1. Introduction
1.1. Objective

According to the Centers for Disease Control and Prevention (CDC) rates of autism have increased from 1 in 150 newborns being diagnosed with autism in 2000 to 1 in 59 in the year 2018 [1]. Stanford University’s Lucile Packard Children’s Hospital found that ASD children tend to have short attention spans [4]. Speech Language Pathologist, Beverly Vicker adds on by saying “...students on the autism spectrum appear erratic in their ability to follow directions” [5]. Stanford University’s Lucile Packard Children’s Hospital did research into how to interact with a child who has autism and found that the following methods were important to better the problem of following directions and maintaining an attention span:

- Interact through physical activity
- Always stay positive (reward system)
- Show your love and interest[4]

Yale University conducted a study in August of 2018 where they witnessed significant improvements in the social skills of children with autism after a month of working with robots[2]. Through our search we are motivated for this project is to build a therapeutic Sydekick for those who have have autism.

Our intention with SydeKick is to create a robot that is a partial humanoid; this means it will have a torso, head, and two arm-like appendages that operate on one rotational axis. Sydekick will have a set of wheels to govern its translation and rotational motion and make it easier for the child to interact. The robot will be able to play Simon Says. This game was picked to try to maintain the attention of a child through practicing following directions. We will not be solving autism, but through tackling the mini-problems of short attention spans and following directions our hope is to create a solution for children with autism to have daily therapeutic sessions focusing on these aspects.

Testing our product with an actual child on the autism spectrum will not be feasible because of the rules and regulations when dealing with robotics and children. Our team, through the advice of TA’s and advisors, has agreed not to fill out at IRB to test this with children that are actually on the spectrum. Instead, through our research, as discussed above, we have identified the following:

1. The challenges that children with autism face
2. The types of activities that can help mitigate some of these challenges

Through this information, our team will be creating Sydekick and verifying that we can accomplish these tasks. Through verifying that SydeKick can perform these tasks with anyone, our team will have created a theoretical product that can work with children on the autism spectrum.
1.2. Background

A good example on the market as of right now is a research exploration through the University of Portsmouth. The University of Portsmouth is working on developing autonomous robots through their DREAM project. Unfortunately, it is at a very nascent state. The University explains that this will have a lot of impact on children with ASD but, current social robots are simply remote-controlled by the therapists and still require lots of time and energy from humans[7]. The competitive advantage with Sydekick is the near-autonomous experience that we are trying to emulate by programming simple games and experiences. Our team hopes that by aiming for a simple design with simple games we can mitigate the larger problem through smaller issues.

1.3. High-Level Requirements

- Sydekick should be able to reward 3 right answers through our reward system over a time period of 2 minutes through a game of Simon Says with a user (TA/student)
- Sydekick should be able to stay in front of the test player (TA/student) at all times maintaining a distance of 1 foot, this includes when Sydekick moves its hands up or down
- Sydekick will not be able to drive past a certain average speed (needs to be calculated); will also only be able to lift arms from hips to shoulders (90 degrees)
2. Design

2.1. Block Diagram

Figure 1: Sydekick Block Diagram
2.2. Physical Design

Figure 2: Sydekick Physical Diagram
2.3. Block Design

2.3.1. Functional Overview

Subsystem 1 - Processing

Central Printed Circuit Board Equipped with ATMega128 Microcontroller

This is the central processor for the robot. It receives power from the battery that will described in its own section. We have chosen the ATMega128 SOC due to simplicity and versatility. The sensors, top LCD screen, servos and the wheels will be driven through this module. The Raspberry Pi will be used for computational power and memory capabilities for software endeavors.

Raspberry Pi 3

This auxiliary device will connect to the module described above and it will receive power from the battery as well. This will be used for computational power and memory density. The games and central LCD screen will be operated and interfaced with the Raspberry Pi.

The use of the Raspberry Pi is necessary as it allows for quick interfacing with external software development kits, which in this case is the Android SDK; referenced below.

Rechargeable Battery

The battery used as the power source for the robot will be, most likely, the Turnigy 5000mAh 6S 40C Lipo Pack with XT90. This battery has been chosen due to large capacity (5000 mAh) as well its power specification. Furthermore, it has rechargeability as well as a high discharge which will be necessary for robot movement and child interaction. Additional details about the power module of this circuit will be determined as the project advances.

Subsystem 2 - Programming

Central LCD Screen

8 inch LCD screen mounted on the “torso” of the robot (similar to a teletubby). It will utilize capacitive touch capabilities as to introduce touch screen functionality for games. The LCD will interface with the Raspberry Pi for power and data.

Android SDK

Will be used for the development of applications on the Raspberry Pi. We will be utilizing a lot of open source resources to leverage the Android SDK. The applications will be interfaced through the central LCD screen.

Google Play Services

Used for the creation of games within the Android applications. We’ll be leveraging past experience on building Android applications to work on developing the Custom Games (mentioned below).

Custom Games

Android games using Google Play Services and will display on the central LCD screen and run on the raspberry pi. Our team will be designing Simon Says which will incorporate pressure sensors across SydeKick and will command the child to touch different areas of the robot.
Stanford University’s Lucile Packard Children’s Hospital explains it’s important to find games that have very basic and concise commands that practice following directions and allow for interaction. Additionally, it’s also important to find a way to incorporate a reward system so the child feels accomplished[5]. With this in mind, our group tried to identify games that could align with those factors. Our group thinks that through these games we can better the attention span/direction following capabilities of children with autism.

**Movement and Dynamics Control Program**

Will be driving the wheels and arm servos with throttled speeds as per safety specifications. This will be run on the ATmega128 Microcontroller. This program will need to ensure that the wheels operate in tandem and do not erroneously choose different directions to revolve in. Since we are working with omni-directional wheels there will be two axes of rotation. The first will set the speed of the robot’s movement and the latter will set the direction. The arm servos, however, will only be operating within one rotational axis.

The speed throttling program will consist of certain safety parameters built into the robot in order to limit the speed and movement of the robot as to avoid any harm to the child. The safety specifications will be determined soon as we get more information, though we aim to make contact with a few different toy and children’s robot designers as to establish these standards.

**Facial Expression Interaction**

This module will operate on the ATmega128 Microcontroller and will effectively use various reactions to give the child a more complete and humanoid playtime experience. It will know when to smile, laugh, gasp, and also have a generic face programmed as well. The emotions will act a reward system for game completion or to make the gaming experience more friendly.

**Subsystem 3 - Sensors**

**Left and Right Pressure Sensors**

Both arms will contain pressure sensors to play games with the child. This will be interfaced through the PCB as described in Subsystem 1. This will be a key component for our high-level requirement (#1) which will allow us to take in data and derive a reward system for the child everytime he/she gets an answer command correct.

**Movement Sensor**

The direction/ability to move will be gauged through a sensor that will be able to measure proximity towards the child. This will allow the team to keep safety in mind and design proper precautions and maintain a set distance between the robot and the child. This will also be interfaced with the PCB as described in Subsystem 1. This will be important for our last high-level requirement (#3) where the movement sensor will be getting data about proximity to the child so that “Sydekick” adheres to safety.

**Subsystem 4 - Arms**

**Left and Right Arm Servos**

The servo motors will be driven through the ATmega128 Microcontroller. They will be utilized to get arm mobility for the robot across one rotational axis. We have decided to use
servos as the motor for the arms as they limit their range of moment to 360 degrees, further mitigating safety hazards.

**Subsystem 5 - Wheels**

*Four Omnidirectional Wheels*

The wheels on the motorized base will be connected through the PCB which is mentioned in Subsystem 1. These will be useful in the robot navigating its way towards the child and making sure that it is in close proximity.

**Subsystem 6 - Miscellaneous**

*Top LCD Screen*

The top LCD screen will be positioned on the head of the robot and will display a variety of emotions using a humanoid face. It will be connected to the central PCB which will use the Face Expression Interaction program to choose which face it needs to display depending on responses to game progression of the child. Power for this module will also be provided by the battery, but will run through the PCBs main power lines.

### 2.3.2. Requirements and Verification

**Subsystem 1 - Processing**

*Central Printed Circuit Board Equipped with ATMega128 System-on-Chip*

<table>
<thead>
<tr>
<th>Requirements</th>
<th>Verification</th>
</tr>
</thead>
<tbody>
<tr>
<td>1) PCB should have seamless tracing with components and limit internal interference</td>
<td>1) No open or short circuits on the PCB that cause the robot to not run altogether</td>
</tr>
<tr>
<td>2) Microcontroller should be able to program the Raspberry Pi auxiliary device at a baud rate of 115200 bps (standard baud rate for Raspberry Pi)</td>
<td>2) Raspberry Pi boots up within 1 minute of robot power-on</td>
</tr>
<tr>
<td>3) Microcontroller should be able to receive input from pressure sensors and transmit the information to the Raspberry Pi</td>
<td>3) Code on microcontroller determines when pressure has been applied to pressure sensors</td>
</tr>
<tr>
<td>4) Microcontroller should be able to receive input from motion-based sensors and adjust distance and speed by transmitting to the servo arms and omni wheels</td>
<td>4) Motion sensors respond to changes in proximity and location of the user.</td>
</tr>
<tr>
<td>5) Utilize on-board memory for sensor data storage and code scripting</td>
<td>5) Data from the storage and running of code can be checked from a console log of the microcontroller</td>
</tr>
<tr>
<td>6) UART, SPI, and I2C data lines run with a latency of less than 50 ms</td>
<td>6) Latency values can be displayed and measured through a console log</td>
</tr>
</tbody>
</table>
### Raspberry Pi 3

<table>
<thead>
<tr>
<th>Requirements</th>
<th>Verification</th>
</tr>
</thead>
</table>
| 1) Receive power from the central PCB of the robot and be able to also power on the central LCD screen.  
2) Compile and execute programs related to the games specified by the components of the ‘Programming’ sub-module (Android SDK and Google Play Services).  
3) Be able to compute game progress and reward system. | 1) When the robot is powered on the central LCD screen will be powered on.  
2) Game library will instantiate displaying available games to play. Moreover, a game can actually be played and interacted with.  
3) Reward system and game progress will be displayed on the central LCD screen as an application in the application library. |

### Rechargeable Battery

<table>
<thead>
<tr>
<th>Requirements</th>
<th>Verification</th>
</tr>
</thead>
</table>
| 1) Supply +3.7V +/- .25V to the PCB as well as all components connected to the battery as specified by the block diagram.  
2) The robot is able to power on and everything using power is active and working.  
3) The battery can be recharged. | 1) Checking of amperage and voltage along all power lines of the robot using an oscilloscope to ensure power and lack of interference.  
2) Power on the robot, verify the robot can move, arms can move and games are functional.  
3) Recharge the battery and see a charge level on the central LCD screen be in charging mode. |

### Subsystem 2 - Programming

#### Central LCD Screen

<table>
<thead>
<tr>
<th>Requirements</th>
<th>Verification</th>
</tr>
</thead>
</table>
| 1) Display the Application library for the user.  
2) Touch inputs work through capacitive touch. | 1) Power on the robot and check the central LCD screen if the application library is there.  
2) Touch the Apps/play a game/touch anything on the interactive parts of the LCD screens and see if it is working. |
### Android SDK

<table>
<thead>
<tr>
<th>Requirements</th>
<th>Verification</th>
</tr>
</thead>
<tbody>
<tr>
<td>1) Android applications are able to programmed and ran through the Raspberry Pi</td>
<td>1) Game selected is successfully loaded and can be played through the central LCD screen</td>
</tr>
</tbody>
</table>

### Google Play Services

<table>
<thead>
<tr>
<th>Requirements</th>
<th>Verification</th>
</tr>
</thead>
<tbody>
<tr>
<td>1) Games are able to launch and run seamlessly</td>
<td>1) In playing a game, the game can run without crashing until completion</td>
</tr>
</tbody>
</table>

### Custom Games

<table>
<thead>
<tr>
<th>Requirements</th>
<th>Verification</th>
</tr>
</thead>
<tbody>
<tr>
<td>1) The Simon Says app displays in the application library, functional in terms that the user can play the game and have their progress tracked and the robot displays the correct responses depending on the users input</td>
<td>1) Start up a game of Simon says and have someone play it for 2 minutes and see if the robot displays the correct response as well as tracks the user’s score</td>
</tr>
</tbody>
</table>

### Movement and Dynamics Control Program

<table>
<thead>
<tr>
<th>Requirements</th>
<th>Verification</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1) The robot goes a set speed with marginal error (within 2% of the set speed)</td>
<td>(1) Run code that activates servo motors and notice robot moves at a steady/safe speed. This can be quantified by logging the registered value of the servo motor on a console log of the microcontroller</td>
</tr>
</tbody>
</table>
## Facial Expression Interaction

<table>
<thead>
<tr>
<th>Requirements</th>
<th>Verification</th>
</tr>
</thead>
<tbody>
<tr>
<td>1) Face LCD should be able to display facial expressions on the screen when games are not being played but robot is on</td>
<td>1) Power on the robot but don’t start a game and check the face LCD</td>
</tr>
<tr>
<td>2) Face LCD should switch from FACE to a “Good Job” or “Try Again” based on how game is going</td>
<td>2) Start up a game of Simon Says and play the game and verify based on the answer if the robot is displaying the correct response.</td>
</tr>
</tbody>
</table>

### Subsystem 3 - Sensors

#### Left and Right Pressure Sensors

<table>
<thead>
<tr>
<th>Requirements</th>
<th>Verification</th>
</tr>
</thead>
<tbody>
<tr>
<td>1) Pressure sensor should respond to a force on the right or left shoulder by projecting a thumbs up onto the LCD on the torso</td>
<td>1) Start up Simon Says game and wait for “Simon Says, touch my shoulder” command. Put pressure on corresponding shoulder and wait for LCD reaction</td>
</tr>
<tr>
<td>2) Fail state: after Simon Says gives prompt and no pressure data is read in for a period of 15 seconds LCD will display a “Try Again”</td>
<td>2) Start up Simon Says game and wait for “Simon Says, touch my shoulder” command. Don’t put pressure and wait for time out and LCD to play “Almost, try again”</td>
</tr>
</tbody>
</table>
### Movement Sensor

<table>
<thead>
<tr>
<th>Requirements</th>
<th>Verification</th>
</tr>
</thead>
<tbody>
<tr>
<td>1) Detects a minimum of 1 foot separation from user through proximity sensor and will maintain this minimum distance in front of the user when arms are pointing downwards</td>
<td>1) Sydekick should not move forward based on proximity sensor data which will give distance from user of approx. 1 foot</td>
</tr>
<tr>
<td>2) Detects a minimum of 1 foot separation from user through proximity sensor and will maintain this minimum distance in front of the user when arms are pointing towards user</td>
<td>2) Sydekick will lifts arms up and will move backwards until proximity data once again is at least 1 foot away</td>
</tr>
<tr>
<td>3) Given a distance of less than 1 foot separation from user, Sydekick will reverse till at least 1 foot away</td>
<td>3) Place Sydekick less than 1 foot from user and then given proximity sensor data Sydekick will reverse</td>
</tr>
</tbody>
</table>
**Subsystem 4 - Arms**

**Left and Right Arm Servos**

<table>
<thead>
<tr>
<th>Requirements</th>
<th>Verification</th>
</tr>
</thead>
<tbody>
<tr>
<td>1) Arms should be able to move up from 0 degrees (pointing downwards) to 90 degrees (pointing at user)</td>
<td>1) Run code that moves arms from 0 to 90 degrees to see full range of motion</td>
</tr>
<tr>
<td>2) Arms should always be stopping at 90 degrees based on safety measures our team desires</td>
<td>2) Run code to move from 0 to 180 degrees and note that it stops at 90</td>
</tr>
</tbody>
</table>

**Subsystem 5 - Wheels**

**Four Omnidirectional Wheels**

<table>
<thead>
<tr>
<th>Requirements</th>
<th>Verification</th>
</tr>
</thead>
<tbody>
<tr>
<td>1) Wheels should be able to move backwards and forwards based on data from proximity sensor in order to maintain distance</td>
<td>1) Run code that shows movement of the robot based on proximity sensor data. Give position of robot in front of user, proximity sensor data should make the robot move forward or backwards or stay still</td>
</tr>
<tr>
<td>2) Wheels should be able to make robot move</td>
<td>2) Run code that makes robot move forward, backward and turn</td>
</tr>
</tbody>
</table>

**Subsystem 6 - Miscellaneous**

**Top LCD Screen**

<table>
<thead>
<tr>
<th>Requirements</th>
<th>Verification</th>
</tr>
</thead>
<tbody>
<tr>
<td>1) Receive power from microcontroller and receive data from the ‘Facial Expression Interaction’ software on the microcontroller</td>
<td>1) Turns on when robot is powered on 2) Displays a particular facial expression depending on game reward system</td>
</tr>
</tbody>
</table>

**Motivation for Blocks**

A breakdown of each of the blocks and why it matters to the overall solution of project is explained below:

**Subsystem 1 - Processing** This is pivotal to success of the project and will act as the brain/processing power of the robot. This will be important to creating a good user experience for the child because it will be the power behind the motors/servos and LCD’s. These are the features that make the humanoid the most life-like and create the best experience for the user.
**Subsystem 2 - Programming:** The Android and LCD will be a huge player in creating the interface that allows the child to play with the robot. Through games like Simon Says and Tickle our team hopes that we can create a fun and therapeutic experience for the child.

**Subsystem 3 - Sensors:** This will be important for the user experience and enjoyment of the child because we are projecting faces onto the LCD screen. This is so that it will emulate a happy and emotion-full environment.

**Subsystem 4 - Arms:** The arms will be helpful to interact during games with the child. They also really help bring together the idea of a human presence that the child can create a symbiotic relationship with. The safety hazard comes into play with the child if the rotation is too fast or too high. We will be looking into this more and making sure that we measure speeds and limit the height of the rotation of the hands.

**Subsystem 5 - Wheels:** The wheels are a crucial part to the robot and it allows our group to circumvent creating “legs” for the robot. This is a simpler design specification to make it so that the robot can follow and be with the child. Our team hopes to create a safety precaution here where the robot will make sure that it stays at least 1 foot away at all times in order to not get too close.

**Subsystem 6 - Miscellaneous:** This block will be incorporating the face for the partial humanoid. Through this, as mentioned before, the team is looking to display faces and a happy environment for the child.

### 2.3.3. Supporting Materials

#### 2.3.3.1. Mechanical Design

Utilized TinkerCAD software to create a mechanical design very similar to the specs given in the **Physical Diagram** under the **Design** section.
2.3.3.2. Circuit Schematics
Figure 4: Prototypical Circuit Schematic

The circuit schematic above consists of the battery connector for the robot as well as the
ATMega128 chip and the base power connections with the Raspberry Pi. This circuit will have
to be elaborated upon for a successful design and is subject to change upon review with course
personnel and instructors.

2.3.3.3. Calculations:

Based on our preliminary research our team has benchmark values that we will be using to
ensure safety of “Sydekick”. Through development of the project if we feel that we can tweak
the speeds we will.

**Speed & Distance:** The typical speed of motorized toy cars for children between the ages three
and five varies between two and three miles per hour with a 6v battery[6]. We want a safe
speed for the robot so that while interacting with the child, if the child were to move away from
Sydekick, Sydekick will follow the child while keeping a safe distance and while keeping a
constant speed with the aid of the accelerometer. Because Sydekick is much larger than a toy
car, the set speed we will use is one mile per hour which is quick enough to follow a young child
and slow enough to not run into or cause damage to the child. The speed will be kept in check
by the motorized base that has four FTC servo motors attached with wheels. The distance to
maintain between the child and Sydekick that will be monitored by the proximity sensor will be
one foot, giving enough distance to ensure Sydekick will not run into the child and close enough
to stay engaged with the child.

**Arm Movement:** Sydekick has two metal bar arms that will be able to raise up to 90 degrees
forward of his body. This is for the games and interaction Sydekick will have with the child such
as touch my hand. This number also adheres to safety protocols to ensure Sydekick does not
have a metal bar pointing straight up which could be hazardous if a child were to fall onto
Sydekick. The arm movement is controlled the FTC servo motors which will ensure the limited
movement.

2.3.3.4. Measurements

Measurements (split by body parts of the robot):

- Head: 10” base for an 4” LCD screen with capacitive touch capabilities.
- Arms: 18” FTC Metal Channel Bars
- Body: 12” x 36” chassis body with an 8” LCD screen with capacitive touch capabilities

More details can be found in the **Physical diagram** section or the **CAD** of Sydekick.
2.3.3.5. Software Flow Chart

Figure 5: “Sydekick” Games Software Flow Chart
2.4. Tolerance Analysis

A critical component of this project is the omni wheels that we will be using for motion of Sydekick. There are a few requirements for the omni wheels to interface effectively with the proximity sensor and that will be discussed below.

The first important factor is figuring out rough coefficient of frictions for the omni wheels in order to understand how much the wheel will slip. Our research led us to find coefficients that had already been calculated for the specific omni wheels that we were using [8]. Radial friction is the friction we consider when Sydekick drives back and forth and Transverse friction is the friction we consider when Sydekick moves sideways. The four frictions that will be important to analyze are: (1) Radial Static Friction, (2) Radial Dynamic Friction, (3) Transverse Static Friction and (4) Transverse Dynamic Friction.

Our research is based on 6” Plastic Omni Wheels and the coefficients are as follows:

6” Plastic Omni Wheel
- Radial Static: 0.9
- Radial Dynamic: 0.8
- Transverse Static: 0.27
- Transverse Dynamic: 0.20

Friction, statically and dynamically, should always be below 1 for normal objects. A frictional coefficient that is greater than one means that the frictional force is greater than that of the normal force meaning there is a considerable amount of friction. It’s clear that the Radial frictions for both static and dynamic are relatively close to 1 which means that this will be noticeable when we move forward and backwards from the user.

Next, this information needs to be applied to the proximity sensor data in order to properly adjust for error and make sure that we are 1 foot away from the user. The proximity sensor we are using is the VCNL4010 Proximity has a range of 20 feet therefore there is not a problem within range for our sensor. With this in mind, our team feels that there should be a level of error that we take into account when we are aiming to be 1 foot away. For this we are aiming to be about 1.3-1.5 feet away in order to account for the friction of moving backwards from the user as well as moving forwards.
3. Cost and Schedule

3.1. Cost Analysis

<table>
<thead>
<tr>
<th>Design Parts</th>
<th>Cost/Unit</th>
<th>Quantity</th>
<th>Total Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Raspberry Pi 3 Model A+</td>
<td>$25.00</td>
<td>1</td>
<td>$25.00</td>
</tr>
<tr>
<td>Raspberry Pi LCD Display 3.5 inch 480x320 TFT Touch Screen Monitor</td>
<td>$20.00</td>
<td>2</td>
<td>$40.00</td>
</tr>
<tr>
<td>FTC Servo Motors</td>
<td>$15.00</td>
<td>6</td>
<td>$90.00</td>
</tr>
<tr>
<td>Wheels (2 pack)</td>
<td>$2.00</td>
<td>2</td>
<td>$4.00</td>
</tr>
<tr>
<td>FTC Channel Metal Bar</td>
<td>$10.00</td>
<td>2</td>
<td>$20.00</td>
</tr>
<tr>
<td>Pressure Sensor</td>
<td>$5.00</td>
<td>5</td>
<td>$25.00</td>
</tr>
<tr>
<td>Accelerometer Sensor</td>
<td>$10.00</td>
<td>1</td>
<td>$10.00</td>
</tr>
<tr>
<td>VCNL4010 Proximity/Light Sensor</td>
<td>$7.50</td>
<td>1</td>
<td>$7.50</td>
</tr>
</tbody>
</table>

Labor

$35/hr * 2.5 * (50hrs (hardware) + 90hrs (software)) = $12,250

Grand Total: $12,472

Raspberry Pi Model A+: Used for the crux of the robot due to the ability to interface with various sensors as well as its diverse wireless capabilities.

Raspberry Pi LCD Display Touch Screen: The LCD Displays will offer the child a way to interact with such as hugging, tickling, etc. There will be another LCD screen to encourage the
reward system for the child playing with the robot and will simply have facial emotions programmed into it as well as a speaker to communicate with the child.

**FTC Servo Motors**: There will be a total of six motors: four for the base of the robot for basic movement and two for the arms that will only allow 1 degree of motion restricting the movement to just raising arms up and down.

**Wheels**: There will be four wheels attached to the servo motors on the base of the robot for the movement aforementioned.

**FTC Channel Metal Bar**: There will be two of these for the arms attached to the servo motors with the restricted movement of up and down.

**Pressure Sensors**: There will be a total of five pressure sensors, one on the base of the robot, one on the body of the robot, one for each arm and one for the head. These sensors will be used for the games that the robot will play with the child.

**Accelerometer Sensor**: This will be used to check if the robot has fallen down by checking the speed. As a safety precaution, if the robot has fallen down, we want the robot to turn off to prevent any danger to the child.

**VCNL4010 Proximity/Light Sensor**: This sensor will also be used for safety to ensure the robot does not hit the child.
3.2. Schedule

<table>
<thead>
<tr>
<th>Week</th>
<th>Goal</th>
</tr>
</thead>
<tbody>
<tr>
<td>2/18</td>
<td>Design Document</td>
</tr>
<tr>
<td>2/25</td>
<td>Gather all components required (except the chassis)</td>
</tr>
<tr>
<td>3/4</td>
<td>3D Print Chasis for the base of Sydekick</td>
</tr>
<tr>
<td>3/11</td>
<td>Begin the hardware design of the Sydekick (make the basic body of the robot)</td>
</tr>
<tr>
<td>3/18</td>
<td>Complete the hardware design of the Sydekick (integrate the LCD screens and sensors)</td>
</tr>
<tr>
<td>3/25</td>
<td>Begin the software design of the Sydekick (user interface and movement)</td>
</tr>
<tr>
<td>4/1</td>
<td>Complete software design of the Sydekick (children games and reward system)</td>
</tr>
<tr>
<td>4/8</td>
<td>Have a working prototype of Sydekick to start evaluating bugs and safety protocols</td>
</tr>
<tr>
<td>4/15</td>
<td>Fixed all issues and bugs, ensures it adheres to our criteria and complete Sydekick</td>
</tr>
<tr>
<td>4/22</td>
<td>Prepare and complete our final presentation</td>
</tr>
</tbody>
</table>

**April 29th, 2019**

**Distribution of Workload:**

Week 1 (3/4): Chassis creation by Sam while Rohan & Bala begin creating the body of the robot without the chassis.

Week 2 (3/11): Bala & Rohan put together the physical design of the robot while Sam begins the software integration of Raspberry Pi with other sensors.

Week 3 (3/18): Everyone work together to complete the physical design as well as the remaining hardware integration with the robot.

Week 4 (3/25): Bala starts working on gathering sensor data for the games and basic movement, Sam starts the user interface for Sydekick and Rohan begins working on the games for the application library.

Week 5 (4/1): Complete the remaining software designs and integrations with the hardware and software (everyone).

Week 6 (4/8): Sam verify the hardware design of the Robot specifically with the LCD screen interface and integration with the robot. Bala verifies the basic movement and sensors. Rohan verifies the software games and the reward system.

Week 7 (4/15): At this point, whatever bugs we find we will tackle as a group and prepare our final presentation.
Safety & Ethics:

As a preface, “SydeKick” has to be built with extra precaution because of the children that are using it, in this case that will be children that have autism. This section will be divided into robotic specific hazards and then hazards related users interfacing with the robot.

During development its important to take into consideration the overall appearance of the robot and making sure that we take care of our parts. The first being the mechanical build, it will be crucial to have all parts completely consumer ready before building. This means that all parts need to be soldered and smoothed out to prevent users getting there fingers caught. The second concern is the wiring, integrating the PCB and Raspberry Pi with the humanoid design will require extensive wiring which will need to be done very diligently. Our group will be extra careful in making the appearance of the product ready in order to make it as safe as possible for the child. This means that nothing will be sticking out and nothing will have a sharp edge where the child could get hurt. To best solve the wiring and soldering problem it will be important as a team to make every part modularly perfect. This means that we can not leave wiring and soldering till the end. During usage of the robot it will important to take in to account the the arms and motorized base. Both of these needs to be tested and retested in order to make sure that the user does not get hurt. Our group will be solving for this by looking running different tests on the robot to see what will be the fastest speed we can operate the robot at. There will be a safety precaution in the arms so that they don’t raise to quickly or too high in order to avoid damage. Additionally, the motorized base will also be fine tuned so that it always stays “x” feet away from the user. This “x” will be decided on later when we get closer to being able to test the product.

An ethical issue that we could see coming into play is whether or not it makes sense to leave a child with autism to play with a robot. Robotics is a very fast-paced and constantly advancing area of study which is not always fully understood. With this being said it creates an ethical puzzle of whether or not it's okay to push the responsibilities of people to nurture kids to a robot. I think this problem is shown in IEEE Code of Ethics (#5) where the goal of IEEE as a whole is to help improve the understanding and capabilities of individuals when it comes to new and emerging technologies[3]. The best way “SydeKick” will avoid this issue is to create an environment where parents need to be around with the child while they are playing and having therapy sessions but allow enough space to where their full attention is not commanded. As an emerging technology continues to grow its important to start giving more responsibility to technology while also not completely unlatching. IEEE Code of Ethics (#9) will be another really important code to keep in mind because safety has to be the paramount priority. Throughout building as well as usage by consumers this project will take every precaution necessary to maintain safety standards. This will include following lab safety trainings as well as taking the best care of our equipment and documenting our processes.

Testing the product will be another ethical issue that we have considered. Getting authorization to work with children with autism is very tricky and the liabilities that surround it
make the environment difficult. Because of this, our team will be finding literature that shows in theory playing these kinds of games with children on the spectrum.
References:


