (2019). *What is State Machine Diagram?* [online] Visual-paradigm.com. Available at: <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-state-machine-diagram/> [Accessed 16 April 2023].

Wireframes

Screenshot	Cross-Reference	Description
	<p>Figure 4: A wireframe of the game when it first runs.</p>	<p>This is how the game looks when it is first run. No question has been set and the answer buttons have placeholder labels. By default, the score always starts at 0 and unless, adjusted, the timer is set to 60 mins exactly.</p>
	<p>Figure 5: A wireframe of the game after a correct answer</p>	<p>This is how the game looks after a button labelled with the correct answer is clicked. The score will raise by 1. It will also randomly select a different question.</p>
	<p>Figure 6: A Wireframe of the game after an incorrect answer</p>	<p>If one of the other three buttons that each contain an incorrect answer are clicked. Another question is randomly chosen as before, but the score will not raise.</p>

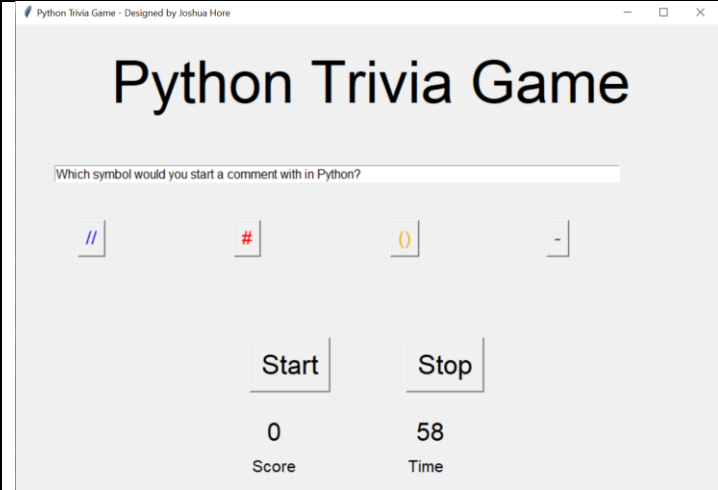
	<p>Figure 7: A wireframe of the game when it first runs.</p>	<p>This is how the game looks straight after the “Start” is pressed. A question is randomly chosen from a set and appropriate answers are assigned to each buttons.</p>
---	--	---

Table 4: My Wireframes

Implementation

Question Dictionaries

```
questiondicta = {  
    "question": "How would you finish a line containing a function  
definition?",  
    "answers": ["():", "();", "()-", "()"]  
}
```

```
questions = [questiondicta, questiondictb, questiondictc]
```

Each possible question has a dictionary containing the question and a list of four possible answers. This is an example of one of them, but similar code exists for every combination of questions and answers.

The first value in each list is the correct one and a set contains the names of every question-and-answer dictionary that I have set up. This will be acknowledged by later code.

I did run into an issue at one point where some of the answers were too large to display in the answer buttons properly, but I was able to solve this by simplifying the answers.

Question and Answer Generation

```
q = questions[random.randint(0,2)]  
chosen_question = q["question"]  
question.set(chosen_question)  
chosen_answers = q["answers"]
```

This code runs whenever a new question needs to be decided. (When the “Start button is pressed and when an answer is chosen)

Each time this code runs, a random value from the “questions” set is chosen. Each value in the set matches the name of a question dictionary and when a certain value is chosen, then the code knows to work with the chosen dictionary’s questions and answers.

Mappings and Permutations

```
answer_mappings = list(itertools.permutations([0, 1, 2, 3]))  
mapping = list(answer_mappings[random.randint(0, len(answer_mappings))])  
  
answer_a.set(chosen_answers[mapping[0]])  
answer_b.set(chosen_answers[mapping[1]])  
answer_c.set(chosen_answers[mapping[2]])  
answer_d.set(chosen_answers[mapping[3]])
```

While mapping questions is straightforward as there is only one per dictionary, answers are more complicated. To allow the answers to be set successfully, I have used a feature of Python known as permutations.

A set called “answer_mappings” is made, containing four values. Another list called “mapping” is set up containing the same 4 values but rearranged in a random order. These numbers are then linked to the “chosen answers” set and thanks to this, each value in the chosen_answers set can be assigned to each of the answer values at random.

Stop Method

```
def stop():
    question.set("")
    timer.time = None
    answer_a.set("Answer A")
    answer_b.set("Answer B")
    answer_c.set("Answer C")
    answer_d.set("Answer D")
```

This code is for the method that triggers when the “stop” button is pressed. This resets all of the values to how they were when the game first opens. The score is reset and set back to 0, the timer reset, the value of the question is made blank and all of the answer buttons are given generic values.

Timer

```
def timer(time):
    TimeSpace["text"] = time
    if time > 0:
        window.after(1000, timer, time-1)
```

This is my timer method. It runs when the “Start” button is pressed and the code is saying to decrease and output the time value by 1 each second (tk.after is measured in milliseconds. One second is 1000 milliseconds), but when the timer reaches 0, then it stops altogether until the start button is pressed again.

20

Testing Table

These are the tests that I carried out to check that my code met the functional requirements.

²⁰ I would like to thank the following sources: for providing me with timer code that I could adapt to fit my own program: Anindya (2023). *How to make a Stopwatch GUI in Python Tkinter*. [online] ThinkInfi. Available at: <https://thinkinfi.com/how-to-make-a-stopwatch-gui-in-python-tkinter/> [Accessed 23 Apr. 2023]. And Stack Overflow. (2017). *python - Basic Tkinter countdown timer*. [online] Available at: <https://stackoverflow.com/questions/34029223/basic-tkinter-countdown-timer> [Accessed 23 April 2023].

Test Number	Requirement	Description	Expected Outcome	Actual Outcome	Status
1.	F1	Randomly select a question to do with Python and output it to the user alongside the matching answers.	A question should be chosen at random when the "Start" button is pressed.	Works as described	Pass
2.	F2	Award a point to the user if they get a correct answer and give them nothing if they get it wrong.	The score should raise by 1 if the answer was correct, but not if it was incorrect.	There is currently a bug in the code that says "index out of range" after certain answer choices.	Fail
3.	F3	Maintain the score each time that a request is made to start the game from the beginning.	The "score" value should go back to 0, whenever the "Start" button is pressed and this should be visible in the program.	This works fine.	Pass
4.	F4	Have a "Start" button which starts the timer and begins the question answering algorithm when it runs.	The timer and question generating algorithms should not begin when the window runs, but should both begin when the "Start" button is pressed.	Works as expected	Pass
5.	F5	Have a "Stop" button that stops the timer if pressed.	If the game is running, then the timer should go back to 60.	Currently, a bug exists in the program that causes the timer to continue running even after a "stop" request.	Fail
6.	F6	Give the user four different options for the answer.	Four different answer buttons, each with distinct colours should display on the screen when the window opens and whenever the game runs.	All four buttons are outputted successfully with matching colours.	Pass
7.	F7	Use a timer to challenge users on how many points they can score.	There should be a 60 second timer present in the game that starts counting down	This has been implemented in.	Pass

			when the code runs.		
8.	F8	Have a "Reset" button that the user to start the game from the beginning regardless of if they're currently running the game or not.	The game should start from the beginning with the timer reset to 60 and score set to 0, but with the timer and question generating algorithms still running.	I chose not to implement this button at all, as I realised that it would make my code simpler to have the "Start" button double as having this feature when the code runs.	Fail
9.	F9	Allow the user to choose the length of the timer from 30 to 300 seconds (1/2 of a minute to 5 minutes exactly)	The timer should be able to count down when any number between 30 or 300 is set as its value, but an error message should be displayed if a lower or higher value than that range is set.	This has not been coded, as the "time" value is recorded as a label.	Fail
10.	F10	Have separate databases for "Easy" and "Hard" questions and provide two start buttons relating to which set of questions the user would prefer.	Two separate buttons should exist in the program for starting the game. Selecting the "Easy Game" button makes the game work with the dictionary of easy questions, while the "Hard Game" button makes it work with the dictionary of hard questions.	I have not implemented different dictionaries or game modes and the one that runs when the "Start" button is pressed is the only one that I have implemented.	Fail
11.	N1	Provide an interface to the user that displays the question and all specified buttons in a single window.	A Tkinter Window displaying the game that I have coded should appear whenever I run the code.	This window can be seen when the code runs.	Pass
12.	N2	Be able to manage scores of at least three digits.	I can set any number from 100-300 as the value of the timer and it can count down from	It can indeed count down from three-digit numbers.	Pass

			this as well as if the value had only two digits.		
13.	N3	When "Start" is pressed, output the first question and answers within three seconds.	There should be notable delay between pressing the "Start" button and the game running.	No notable delay exists when starting the game.	Pass
14.	N4	When the previous question has been answered, select each follow-up question and answer also within three seconds.	The follow-up questions and answers should be generated as fast as they should when the "Start" button is pressed.	There is no delay when doing this either.	Pass
15.	N5	Generate a follow-up question regardless of if the answer was right or wrong.	A new question is generated by the program when the answer is correct and this should also happen even if the answer was incorrect.	It can generate a new question regardless of a right or wrong answer.	Pass
16.	N6	Stop the timer if "Stop" is pressed, but have its value stay at its current value without going down.	The timer should freeze on the value that it is currently set to if "Stop" is pressed and not change.	Currently there is another bug in the code that causes this to not work properly.	Fail
17.	N7	Start the timer at the number that the user requests and then decrease it to 0 at intervals of 1 second exactly.	The timer value goes down by the second when the game runs.	It does go down by the second.	Pass
18.	N8	Be able to output at least thirty different questions to do with Python as well as appropriate answers.	The program contains a set of thirty questions and answer dictionaries and has an equal chance of using any of them each time a new question is needed.	The code that I am working with only works with three different questions.	Fail
19.	N9	Remember the best scores and times and keep a leader board of them.	The game should be able to save the highest scores and record them in a leaderboard that any player has the option to view.	The game does not have a score saving system or a leaderboard.	Fail

20.	N10	Run perfectly on a Windows Operating System.	The program should not crash at all when I try to run it or carry out any of the above tests, for which I will be using a Windows 10 operating system to test the game.	It doesn't crash. Windows 10 has been the perfect operating system for me when coding and running my game.	Pass
-----	-----	--	---	--	------

Table 5: Testing Table

Project Evaluation

From explaining my program in full detail in this report, I hope that I have given an idea of the research and coding that it has taken to bring my project to life. It has not been an easy past few months for me, but I am proud to have finished this project with a program that works and I can present and document.

Python has indeed proven useful for designing my project, as has the Tkinter library, just as I thought that it would be. However, like when using most Python libraries, it's not always going to be straightforward how I write my code. I have to look up the best methods or elements to get every part of the code, including both interfaces and algorithms working.

You can see from the many sources that I have added, how much dedication that I have put into looking into every method that I need to get my project working. It has not been easy, but I consider learning the methods the most difficult part of writing a project. Should I need to write programs containing a timer or random statement generation again, I feel that it would be easier. Though I still feel that I have a long way to go before I feel like I've truly mastered using Python.

I have taken measures to ensure that I am meeting with my supervisor to discuss my project at least every two weeks, from reviewing my meeting logs, I can see that I have aimed to ensure that I do not forget about my project no matter which week it is. I have not always been able to stick to my plans. For example, the timer was meant to be one of the first elements that I implemented to my game, it actually became one of the last. Regardless, the closer I came to the deadlines, the better I became at sticking to my plans. I aimed to at least have the question generating algorithm, a scoring system and a timer working before I sent the project. I am glad to have finished all three before submission.

While it was difficult to decide what to include in my literature review to provide a good summary of how my game could fit into a wider world for computer users, I am still proud of what I came up with. The information there includes the concepts that I have considered when working on my project, what programming languages I hoped to make use of for my project and if there is anything out there already that is similar to my own project and how they compare. While it was difficult finding the right sources, I have used a mixture of the internet and scholarly articles for my research. This demonstrates that I can vary the sources that I look at, yet still talk about all of them when I write my review.

In terms of how the project itself turned out, I believe that I have made my interface clear and my question and answer generating algorithm works well. Evidence of how well my game has turned out can be seen in the testing table. However, due to time constraints, I could not achieve everything that I planned.

My final trivia game was intended to have at least thirty questions, but due to difficulties in testing the code and choosing to prioritise this, I was only able to implement four questions. However, the fourth one was a late addition. This suggests that my code is able to manage new questions and adapting it to do this is not difficult.

I am glad to get my timer working. It counts down as expected, but I had little success in getting it to stop when "Stop" is pressed. This is a concern, but I am certain that the solution is a simple one that will require just a bit more research.

I am also glad that my question generating algorithm works. At first, I was unsure about where to look at all for the right information to achieve it, but my supervisor was incredibly helpful in

explaining to me what I needed and where I should look. My main concern with it at the moment, however, is a bug within the code that seems to make it not always identify the right answer as correct.

I am disappointed that I couldn't solve all of my problems or achieve my code exactly as I wanted it to look, but I can't put myself down completely. I was still able to figure out permutations, which was a major concern when working on my project.

Regardless I am glad to have had some success in envisioning an idea for a program and learning and understanding how there is always a method in Python that can help me to code these ideas.

Further Work and Conclusions

As I have already explained in my evaluation, while I came up with a lot of promising ideas for how I could design my project and make it stand out, many of my ideas proved to be too ambitious for my abilities within the short time frame that I had to design my project. However, from considering where my project lacks, I can see the areas in which I can do better and what I would do differently if I had more time.

Firstly, I would expand the number of question dictionaries to thirty. My code works well with four questions and as long as the methods for choosing questions at random are kept aware of how long the dictionary currently is, expanding the set of dictionaries will not be difficult. My intention from the beginning was to help people to learn Python and this intention has not changed throughout working on this project.

If I really want to continue to make it more helpful, I've had other ideas for how to improve it, such as making sure that it doesn't output the same question twice and considering how I can make separate dictionaries for easy and hard questions. Additionally, the answer buttons cannot fully display answers if they are too long, I will need to find a way around this as well, as it will improve what I could ask.

I also think that I need to work out how I debug the program to improve how well it works. The issues that I have to solve at the moment include figuring out why the code is not matching correct answers properly, find out how to get the code to stop the current timer when "Stop" is pressed, and reset the timer altogether rather than run it twice when "Start" is pressed and figure out how to ensure that the game only runs when "Start" is pressed and not have this happen if the answer buttons are clicked.

If I can solve these issues and be happy with the results, then I could write further trivia games in different styles. Ideas for what I could come up with next can be seen in my literature review. I could also experiment with other Python libraries to design this same game.

While it would not be easy to learn a new Python library from scratch, something that my literature review mentions. I could try designing this same game using the PyQt library that I mentioned in my literature review to find out if it enables me to design similar programs with less errors or at least a better looking interface. Knowing the libraries best suited for certain purposes will improve the quality and possibly time taken to design any further project that I plan.

Finally, while my trivia question format and scoring system already demonstrate how Gamification can be used to learn Python, which was always my intention, through research, I have learnt that there are more ways to use gamification to learn Python than I initially realised. While I considered adding a leaderboard at the start of the project, I wonder if adding a badge system might also be useful to allow people to share their accomplishments with others, which is another way beyond points to reward people with good Python knowledge.

Overall, while I have noticed a lot of areas that need to be fixed, could be improved and new ideas for what I could add, I still feel like I have achieved my original purpose of demonstrating how Python's libraries can be used to create applications that use gamification to help people learn. I believe that I have demonstrated and explained some of these features well and in the process have learnt a few new Python techniques myself, but I also know that there is always room for improvement and as I get better at Python, so will my ability to make more ambitious applications and more gamification techniques within them.

References / Bibliography

Adcock, K. (2018). *How to make cross references in word*. YouTube. Available at: <https://www.youtube.com/watch?v=gHqsqFNZy0> [Accessed 14 Apr 2023].

Admin (2021). *Difference Between High Level Design and Low Level Design*. [online] BYJUS. Available at: <https://byjus.com/gate/difference-between-high-level-and-low-level-design/> [Accessed 16 April 2023].

Altexsoft (2021). *Functional and Nonfunctional Requirements: Specification and Types*. [online] AltexSoft. Available at: <https://www.altexsoft.com/blog/business/functional-and-non-functional-requirements-specification-and-types/> [Accessed 13 November 2022].

Anindya (2023). *How to make a Stopwatch GUI in Python Tkinter*. [online] ThinkInfi. Available at: <https://thinkinfi.com/how-to-make-a-stopwatch-gui-in-python-tkinter/> [Accessed 23 Apr. 2023].

Arora, S. (2019). *Why Learn Python? Reasons and Benefits of Learning Python*. [online] Simplilearn.com. Available at: <https://www.simplilearn.com/why-learn-python-a-guide-to-unlock-your-python-career-article> [Accessed 18 March 2023].

Bi Worldwide (2022). *What is gamification?* [online] BI WORLDWIDE. Available at: <https://www.biworldwide.com/gamification/what-is-gamification/> [Accessed 31 January 2023].

Codecademy. (2023). *Python Fundamentals: Python Dictionaries Cheatsheet*. [online] Available at: <https://www.codecademy.com/learn/dscp-python-fundamentals/modules/dscp-python-dictionaries/cheatsheet> [Accessed 7 Apr. 2023].

Code Review Stack Exchange. (2019). *List all possible permutations from a python dictionary of lists*. [online] Available at: <https://codereview.stackexchange.com/questions/171173/list-all-possible-permutations-from-a-python-dictionary-of-lists> [Accessed 7 Apr. 2023].

Code Review Stack Exchange. (2019). *performance - Python tkinter bouncing ball animation*. [online] Available at: <https://codereview.stackexchange.com/questions/175813/python-tkinter-bouncing-ball-animation> [Accessed 19 Feb. 2023].

D. Basten, "Gamification," in IEEE Software, vol. 34, no. 5, pp. 76-81, 2017, doi: 10.1109/MS.2017.3571581.

De Paz, A.R., 2018. Tkinter GUI Application Development Cookbook: A practical solution to your GUI development problems with Python and Tkinter. Packt Publishing Ltd.

GeeksforGeeks. (2016). *Permutation and Combination in Python*. [online] Available at: <https://www.geeksforgeeks.org/permutation-and-combination-in-python/> [Accessed 5 Apr. 2023].

GeeksforGeeks. (2019). *Python / after method in Tkinter*. [online] Available at: <https://www.geeksforgeeks.org/python-after-method-in-tkinter/>. [Accessed 19 February 2023]

GeeksforGeeks. (2020). *Create Countdown Timer using Python-Tkinter*. [online] Available at: <https://www.geeksforgeeks.org/create-countdown-timer-using-python-tkinter/> . [Accessed 1 March 2023].

GeeksforGeeks. (2020). *Python Tkinter - Entry Widget*. [online] Available at: <https://www.geeksforgeeks.org/python-tkinter-entry-widget/>. [Accessed 8 March 2023].

GeeksforGeeks. (2022). *Select Random Element from Set in Python*. [online] Available at: <https://www.geeksforgeeks.org/select-random-element-from-set-in-python/> [Accessed 22 March. 2023].

Growth Engineering (2018). *What is the Definition of Gamification and What Does it Mean?* [online] Growth Engineering. Available at: <https://www.growthengineering.co.uk/definition-of-gamification/> [Accessed 31 January 2023].

Kaufmann, D.A. (2018). Reflection: Benefits of Gamification in Online Higher Education. Journal of Instructional Research, [online] 7, pp.125–132. Available at: <https://eric.ed.gov/?id=EJ1188367> [Accessed 16 April 2023].

Lin, J. W., 2012: Why Python Is the Next Wave in Earth Sciences Computing. Bull. Amer. Meteor. Soc., 93, 1823–1824, <https://doi.org/10.1175/BAMS-D-12-00148.1> [Accessed 16 April 2023].

Mulkeen, D. (2021). *The Top 5 Benefits of Gamification in Learning*. [online] Learnlight. Available at: <https://www.learnlight.com/en/articles/5-benefits-of-gamification-in-learning/> [Accessed 19 March 2023].

Peek, S. (2013). *What Is Agile Scrum Methodology?* [online] Business News Daily. Available at: <https://www.businessnewsdaily.com/4987-what-is-agile-scrum-methodology.html> [Accessed 20 March 2023].

Pynative. (2018). *Python random.choice() to choose random element from list, String, array*. [online] Available at: <https://pynative.com/python-random-choice/>. [Accessed 19 February 2023]

Scrum.org. (2023). *What is Scrum?* [online] Available at: <https://www.scrum.org/learning-series/what-is-scrum> [Accessed 21 March 2023].

Second Street. (2023). *5 Tips for Building Awesome Trivia Quizzes*. [online] Available at: <https://uplandsoftware.com/secondstreet/resources/blog/building-trivia-quizzes/> [Accessed 31 Jan. 2023].

Simons, Chris. and Jayatilake, Dilshan. (2022) Lifecycles for Large Scale Software Development [PowerPoint presentation]. Available at: https://blackboard.uwe.ac.uk/uwenav/ultra/courses/351106_1/cl/outline [Accessed 20 March 2023].

Stack Overflow. (2017). *python - Basic Tkinter countdown timer*. [online] Available at: <https://stackoverflow.com/questions/34029223/basic-tkinter-countdown-timer> [Accessed 23 April 2023].

Stack Overflow. (2018). *python - Why it is returning PY_VAR0 instead of a number*. [online] Available at: <https://stackoverflow.com/questions/50427863/why-it-is-returning-py-var0-instead-of-a-number> [Accessed 1 March. 2023].

Stack Overflow. (2022). *list - random.choice from set? python*. [online] Available at: <https://stackoverflow.com/questions/15837729/random-choice-from-set-python> [Accessed 6 March. 2023].

Stack Overflow. (2022). *python - How would I access variables from one class to another?* [online] Available at: <https://stackoverflow.com/questions/19993795/how-would-i-access-variables-from-one-class-to-another> [Accessed 22 March 2023].

Stack Overflow. (2022). *python - In Tkinter is there any way to make a widget invisible?* [online] Available at: <https://stackoverflow.com/questions/3819354/in-tkinter-is-there-any-way-to-make-a-widget-invisible> [Accessed 13 April 2023].

Stack Overflow. (2022). *python - How do I generate all permutations of a list?* [online] Available at: <https://stackoverflow.com/questions/104420/how-do-i-generate-all-permutations-of-a-list> [Accessed 29 Mar. 2023].

Stack Overflow. (2023). *python - How to get a random value from dictionary?* [online] Available at: <https://stackoverflow.com/questions/4859292/how-to-get-a-random-value-from-dictionary> [Accessed 6 March 2023].

support.microsoft.com. (2023). *Create a cross-reference - Microsoft Support.* [online] Available at: <https://support.microsoft.com/en-us/office/create-a-cross-reference-300b208c-e45a-487a-880b-a02767d9774b> [Accessed 14 April 2023].

TechVidvan Team (2019). *Python Advantages and Disadvantages - Step in the right direction - TechVidvan.* [online] TechVidvan. Available at: <https://techvidvan.com/tutorials/python-advantages-and-disadvantages/> [Accessed 18 March 2023].

VIDUKA, D., KRAGULJAC, V. and LIČINA, B., 2021. A COMPARATIVE ANALYSIS OF THE BENEFITS OF PYTHON AND JAVA FOR BEGINNERS. *Quaestus*, (19), pp. 318-327.

Visual Paradigm (2019). *What is State Machine Diagram?* [online] *Visual-paradigm.com*. Available at: <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-state-machine-diagram/> [Accessed 16 April 2023].

Voloshina, A. (2020). *The 6 Best Software Apps for Running Your Own Trivia Night.* [online] *Trivia Bliss*. Available at: <https://triviabliss.com/trivia-night-software/> [Accessed 5 Apr. 2023].

W3schools.com. (2018). *Python Dictionaries.* [online] Available at: https://www.w3schools.com/python/python_dictionaries.asp [Accessed 6 March. 2023].

www.programiz.com. (2023). *Python Set (With Examples).* [online] Available at: <https://www.programiz.com/python-programming/set> [Accessed 22 March 2023].

www.tutorialspoint.com. (2021). *How to show and hide widgets in Tkinter.* [online] Available at: <https://www.tutorialspoint.com/how-to-show-and-hide-widgets-in-tkinter> [Accessed 13 Apr. 2023].

www.tutorialspoint.com. (2021). *How to create a timer using tkinter?* [online] Available at: <https://www.tutorialspoint.com/how-to-create-a-timer-using-tkinter> [Accessed 1 March. 2023].

www.tutorialsteacher.com. (2022). *Create UI using Tkinter in Python.* [online] Available at: <https://www.tutorialsteacher.com/python/create-gui-using-tkinter-python> [Accessed 1 November. 2022].

Zietek, T. (2016). *12 gamification platforms that help learn coding.* [online] *Medium*. Available at: https://medium.com/@tom_z_official/12-gamification-platforms-that-help-learn-coding-814aeb04341e [Accessed 4 April 2023].