Tech review

Roy Simons

Fall Term: May 26, 2020

Project: Neuromorphic for kids

Team 67: "Get Good, Kids!"

I. Project Role Roy Simons

The personal responsibilities for this project are level/lesson design, music and sound, and user interface/menu design. The majority of the project is the creation of level/lesson, which must be well designed. If the project contains poor level/lesson design, it will lose appeal to the users. The second responsibility, music and sound, is of great importance for the project, to give the project a complete feel. Music and sound, will be used in the project to create more engagement and appeal for the user, which makes it important to find and/or create the perfect sounds. The last personal responsibility is the interface and menu design, which is the front of the project. Menu design is a make or break for a user of the project. The menu must be inviting, appealing, easy of use, and quick to understand, if these requirements are not met there is a chance to lose users. User interface must also meet the requirements of being appealing, easy to use, and quick to understand. Users must have as little difficulty with the use of the project as possible.

II. Project Objective

To teach Native American high school students basic coding skills and computer science concepts, including upcoming technologies like neuromorphic computing. The goal of teaching Native American high school students these skills is to increase their interest and chances of pursuing an education and career in computer science and tech. The system will supplement Native American high school student Computer Science education, with lessons / challenges that will engage them and gradually increase in difficulty. The lessons/challenges will not only increase in difficulty but also in the amount of code that has to be written to complete the lesson. The system will hold the data of users, which will mainly be their progress. The system should be able to be used in public high schools that have limited access to technology. The program should be able to run on older systems.The system should check the inputs of the users, depending on the correctness of the answer, the user will receive a score as an output. Each score belongs to a lesson the user has attempted. Primary users of this system will be Native American high school students living on reservations who do not have access to quality technical education or the resources to teach them. The users can have experience with some tech devices, like mobile phones, but it is expected that most users have limited availability to tech.

III. Level/Lesson design

Objective: The project mainly consist out of level/lessons, which makes the design of the levels one of the most important parts of the project. Each level must be designed with certain standards and requirements.

Goal: The main requirements for each level is that it must look appealing. The reason for the lesson to be appealing, is to keep the user engaged and interested with the material of the lesson.

The second requirement for the lessons is that they must be self explanatory. The user must quickly understand what the lesson wants the user to do, which can be achieved through clear explanation, visuals and if needed a help function. The lesson must have a clear section for the user to fill out their answer, which will depend on the difficulty and design of the lesson.

Another requirement is to have the lessons engaging, for which the plan is to combine the results of multiple lessons, and the created code of the users, to be part of one big result. A big "end" result of a group lessons, will lead to a greater feel of accomplishment for the user, helps with concept understanding of the material taught in the lesson, and to see a bigger picture than just little snippets of code.

Options: The lessons should be designed to fit a phone screen, which would be the smallest format. The technology for which the lessons will be designed should be expected to be older models, which makes it of great importance to test how the lessons perform on older mobile phone devices.

The second option is that the lessons should be designed to fit a raspberry pi or an Intel NUC. The Raspberry Pi is a low cost, credit-card sized computer that plugs into a computer monitor or TV, and uses a standard keyboard and mouse[1]. It is a capable little device that enables people of all ages to explore computing, and to learn how to program in languages like Scratch and Python [1].

Compare: The project is made for Native American high school students that don't have a lot of technology, but most people do have a phone. It would be a good thing to develop a phone application, because it will be more likely to reach more students. A downside for the phone application is that it is not an convenient way to code with, an actual keyboard and monitor is a lot nicer.

The Raspberry Pi or Intel NUC, are also great tools to use when creating an application, but the downside is that most high schools do not have access to these devices. The Raspberry Pi, or Intel NUC must be provided to be able to be used by the high school students.

Selection: The end goal is to deliver lessons that are complete, by which the lesson must be of a correct difficulty, great appeal, self explanatory, engaging, and fit for the expected technology. The more complete lessons created, the better it is for the project, but only the complete lessons should be a part of the end product. Using both options, phone and Raspberry Pi or Intel NUC, will lead probably to the best result. Start the simpler lessons on the phone application, and have the harder more code intensive lessons on the Raspberry Pi or Intel NUC.

IV. Music and Sound

Objective: The project must contain music and sound to improve the user experience. The music and sound are background music, conformation sound, etc. which will help the program stand out more than a soundless version. The music and sound should have a neutral theme, but allow for future options to give it a specific style.

Goal: Music and sound is an underestimated part of a project, but it can do a lot of good for any project. The first application of sound in the project would be little tunes/sound bits for completion of a lesson. These sound bits will also be used when calculating and presenting the score. Another use for the sound bits is to have a certain sound when clicking on a specific button, which can be seen as a verification sound for the user that the button has been clicked.

The project will use music, which must be simple but appealing background music. It is the goal that the music will appeal to the user and give a sense familiarity when using the program. If the background music is poor, it has the chance to irritate the users, which will keep them from using the program. Picking a fitting background, must be well tested, to assure that it fits the program and does not lead to irritation.

An important feature for music and sound, is that the user must always have the option to turn all sound options on or off. A user must not be forced to listen to music and sound, when using the program.

Options: The implementation of the music and sound will be done with music files. The implementation should not need any special tech, but creation of the sounds can be more difficult. The ideal situation is to find suiting music and sound files on the internet, but if this cannot be achieved technology is needed to record and create sound files.

Compare: Taking the music and sounds from the internet or making themselves, will lead to the same result. The only

difference is that the internet will more likely have a larger option of music and sound, and will more likely be cheaper. Making the music and sound, will allow to create a more specific style of sound, but the project requires a neutral theme.
Selection: The best option will be to find free music and sound on the internet, since the project theme is to be neutral sound and music. If no fitting sound is found on the internet, music and sound has to be created, which can also be applied to certain users wanting a specific style for the program.

V. User Interface and Menu Design

Objective: The menu is the face of the app, it is the first thing the user sees when the program is started. The menu must give the user a familiar feeling, each time the user starts the program.
Goal: The menu must appeal to the user, which should be achieved by a clean and simple design. If the menu appears to busy, to vague, or visually unpleasing, it gives a reason to stop wanting to use the program.
When a user clicks on a button, the button must make an effect to show a visual confirmation, which will be followed by the actual requested part of the program. The exit should be in the spot in the menu and the user interface, allowing the user to exit the lesson or program when wanted, a conformation pop-up should appear to confirm the exit request.
Options: The first option would be to use Reactjs, mentioned by the client, which is a javascript library for bulding user interfaces. React makes it painless to create interactive UIs [2]. Design simple views for each state in your application, and React will efficiently update and render just the right components when your data changes [2]. There are more options for user interface design, like Inferno JS, Preact, React-lite, Backbone JS, Aurelia, Ember JS, Riot JS and Mithril, which are all the same idea as Reactjs.
Compare: It is hard to compare which one is better for the project, since all options are very similar. Reactjs was mentioned by the client, which makes it stand out from the other options
Selection: Reactjs will be used for the project, but if the other options turn out to be more fitting for the project a switch will be made. For the user interface and menu design a final decision is hard to make, but the first choice for now is to use Reactjs.

VI. References

[1] "What is a Raspberry Pi?" Available at https://www.raspberrypi.org/help/what-%20is-a-raspberry-pi/

[2] "React A JavaScript library for building user interfaces" Available at https://reactjs.org/