CS CAPSTONE DESIGN DOCUMENT

MAY 26, 2020

PROGRAMMING FOR UNDERREPRESENTED HIGH SCHOOL STUDENTS

PREPARED FOR

INTEL, INC

SHASHI JAIN _____ _____
Signature *Date*

PREPARED BY

GROUP 67

GET GOOD, KIDS!

ISAK FOSHAY _____ _____
Signature *Date*

REBECCA CROYSDALE _____ _____
Signature *Date*

CRISTIAN MANN _____ _____
Signature *Date*

ROY SIMONS _____ _____
Signature *Date*

Abstract

A design document detailing how each component will be built and used, relating to teaching kids programming. This app intends to inspire underprivileged high school students to seek an education and career in computer science through a set of computer science lessons. These lessons will be served from an Intel NUC with an NCS2 for machine learning lessons. Students will access the lessons with a mobile device on a web page served by the NUC.

CONTENTS

1 OVERVIEW

1.1 Scope

This document describes the software component relations, organization, and functions in the Programming for Underrepresented High School Students project. The document will present how each of the app's components will interact and depend on one another, as well as high-level descriptions on how each element will operate.

1.2 Purpose

The purpose of the project described is to teach underrepresented high school students through an appealing and rewarding approach. After completing the lessons, the intended audience should want to learn more computer science concepts and to pursue an education and career in the field of computer science.

1.3 Intended Audience

The intended audience of the project described is high school students that are underprivileged in terms of access to technology. These are students of high schools that have little to no access to computers, internet or computer science classes.

1.4 Stakeholders

The stakeholders of this project include Shashi Jain of Intel Corp, underrepresented high school students, and the teachers of those students.

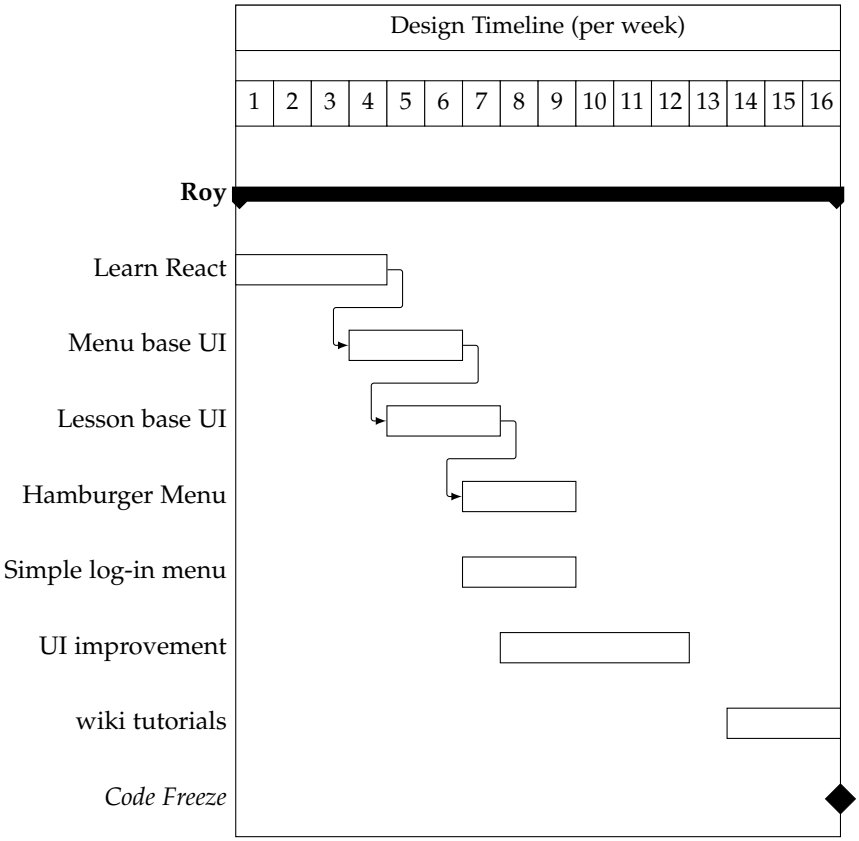
1.5 Conformance

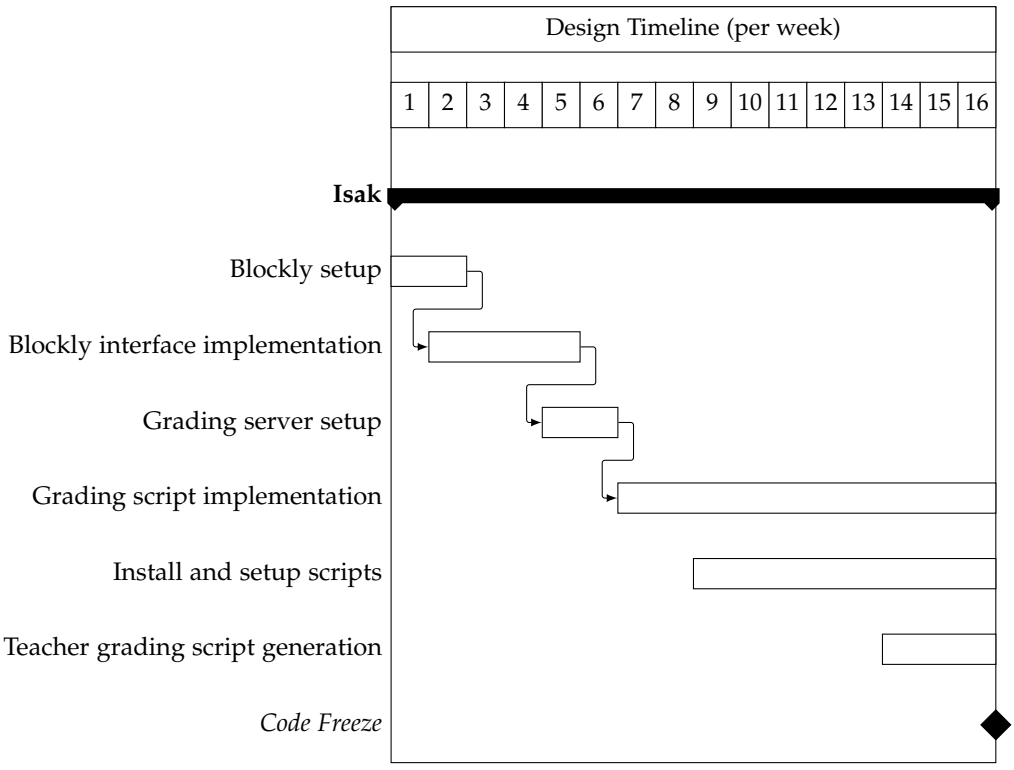
This document conforms with the requirements and specifications agreed upon thus far by the client and design team.

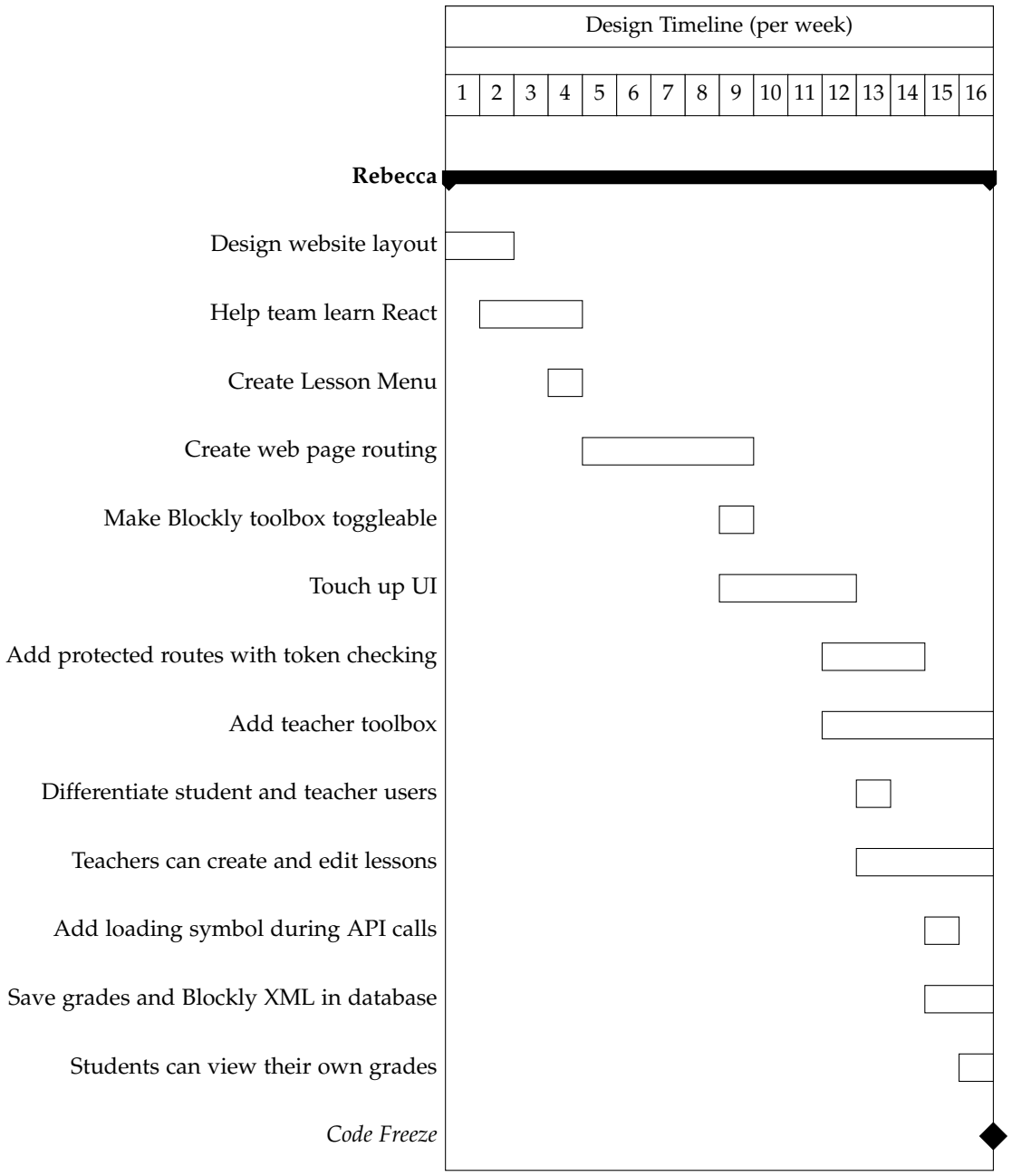
1.6 Definitions

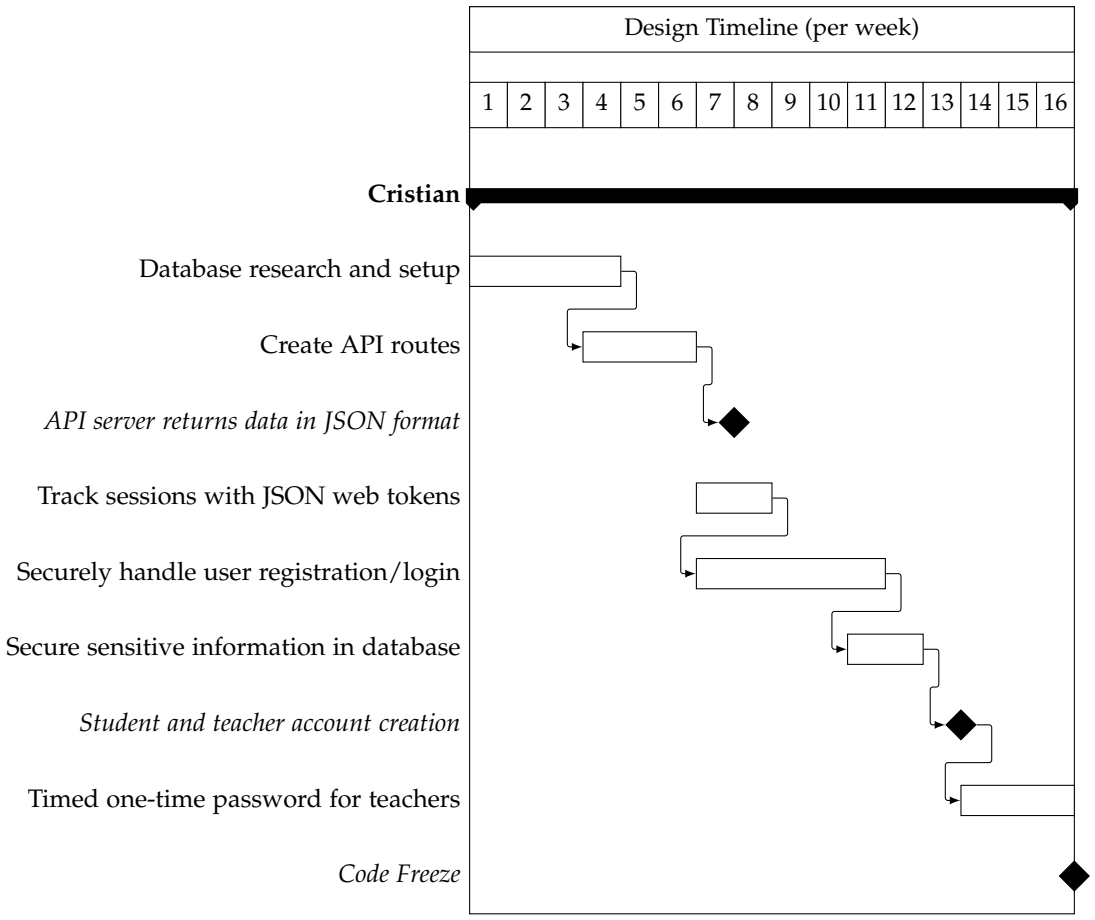
1) NCS2: The Neural Compute Stick 2 is "a toolkit that accelerates solution development and streamlines deployment" for machine learning development [**?**]
2) Intel NUC: The Intel Next Unit of Computing is a 4 inch by 4 inch mini PC.
3) Blockly: Blockly, created by Google, is a block-based visual language targeted towards children.

2 DESIGN TIMELINE




Design Timeline (per week)
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
Isak
Blockly setup
Blockly interface implementation
Grading server setup
Grading script implementation
Install and setup scripts
Teacher grading script generation
Code Freeze


Design Timeline (per week)
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
Rebecca
Design website layout
Help team learn React
Create Lesson Menu
Create web page routing
Make Blockly toolbox toggleable
Touch up UI
Add protected routes with token checking
Add teacher toolbox
Differentiate student and teacher users
Teachers can create and edit lessons
Add loading symbol during API calls
Save grades and Blockly XML in database
Students can view their own grades
Code Freeze


Design Timeline (per week)
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
Cristian
Database research and setup
Create API routes
API server returns data in JSON format
Track sessions with JSON web tokens
Securely handle user registration/login
Secure sensitive information in database
Student and teacher account creation
Timed one-time password for teachers
Code Freeze

3 Stakeholders and their concerns

3.1 Client: Shashi Jain of Intel Corp

Shashi Jain's design concerns consist of the following:

1) CS Lessons and Concepts taught: The lessons need to teach the basic concepts of computer science and lead to machine learning concepts. It is preferred to have a semi-automated lesson generator to create the lessons for the users.
2) Security of the project: The hardware should be secure to keep any proprietary information on the NUC or NCS2 secure.

3.2 User: Student

The student's design concerns consist of the following:

1) CS Lessons and Concepts taught: The concepts in the lessons must be presented and taught in a way that interests, excites, challenges, and inspires the students to want to learn more about computer science. The concept explanation should further explain what was taught in each specific lesson, to help the students understand the concepts.
2) Security of the project: Students will want their lesson progress to be secure so that they can continue to use them in future sessions.

3.3 User: Teacher

The teacher's design concerns consist of the following:

1) CS Lessons and Concepts taught: The lessons and concepts need to be understandable for the teacher, which will allow the teacher to help students that are struggling, confused or need extra help. The concept explanation should assist the teacher in helping the student work through the lessons and understand the taught concepts.
2) Lesson Creation: The lesson creation page should be easy to use and understand. Lesson creation should be implemented via Blockly. A teacher should be able to edit or delete a lesson after creating it.
3) NUC implementation: The teachers need to set up the NUC to serve the lessons to the students. The server should be easy to set up using the NUC.
4) Security of the project: If the teacher intends to use the project for grades in the class, then they will want the progress data to be secure.

4 Design Viewpoints

1) Context Viewpoint: The Context Viewpoint will examine the users and the interactions of the users with the application.
2) Composition Viewpoint: The Composition Viewpoint will define the systems, subsystems, frameworks and other components that the application uses, and how they interact.
3) Information Viewpoint: The Information Viewpoint will examine the security and storage of student's and teacher's personal data.
4) Pattern Viewpoint: The Pattern Viewpoint will provide a template for how each CS Lesson will be presented and executed.

5 Design Views

5.1 Context View

5.1.1 Design concerns

The Context Viewpoint will examine the users and the interactions of the users with the application.

5.1.2 Design elements

1) **Design entities**
    a) Student: The main user of the application
    b) Teacher: The user that guides and helps the students with the application. The user that can add, adjust and delete lesson for the student users. The user that can check the progression of the student user.
    c) Concepts: The computer science concepts that will be taught, as beginner concepts and machine learning.
    d) Lessons: The lessons used to convey each computer science topic
    e) Lesson creation: A function to create more lessons
2) **Design relationships**
    a) The teacher will teach the concepts to the students through the lessons. The teacher can add, adjust and delete lessons to the needed standards.
    b) The students have gain access to their progression by completing the lessons.
    c) Lessons will be used to supplement the teacher's efforts.
    d) Teachers will have the ability to create lessons to teach other concepts than the available ones.
3) **Design constraints**
    a) Teachers that will use the application most likely have low understanding of the taught concepts.
    b) The developers understanding and grasp of the topics being conveyed, meaning that if the developer understood a concept wrong they are now teaching that concept wrong through the lesson.

5.2 Composition View

5.2.1 Design concerns

The Composition View addresses how users will interact with the Intel NUC in order to interface with the application.

5.2.2 Design elements

1) **Design entities**
    a) Smart phone application
    b) Programming language
    c) Intel NUC
    d) School's Wifi
2) **Design relationships**
    a) The smartphone application should be designed to work for android and iOS.
    b) The programming language for the user interface will use JavaScript, specifically the React library.
    c) The Intel NUC will need to be connected to the school's WiFi by the Teacher.
    d) The Student will need to use their phone to connect to the web page served by the Intel NUC.
3) **Design constraints**
    a) The quality of the phones of the users, because the application must work for older devices.
    b) School's WiFi connection quality.
    c) Teacher's ability to understand the NUC setup.
    d) Some users might not own a mobile device that can access the web page.

5.3 Information View

5.3.1 Design concerns

The Information View will address how the database will be designed and how data will be stored and secured.

5.3.2 Design elements

1) **Design entities**

   a) Block: A block of code that a user can create and save to be used in Lessons.
   b) Device: What users will use to interact with the program.
   c) Lesson: Where users will learn about a concept and test their knowledge of it through example code blocks.
   d) Lesson Progress: The lesson progress saved after a user starts a lesson.
   e) Intel NUC: What will be used to serve and store data from the user and game.
   f) User: Includes a username, password, and first and last name.
   g) Student: The representation of the student user in the system.
   h) Teacher: The representation of the teacher user in the system.

2) **Design relationships**

   a) Blocks will be a simplified code library a Student has access to for use in Lessons.
   b) Lessons will all be available, which allows students to attempt the lesson that fits to their experience.
   c) The student user data will be stored on the NUC. This includes the lesson progress and score data as well as any custom blocks that a user makes. This user data will be secured using hashes.
   d) A user can be either a student or a teacher. The user type determines what components a user can view when logged in on the application.
   e) A student user can start a lesson and later continue from their saved lesson progress. A user does not need to restart the lesson each time the lesson page loads.

3) **Design constraints**

   a) The number of active Students can affect the response time from the NUC, among other things.
   b) Monitoring output of code and Blocks to limit security vulnerabilities like SQL injections.

5.4 Pattern View

5.4.1 Design concerns

The Pattern view outlines how each lesson will follow a similar pattern to convey the topic/concept at hand.

5.4.2 Design elements

1) **Design entities**

   a) Lesson description
   b) Blockly environment
   c) Lesson solution

2) **Design relationships**

   a) For each lesson, the user will first be presented with a description of the challenge at hand. Then, the user will be presented with a Blockly environment where they will attempt to solve the challenge by utilizing different Blockly blocks. As the user adds blocks to the Blockly environment, their progress is saved. Once the user feels they have solved the problem, they run their solution against the lessons' grading script. If their result receives a high enough score compared to the lesson solution then the lesson is complete, and they receive the corresponding grade.

3) **Design constraints**

   a) Some lessons may require special AI blocks to be used, which will alter the comparison of the user's solution compared to the lesson solution.

6 DESIGN OVERLAYS

6.1 Context View overlay

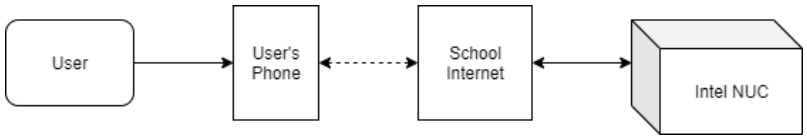


Fig. 1. Context view diagram

6.2 Pattern View overlay

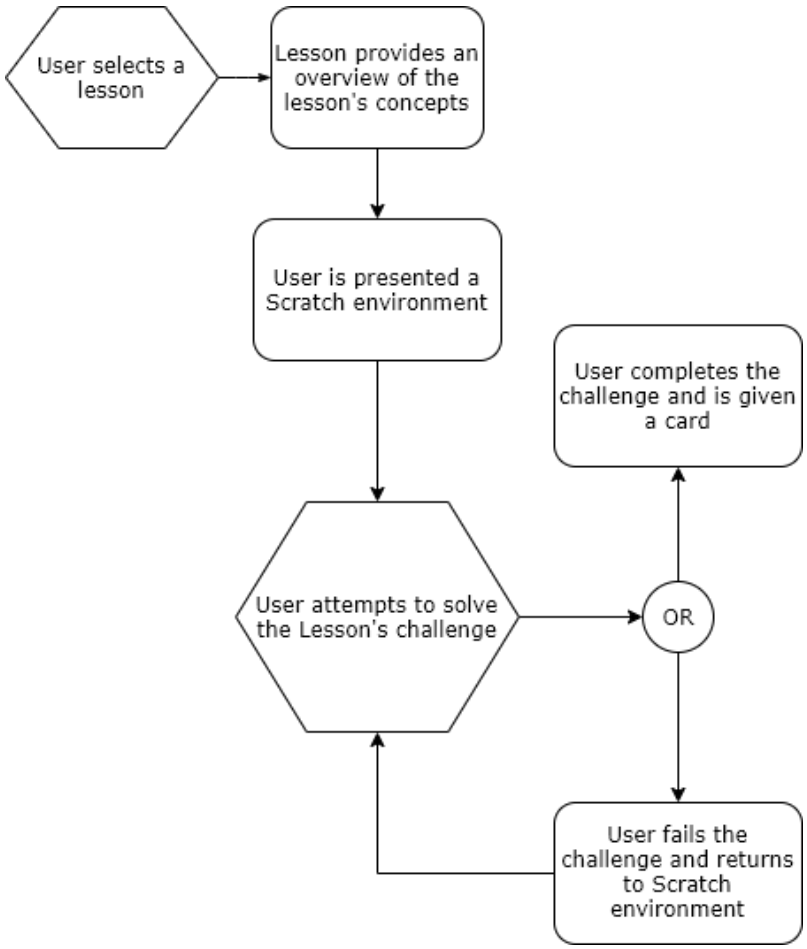


Fig. 2. Pattern view diagram

7 Design rationale

When the team was assigned the goal of teaching underrepresented high school students computer science topics via a game, we were unsure at first how we would accomplish the task. In the beginning, the team had our minds set on creating a desktop-based application, because controlling a character on the screen is much easier on the desktop compared to using the phone. Additionally, using a neural compute stick limits a user to a desktop-based environment in order to interface with the NCS2. In one of our meetings, Shashi Jain confirmed that he would like the project to be built for the phone. The team debated for a while on the best way to convey topics in a game format while also being intuitive to use on the phone. Eventually, we settled on having the game play be a card game because this results in simple, intuitive controls. Another roadblock we encountered was that nobody on our team owns a Macbook, so we were unable to develop for Apple products easily. We overcame this constraint and solved the issue of the NCS2 incorporation by settling on a web-based, NUC served application. This allows us to develop in Javascript, which will run on both Android and Apple, as well as run commands on the NCS2.

Choosing to incorporate Blockly in our design was based on the fact that it simplifies writing code down into manageable block-sized pieces. This comes with the trade-off of not being able to communicate proper syntax to the students using the system. Blockly also allows for limited customization as the blocks are predefined for the user. This also adds a layer of protection since a user has fewer tools at their disposal for possible nefarious use.

In a later meeting Shashi Jain removed the constraint of a card game based reward system. As such, we were able to focus more effort on creating the tools for a teacher to create and edit lessons as well as view their student's progress.

We chose to use REACT as our main interface and modeling library for its customization and modularization aspects.

8 Conclusion

The Programming for Underrepresented High School Students app will be served via an Intel NUC using a school's WiFi connection. The application will consist of a variety of different computer science lessons, which can be created and edited by teachers and then completed by students.

9 Change Log

- Main users went from mainly student, to student and teacher.
- Teacher is able to add, adjust and delete lessons.
- Student after completing lesson will see score in progression, instead of concepts.
- Lessons are all unlocked, instead of being unlocked by progression.
- Card game removed, instead show progression per lesson for student users.