Enhancing an Existing Python Learning Tool to Make It More First-Year Friendly

Mufhulufheli Mabilo
Department Of Computer Science
University of Cape Town
Cape Town, Western Cape
mblmuf001@myuct.ac.za

Siviwe Qolohle
Department Of Computer Science
University of Cape Town
Cape Town, Western Cape
qlhsiv001@myuct.ac.za

ABSTRACT

The demand for programming skills in the workplace is swiftly increasing, however the supply is growing at a much slower pace. Part of the reason is that first-year university students are struggling in their introductory programming courses, resulting in them feeling demotivated and eventually deregistering from the courses. Tools do exist which assist students in growing their skills, these tools may however not be simple enough for a first year. This proposal suggests ways in which Online Python Tutor, an existing Python learning tool, can be improved to be more beginner friendly and interactive. It also discusses how these enhancements can be implemented. The proposed enhancements are an audio code explanation feature, reduced context-switching and enhanced visualization and interactivity.

KEYWORDS

Context Switching, Visualization, Audio Generation, Interactivity

1 PROJECT DESCRIPTION

Computing is a practice that is rapidly growing worldwide, and coding is a skill that is being incorporated into many industries [8]. It is becoming crucial for all individuals (not just those in STEM) to possess the skill [8]. Seeing that the demand for coding skills in the workplace is increasing, how educational institutions approach the teaching of the subject is critical.

Many first years are arriving at tertiary institutions without any prior coding experience. Seeing that coding requires a different way of thinking and is not like any of the subjects first year students did in high school, adapting to the course is extremely challenging. First year students are failing to grasp the fundamentals of programming [6]. These fundamentals include loops, if-statements and understanding the concept of variables. As a result, the drop-out and failure rates for computer science students doing introductory computer science courses are extremely high [2]. Introductory programming, therefore, plays a critical role in determining whether students take part in learning the skill of coding. Steps need to be taken to ensure that students with no prior coding experience have access to platforms which assist them in fully grasping the basics of the skill. Platforms do exist which thrive to provide beginner programmers with an opportunity to master the basics of Python by allowing them to step through their code and see the effects of each line of code on the overall program. These platforms may however not be basic enough for first year students. Besides providing error messages and showing the user the effects of each line of code, the platforms have no other features which assist students in getting a better understanding of their code.

The aim of this project is to investigate the effectiveness and usability of an introductory python learning application that consists of an audio feature, reduced context switching and enhanced visual and interactive features. This project will mainly focus on improving the usability and effectiveness of Online Python Tutor, focusing on loops, if-statements, and assignment of variables.

2 PROBLEM STATEMENT

Computing Tools which provide users with the opportunity to improve their Python skills by providing a platform for users to step through their code and see the effects of each line do exist, they, however, may not be efficient enough to teach a beginner programmer how to code. The specific tool the project will be adding to is Online Python Tutor which is open source and incorporates a simplified and visual debugger [7] (Please see Appendix D). Online Python Tutor is a web application that allows a user to enter a piece of code and provides them with a visual representation of that code. As each line of code is executed, the effect of each line is displayed in the code visualization tool. The output of the code is also provided as well as error messages when an error is detected. The application has not been used at the University of Cape Town; however, its success can be seen by the fact that it is utilized in top universities such as MIT, University of Washington, and UC Berkeley to teach first year students Computer Science [7]. In 2013 it was reported that over 30,000 individuals used Online Python Tutor per month [7]. The web application has been included in 3 web based digital Python textbook projects [7]. The learning tool may,

however, not be as user-friendly for a beginner programmer. The tool only caters to the visual and the reading/writing learning styles. In both the code visualization tool and the error messages, complex terminology, which beginners may not understand, is used. Although Online Python Tutor does provide a visual debugger, it is not as clear to understand for students as it only highlights the current line the program is interpreting but not the variables being changed and/or referred to.

We propose the following features be added to the Online Python Tutor web application to make it more usable and beneficial for beginner programmers:

1. **Enhanced Visualization**. Incorporating visualization into the teaching of programming helps students graphically understand how a program works [9]. It helps them see how objects are affected by sequences of instructions [4]. Proposed visualization features are:

    1.1. A simplified code visualisation tool, fully showing how the frames interact with the objects. The code visualisation tool will also highlight the variables and objects being affected by each line of code (as it is executed).

    1.2. Unlike with Online Python Tutor where the entire line is outlined in red where there is an error, the specific error in the line will be highlighted. The aim of this is to assist the user in quickly finding their error.

    1.3. Explanations will be given when the user hovers over a specific section. For example, in the simplified code visualisation tool, when the user hovers over the word 'position' (of a created list), an explanation of what the word represents will be given.

2. **Enhanced Interactivity.** Interactivity allows students to interact with the code and gives them a deeper understanding of how the code operates. Interactivity also often tests whether students have fully understood the code [3]. The proposed interactivity feature is the introduction of a testing feature. To test whether the student has fully grasped the specific coding topic, they will be given a piece of code where the values of variables will change as the code goes and at different lines, the user will be asked to give the updated value of a specific variable. There will be tests for the three fundamental coding topics the project will be focusing on (if-statements, loops, and assignment of variables).

3. **Audio Explanations.** Not all students are visual learners or learn best via reading or writing [11]. Online Python Tutor currently only makes use of the above stated learning styles. The inclusion of another method of teaching the content allows the platform to cater to more students and their learning styles. An aim of this project is to cater to another learning style. This specific style is the auditory style, where an individual is more receptive to content once they hear it [11]. The proposed feature is therefore an audio feature where more comprehensive explanations are given for the most common error messages and for code that students may be unable to understand. The explained code includes loops, if-statements and assignment of variables.

4. **Reduced Context Switching**. Context switching is the alternating between two interfaces when completing a task. This can have a negative effect on the learning process as it results in a large cognitive load [1]. When it comes to coding, a reduced cognitive load reduces distractions and allows one to fully focus on the code. This reduces the amount of time it takes for one to fully understand the code. The feature that will be incorporated to reduce context switching is an additional window in the interface consisting of content explaining the coding concepts we will be focusing on, as well as tasks students will need to solve using code.

The aims for the project are:

1. Improve Visualisation features of Online Python Tutor.
2. Create an interactive testing mode for the web application which provides students with a code and tests their understanding of that code.
3. Create an audio feature which explains common errors as well as loops, if-statements, and assignment of variables using audio.
4. Create a content window which provides explanations for the coding topics the project will be focusing on, as well as tasks students will need to complete using code.

3 PROCEDURES AND METHODS

3.1 Software Architecture

The platform will be implemented in three-layered architecture. Please see Appendix C for a diagram of the Architecture.

The following are the main modules that will make up the web application. The Parser, Interpreter and Trace Generator Modules used will be the ones used in the Online Python Tutor web application.

3.1.1 Frontend. The frontend will be implemented using HTML, CSS, and JavaScript. The frontend provides a user-friendly interface for submitting python code, viewing the execution code, voice step through of code, and interacting with the visualization. The frontend also includes features like code highlighting, zooming, and panning which improves user experience and reduces context switching.

3.1.2 Backend. The backend will be implemented using Python and it will provide a Rest API for communication with the frontend. The backend will include modules for parsing, interpreting, and generating the execution trace of Python code. It

will also include a database layer for storing user data and session information.

3.1.3 Parser. The parser module will be responsible for converting the user's Python code into an abstract syntax tree. It will be done using the python built-in AST module, which provides a simple interface for parsing Python code.

3.1.4 Interpreter. The interpreter module will be responsible for simulating the execution of the Python code and generating the execution trace. The interpreter will be implemented as a stack-based interpreter, which will be maintaining a stack of frames that will be representing the call stack of the program.

3.1.5 Trace Generator. The trace generator will be mainly responsible for processing the list of events generated by the interpreter and it will be converting them into a format that can be displayed in the visualization. Secondly it will also perform additional processing, such as identifying loops and conditionals in the code and will be highlighting them in the visualization.

3.1.6 Visualization Engine. The visualization engine will be responsible for displaying the execution trace in an interactive animation. The visualization engine however will be implemented using HTML, CSS, and JavaScript similar with front-end and the features provides includes highlighting, zooming, and panning.

3.1.7 Audio feature. This module will receive Python code as input, and it will use a Python parser to generate a tree-like structure that will represent syntax and semantics. The parser that will be used will be the built-in AST module in python, which is capable of parsing python code and generating an AST that represents the structure of the code. With the AST we will need to convert it to human-readable format using a audio engine.

The way in which the audio feature will work is that there will be a specific sentence structure for if-statements, loops, and assignment of variables. Each sentence will consist of constants, which are the words that will stay the same regardless of the sentence, and variables, which are words that change according to the specific sentence. For example, if a line had the following: "Apple = 9". The explanation would be, "The variable Apple is equal to 9". In this example, the constants are "The variable" and "is equal to" and the variables are "Apple" and "9". When it comes to error messages, the most common errors will have audio explanations. The structure will be like the structure explained above for the loops, if-statements, and assignment of variables. An example of an error message for an undefined variable is: "the value 'm' does not exist, please first assign a value to 'm'". An example of an error message for 'index out of bounds' is: "the array or string is of length x, meaning positions go up to x-1. This means position y (entered position) does not exist.".

3.1.8 Database. The database module will be developed using a modern database management system which will be MySQL and a custom ORM (Object-Relational Mapping) layer will also provide a simple and consistent interface for interacting with the database.

3.2 Implementation Strategy

An agile development plan will be followed. This means that we will constantly be following the cycle of designing, then developing then evaluating. In the early iterations, the developing stage will consist of prototype design. In each stage the prototype will develop eventually going from being low fidelity to high-fidelity. In the evaluation stage of the early iterations, discount usability techniques will be used rather than participants. This is to save time and reduce the number of times participants are recruited. The discount usability technique which will be used in the early stages is heuristic evaluation. This process does not require participants and can simply be completed by comparing the product to the Ten Usability Heuristics.

3.3 Participants

Participants will be recruited for the testing of the web application. The participants will be students from the University of Cape Town enrolled in either CSC1011H or CSC1010H. 20 participants will be recruited, and they will be recruited from our supervisor's lectures.

3.4 Evaluation

We will evaluate the system by making use of both qualitative and quantitative methods. The evaluations will focus on determining the usability and effectiveness of the four different features. To determine the usability, qualitative methods will be used. For each of the four features, individuals will be given tasks to complete. The users will be observed as they navigate through that specific feature. They will be given no assistance as they navigate through the application. For example, with the audio feature, the user will be asked to play the audio for a specific code or error. The time the user takes to complete the task, the errors made and the questions they ask will all be noted. To determine the effectiveness of each of the features, quantitative testing methods will be used. A survey will be conducted consisting of four sections. Each section will focus on each feature. The survey will consist of a Likert Scale (ranging from 1 to 5) and the users will be asked to rate the features according to specific topics. For example, the user will be asked to rate the helpfulness of each feature.

4 ANTICIPATED OUTCOMES

4.1 System

The completed system will consist of a web application, which will be deployed on the web. The web application will consist of an audio feature which explains errors and the above stated coding topics; a visualization feature which makes the interface more beginner-friendly; a content window which reduces context switching and an interactivity feature which allows users to test their understanding of the topics.

4.2 Expected impact

The system provides a visual representation of Python code execution. The system will be able to help first year computer

science students to better understand how code works, which leads to better code quality and fewer errors.

The system's accessibility and ease-of-use will make it an attractive tool for students. The overall expected impact of the system is to assist the first years in improving their Python coding skills.

4.3 Key success factors

- *Beginner friendly interface*. The system will have an intuitive and easy-to-use interface, consisting of beginner friendly visualizations which assist students in grasping the basics of coding.
- *Successful implementation of new interactive feature*. The system's new interactive feature aims at allowing students to test their understanding of the specified fundamental topics in coding.
- *Successful implementation of audio feature*: feature should provide an audio description of common error messages as well as the three coding topics the project focuses on.
- *Effectiveness and usability of features:* from the evaluations, all four features need to be considered usable and effective.
- *Web-based platform*. The system will require no installation or setup, making it accessible to every student.

5 ETHICAL, PROFESSIONAL AND LEGAL ISSUES

The participants in the study will be Computer Science students from the University of Cape Town. The students will either be enrolled in CSC1010H or CSC1011H. Seeing that all the participants are from UCT, ethical clearance from UCT will be required prior to testing the application with the participants [12]. Our request to use these students as participants will need to be reviewed by the Science Faculty Ethics subcommittee, followed by Student Affairs. Identifiable participant information will not be released, and the tests will cause no physical harm to the participants. All research and software produced because of this project will be the Intellectual Property of UCT and code will be open source [13].

6 RELATED WORK

Many Python learning tools have been created to serve as a supplementary platform for individuals to improve their programming skills. These tools (including Online Python Tutor) are, however, not basic enough for beginners. They do on the other hand give ideas on how to approach creating a supplementary learning tool for beginners.

UUhistle is a Python learning tool that focuses on using visualization and interactivity to improve one's understanding of a piece of code [5]. The program has a predominantly visual mode which is referred to as the Controlled Viewing mode. The program also has a more interactive mode that is referred to as the VPS (Visual Program Simulation) mode. The VPS mode allows the user to take on the role of the computer by, for example, making the changes in the debugger as the code is stepped through line by line. The Controlled Viewing mode can basically be seen as a more complicated version of Online Python Tutor with extra features on the debugging side of the program.

Improv is a teaching tool that allows Live-Coding and reduces context switching [1]. Multiple features can be displayed on one page. For example, on one interface (similar to a PowerPoint slide) one can find the code, the terminal and the set of slides the educator will be using to teach a certain topic.

TigerJython is a Python environment that helps learners recover from errors by providing users with comprehensive error messages as well as a suggestion on how to correct the error [10].

This project aims to incorporate into the final product, one of the goals of Improv which is to reduce context switching. Instead of providing users with comprehensive error messages and potential solutions as TigerJython does, this project will focus on highlighting the specific error in the user's code.

6 PROJECT PLAN

6.1 Risks

Please see the Risk Table in Appendix A. The probability and potential impact of the risk are rated out of 10. How the potential issue will be monitored (Monitoring) and prevented (Mitigation) is included. Should the risk occur, how it will be managed is also included (Management).

6.2 Timeline

Please see the Gantt Chart in Appendix B

6.3 Resources Required

6.3.1 Software: The web application will be created using HTML, CSS, Python, React Native, Node.js, visualisation engine using iframe embedding, MySQL database, audio generation engine and Rest APIs. Participants will complete the surveys using Google Forms.

6.3.2 Hardware: Both researchers should have access to their own personal computers. The computers will be used for both the development of the web application as well as for the testing of the web application.

6.3.3 Human Resources: Participants who are enrolled in introductory programming courses at the University of Cape Town.

6.4 Deliverables

Multiple deliverables are required for the project. Below is a list of deliverables consisting of both those that are completed as well as those that still need to be completed:

- Two Literature Reviews
- Project Proposal
- Revised Project Proposal
- 2 Iterations of Prototype
- Final Paper Draft
- Project Paper Final Submission
- Project Code Final Submission
- Final Project Demonstration
- Final Web application
- Poster
- Website

6.5 Milestones

Important milestones include the submission of the final project proposal on the 5th of May, the submission of final project paper on the 11th of September and the submission of final project code on the 15th of September. Please see Gantt chart for more information.

6.6 Work Allocation

Work will be split amongst the team members in the following way: Siviwe Qolohle will focus on the Enhanced Visualization feature and the Enhanced Interactivity feature. Mufhulufheli Mabilo will focus on the Reduced Context Switching feature and the Audio feature. Should one member not be able to complete their section, the other member will still be able to continue their section and evaluate it. This is because each of the four sections can be evaluated independently.

REFERENCES

[1] Charles H. Chen and Philip J. Guo. 2019. Improv: Teaching Programming at Scale via Live Coding. In Proceedings of the Sixth (2019) ACM Conference on Learning @ Scale (L@S '19). Association for Computing Machinery, New York, NY, USA, Article 9, 1–10. https://doi-org.ezproxy.uct.ac.za/10.1145/3330430.3333627

[2] G. Silva-Maceda, P. David Arjona-Villicaña and F. Edgar CastilloBarrera, "More Time or Better Tools? A Large-Scale Retrospective Comparison of Pedagogical Approaches to Teach Programming," in IEEE Transactions on Education, vol. 59, no. 4, pp. 274-281, Nov. 2016, doi: 10.1109/TE.2016.2535207

[3] Ibrahim Cetin & Christine Andrews-Larson (2016) Learning sorting algorithms through visualization construction, Computer Science Education, 26:1, 27-43, DOI: 10.1080/08993408.2016.1160664

[4] Imre BENDE. 2022. Data Visualization in Programming Education. Acta Didactica Napocensia 15, 1 (2022), 52-60. DOI: https://doi.org/10.24193/adn.15.1.5

[5] Juha Sorva and Teemu Sirkiä. 2010. UUhistle: a software tool for visual program simulation. In Proceedings of the 10th Koli Calling International Conference on Computing Education Research (Koli Calling '10). Association for Computing Machinery, New York, NY, USA, 49–54.https://doiorg.ezproxy.uct.ac.za/10.1145/1930464.1930471

[6] Michael McCracken, Vicki Almstrum, Danny Diaz, Mark Guzdial, Dianne Hagan, Yifat Ben-David Kolikant, Cary Laxer, Lynda Thomas, Ian Utting, and Tadeusz Wilusz. 2001. A multi-national, multiinstitutional study of assessment of programming skills of first-year CS students. SIGCSE Bull. 33, 4 (December 2001), 125–180. https://doiorg.ezproxy.uct.ac.za/10.1145/572139.57218

[7] Philip J. Guo. 2013. Online python tutor: embeddable web-based program visualization for cs education. In Proceeding of the 44th ACM technical symposium on Computer science education (SIGCSE '13). Association for Computing Machinery, New York, NY, USA, 579–584. https://doi-org.ezproxy.uct.ac.za/10.1145/2445196.2445368

[8] Robert P. Taylor. 1977. Teaching programming to beginners. In Proceedings of the seventh SIGCSE technical symposium on Computer science education (SIGCSE '77). Association for Computing Machinery, New York, NY, USA, 88–92. https://doi-org.ezproxy.uct.ac.za/10.1145/800104.803365

[9] Šimoňák, Slavomír..2014 "Using algorithm visualizations in computer science education" Open Computer Science, vol. 4, no. 3, pp. 183-190. https://doi.org/10.2478/s13537-014-0215-4

[10] Tobias Kohn and Bill Manaris. 2020. Tell Me What's Wrong: A Python IDE with Error Messages. In Proceedings of the 51st ACM Technical Symposium on Computer Science Education (SIGCSE '20). Association for Computing Machinery, New York, NY, USA, 1054–1060

[11] 2021. 8 Types of Learning Styles | The Definitive Guide. Bay Atlantic University. Retrieved May 5, 2023 from https://bau.edu/blog/types-of-learning-styles/

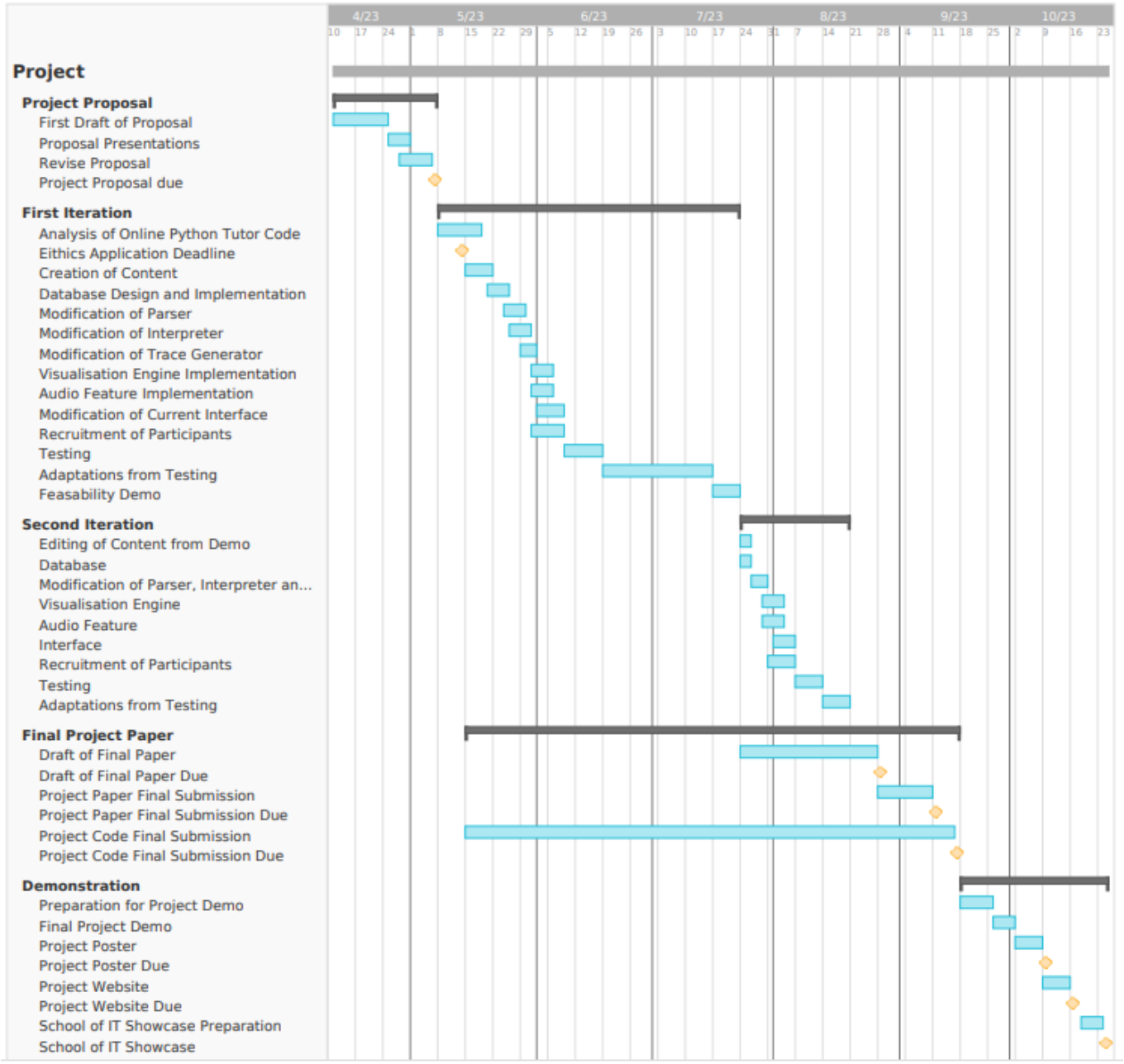
[12] UCT Computer Science Department. 2018. Applying for Ethics Clearance for a Study or Experiment. Retrieved April 19, 2023 from https://www.cs.uct.ac.za/Ethics

[13] University of Cape Town. 2011. University of Cape Town Intellectual Property Policy. Retrieved April 19, 2023 from http://www.uct.ac.za/sites/default/files/image_tool/images/328/about/policies/Policy_Intellectual_Property_2011.pdf

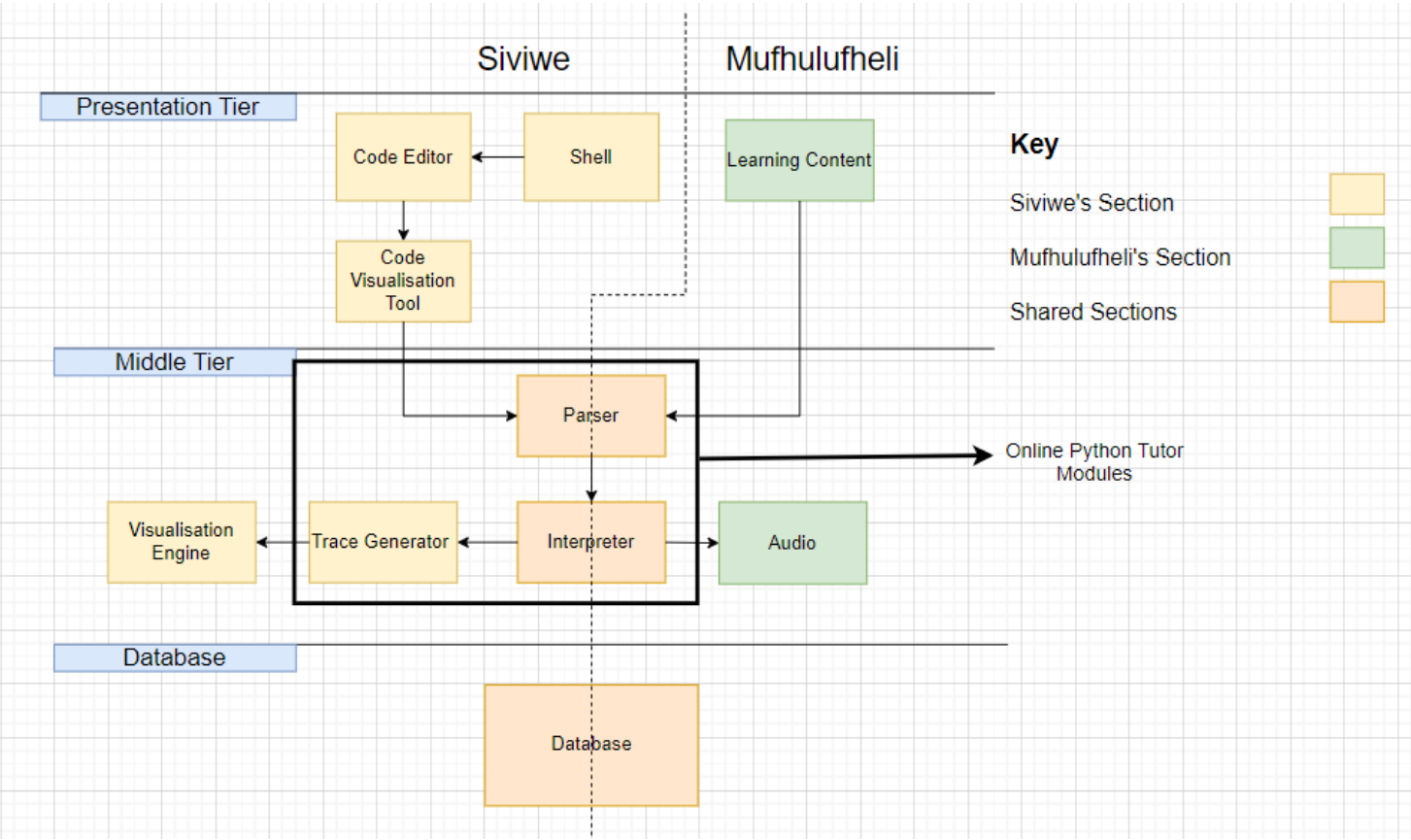
Appendix A

Risk	Proba-bility	Impact	Monitoring	Mitigation	Management
Delay in obtaining Ethical Clearance	5	10	Frequently checking emails	Submit Request as soon as possible	Reduce scope to ensure final product is completed on time
Integrating the various APIs	4	10	Frequent testing and logging	Keep APIs up-to-date and consider using API management tools	Will use modular design and document everything
Loss of Data	3	10	Check that the saved data is the most recently edited data	Ensure data is backed up	Distribute more surveys and gather new participants
Conflict between team members	2	6	Frequent checkup sessions	Maintain good communication between group members	Have a session where all concerns are voiced
Not obtaining enough participants	1	8	Check how many individuals are volunteering daily	Recruit participants as soon as possible	Broaden group of potential participants (with permission from supervisor)

Appendix B


4/23
5/23
6/23
7/23
8/23
9/23
10/23
Project
Project Proposal
First Draft of Proposal
Proposal Presentations
Revise Proposal
Project Proposal due
First Iteration
Analysis of Online Python Tutor Code
Eithics Application Deadline
Creation of Content
Database Design and Implementation
Modification of Parser
Modification of Interpreter
Modification of Trace Generator
Visualisation Engine Implementation
Audio Feature Implementation
Modification of Current Interface
Recruitment of Participants
Testing
Adaptations from Testing
Feasability Demo
Second Iteration
Editing of Content from Demo
Database
Modification of Parser, Interpreter an...
Visualisation Engine
Audio Feature
Interface
Recruitment of Participants
Testing
Adaptations from Testing
Final Project Paper
Draft of Final Paper
Draft of Final Paper Due
Project Paper Final Submission
Project Paper Final Submission Due
Project Code Final Submission
Project Code Final Submission Due
Demonstration
Preparation for Project Demo
Final Project Demo
Project Poster
Project Poster Due
Project Website
Project Website Due
School of IT Showcase Preparation
School of IT Showcase

Appendix C


Siviwe
Mufhulufheli
Presentation Tier
Code Editor
Shell
Learning Content
Code Visualisation Tool
Key
Siviwe's Section
Mufhulufheli's Section
Shared Sections
Middle Tier
Parser
Online Python Tutor Modules
Visualisation Engine
Trace Generator
Interpreter
Audio
Database
Database

Appendix D

Online Python Tutor

```
1  numbers = [2,5,2,4]
2  for p in range(len(numbers)):
3     print(p)
4  print(m)
```

Edit code

<< First | < Back | Program terminated | Forward > | Last >>

NameError: name 'm' is not defined

line that has just executed | next line to execute

Program output:

```
0
1
2
3
```