Soft Skills

Gruppe 2

21. August 2022

Sommersemester 2022

Soft Skills Projekt: Smart Start

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einleitung

Dieses Dokument beinhaltet die Dokumentation des Soft-Skills Sommerprojekts der Gruppe 2. Das Projekt fand im Sommersemester 2022 statt und verfolgte das Ziel einen Dienst / eine Automatisierung für Wohn- und Arbeitsflächen zu entwickeln, um die Sicherheit, Wohn- und Lebensqualität und / oder die Energieeffizienz steigern.
Ein weiteres Ziel ist es im Team zu agieren und gelernte Inhalte anzuwenden.

GitLab: https://gitlab.uni-oldenburg.de/nier0394/soft-skills-smart-start.git Für den Zugriff auf das GitLab melden Sie sich bitte bei:
Joshua.lenhoff@uni-oldenburg.de

1.1 Projektvorgaben

Als Vorgabe für das Projekt wurde dieses Semester ein Dienst oder eine Automatisierung für Wohn- und Arbeitsflächen gesucht, die die Sicherheit, die Wohn- und Lebensqualität oder die Energieeffizienz steigert.
Für die Umsetzung werden Sensoren oder Aktoren empfohlen aber auch das anwenden von gelernten Methoden aus der Vorlesung des Moduls Soft Skills.

1.2 Ideenfindung

Bevor ein Projekt umgesetzt werden kann mussten Ideen gesammelt werden. Dafür verwendete unsere Gruppe eine Mischung aus Brainstorming, Brainwriting und der 6 3 5 Methode aus der Vorlesung.
Wir verwendeten die Website retrotool.io um die Ideen zu sammeln und diese zu erweitern. Die Ideenfindung wurde durch die ruhige und lockere Atmosphäre innerhalb der Gruppe unterstützt.

Abbildung 1: Ideenfindung per Retrotool

Abbildung 1 veranschaulicht das retrotool und die Karteikarten, die erstellt wurden. Die ersten fünf Ideen der Gruppe waren ein Smart Pc, ein Smart Mirror,

eine Smart Wardrobe, eine Smarte Klingel und eine Smartwatch. Weitere Karteikarten wurden diesen fünf Ideen hinzugefügt, um die Ideen zu erweitern.

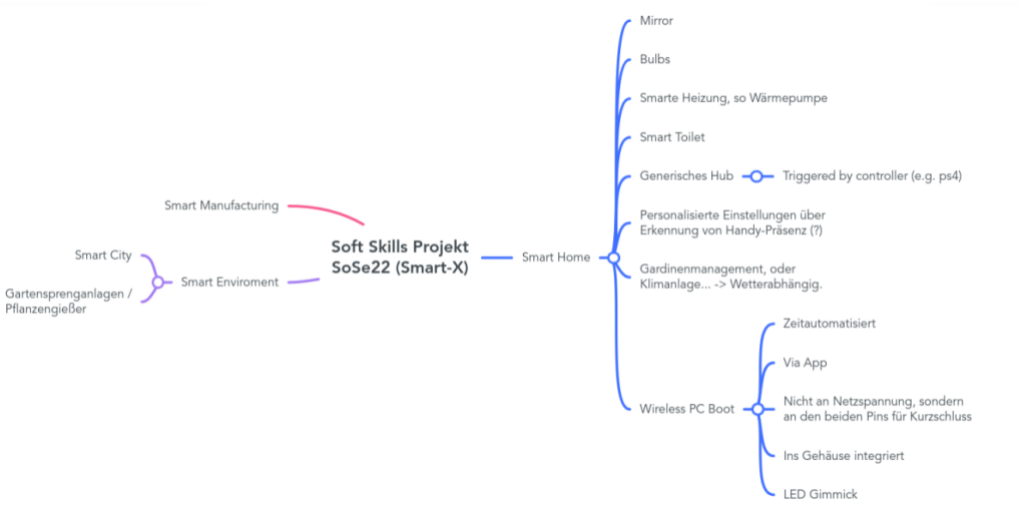

Abbildung 2: Ideensammlung

Abbildung 2 dient der Zusammenfassung all unserer Ideen für das Projekt und der Aufteilung in die verschiedenen Bereiche, in welchen diese Anwendung finden. Ein großes Augenmerk wurde hierbei auf den Bereich Smart Home geworfen, wie man der Abbildung 2 entnehmen kann.

1.3 Motivation

Die Motivation zur Entwicklung des Projekts entstand aus Alltagsgedanken. Wenn der Computer lange für das Hochfahren braucht oder man kurzfristig außer Haus muss, ist man gezwungen, bis zum Computer zu gehen und diesen per Hand an/auszuschalten. Braucht der Computer nun lange zum Starten, dann verliert man Zeit, die man anders nutzen könnte (Lebensqualität). Hat man vergessen, den Computer auszuschalten und man muss kurzfristig aus dem Haus, dann wird der Computer meist angelassen (Energieeffizienz). Nun soll eine Möglichkeit entwickelt werden, den Computer und weitere technische Geräte über eine App ein- und ausschalten zu können.
Somit deckt dieses Projekt die Projektvorgaben ab und dient zur Steigerung der Lebensqualität sowie der Energieeffizienz.

2 Analyse

Der Abschnitt Analyse dient der Analysierung des Projektes im Bereich der Anforderungen sowie der Projektplanung.

2.1 Anforderungen

Bevor das Projekt weiter geplant werden konnte, mussten die grundlegenden Funktionalitäten des Projekts festgelegt werden. Dabei wurden die Funktionalitäten so minimal wie möglich gehalten. Somit hätte das Projekt durch kleine Erweiterungen noch verbessert werden können, falls der Prototyp so weit funktioniert. Im Folgenden sind die Anforderungen aufgelistet:

- Als Nutzer möchte ich meinen Computer aus der Entfernung über eine App an- / ausschalten können
- Das Produkt muss verschiedene Arten von Computern einschalten können, sodass das System individuell einsetzbar ist
- Das Produkt muss ohne Aufwand an dem Computer befestigt werden können, sodass das Gehäuse unbeschadet bleibt

2.2 Projektplan

Damit das Projekt in dem geplanten Zeitraum umgesetzt werden kann, wurde schnellstmöglich ein Meilensteinplan entwickelt. Anhand der Meilensteine werden die Aufgaben bei den wöchentlichen Treffen ausgearbeitet, welche zur Erreichung des Meilensteins nötig sind. Die Fertigstellung des Projekts ist so angesetzt, dass von der Fertigstellung bis zur Präsentation noch eine Woche Spielraum für eventuelle Komplikationen ist.

08.06.2022	Was brauchen wir damit wir die Bestellung fertig machen können?	
15.06.2022	Start des Projekts	
22.06.2022	Treffen in der Woche im Makerspace, Doku auf den aktuellen Stand bringen und Webseite gestalten	
29.06.2022	Wifi Verbindung, App und Design	
06.07.2022	Motorfunktion mit Standfeststellung nach Stromausfall etc.	fertiger 3D Druck
14.07.2022	Fertigstellung des Projekts	

3 Realisierung

Der Abschnitt Realisierung dient der Veranschaulichung der Realisierung des Projekts.

3.1 Hardware

Als Hardware entschied sich unsere Gruppe einen Motor und einen Mikrocontroller zu verwenden.

Servomotor:

Abbildung 3: RoboMall SG90 9g Mirco Servermotor

Der Servomotor 3 dient dazu präzise und zuverlässige Steuerungen zuzulassen. Er ist dafür zuständig, das Gelenk im Inneren der Vorrichtung so zu bewegen, dass der Än-Knopf"gedrückt werden kann.

Im inneren des Servomotors befinden sich ein Gleichstrommotor, ein Potentiometer und der Steuerkreis.

Das Funktionsprinzip des Servomotors wird nachfolgend beschrieben:
Schritt 1: Der Servomotor empfängt ein Steuersignal und dreht sich in die gewünschte Position. Normalerweise kann sich ein Servomotor nur innerhalb eines bestimmten Winkels drehen, doch in unserem Fall lassen wir den Servomotor bis zu 180 Grad drehen.
Schritt 2: das Potentiometer kann die Position der Welle erkennen, wenn der Motor dreht. Erreicht die Welle den gewünschten Winkel, erkennt der Steuerkreis dies über das Potentiometer und hält den Motor an. Dann kommt der zum Anfangspunkt zurück.

Mikrocontroller:

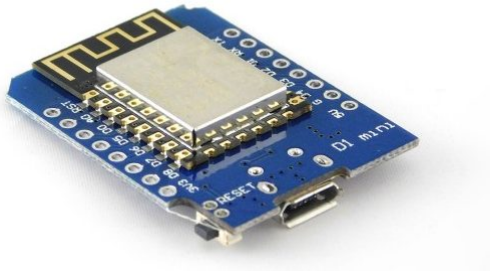

Abbildung 4: D1 mini ESP8266 ESP-12

Der Mikrocontroller aus Abbildung 4 dient zur Weiterleitung von präzisen Steuersignalen an den Servomotor aus Abbildung 3 und als Verbindungspunkt für unsere Website / App, worüber das An-/Aussignal gesendet wird.

Der WeMos D1 Mini-Modul (Abbildung 4) basiert auf den ESP8266 ESP-12F Wi-Fi Mikrocontroller und kann somit mit der Arduino IDE programmiert werden. Mit der Steuerung durch Arduino beschäftigen wir uns in dem Kapitel SSoftware".

3.2 Modellierung

Erste Planung:
Das Projekt benötigt ein Gehäuse, welches später an das technische Gerät angebaut werden kann. Dabei muss das Gehäuse einerseits groß genug sein, um Platz für jegliche Hardware zu haben, andererseits sollte es aber so klein wie möglich sein, um später nicht zu viel Platz zu verbrauchen und handlich zu bleiben. Dafür wurden die Größen des Mikrocontrollers und des Servomotors aufgenommen. Außerdem gilt es zu überlegen, ob die beiden Komponenten eher übereinander oder nebeneinander platziert werden sollen.

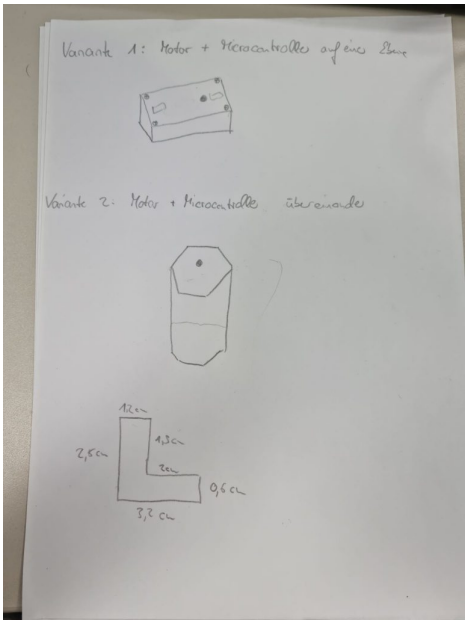

Abbildung 5: Erstes Modell

Die erste Planung der Modellierung erfolgte per Zettel und Stift, wie man der Abbildung Abbildung 5 entnehmen kann. Hierfür wurden zwei Formen als Gehäuse vorgeschlagen. Ein Quader und ein Zylinder wie auf der Abbildung zu sehen[Abbildung 5]. Beide Modelle besitzen ein kleines Loch auf der Unterseite, durch welches später ein Stab den Knopf des anzuschaltenden Geräts gedrückt werden kann. Allerdings hat der Quader den Vorteil der größeren ebenen Fläche. Diese ist essenziell für das Projekt, da das Gehäuse später auf dem Gerät gut halten muss. Daher fiel die Entscheidung auf den Quader.

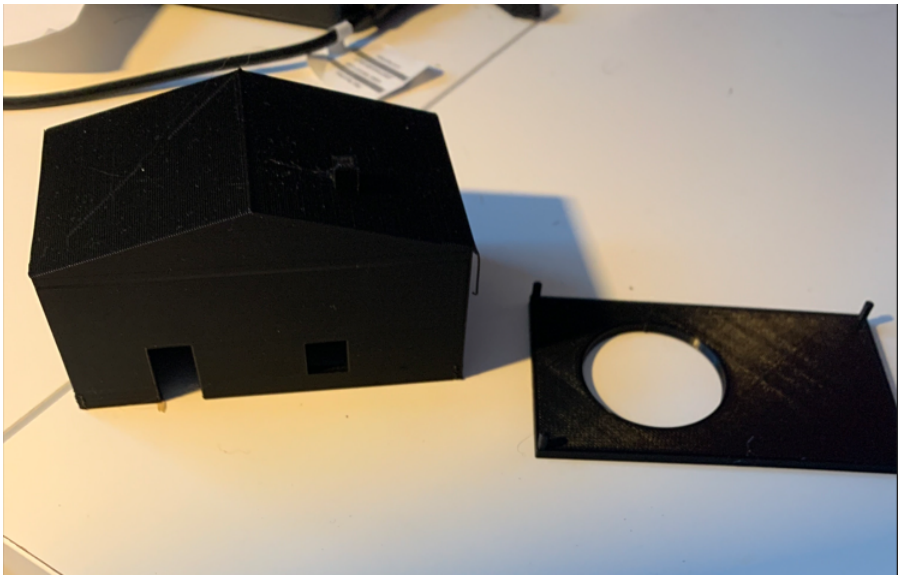

Abbildung 6: Fertiger Druck des ersten Entwurfs

Nun galt es, den Quader zu Modellieren, hierfür entschieden wir uns für das aus der Vorlesung bekannte Programm BlocksCAD. Nach der ersten Planung entstand das erste Modell mit dem Körper eines Quaders (siehe Abbildung 6). Dazu kam die Idee, dass eine schlichte Box nicht sehr ansprechend sei und daher wurde Form eines Hauses gewählt. Außerdem wurde das Modell so modelliert, dass es Möglichkeiten für LEDs gab. Die Bodenplatte des Hauses hat ein großes Loch, durch welches der Motor seinen Stab bewegen kann. In den Ecken des Hauses sind auf der Unterseite kleine Löcher eingebaut, welche nur minimal größer sind als die kleinen Zylinder an den Ecken der Bodenplatte. So sollen die beiden Komponenten zunächst zusammenhalten.

Zweiter Entwurf:
Nach dem Erstellen des ersten Modells fiel die Entscheidung auf ein schlichteres Gehäuse, sodass die Haus-Idee wieder verworfen wurde. Allerdings musste das Problem gelöst werden, dass aus der sich drehenden Bewegung des Motors, eine vertikale Bewegung gemacht werden soll. Dafür wurde an den Motor ein Stab angebaut, mit einem Gelenk. Dieses Gelenk drückt während der drehenden Bewegung auf den Stab und drückt dieses somit runter und zieht ihn wieder hoch.

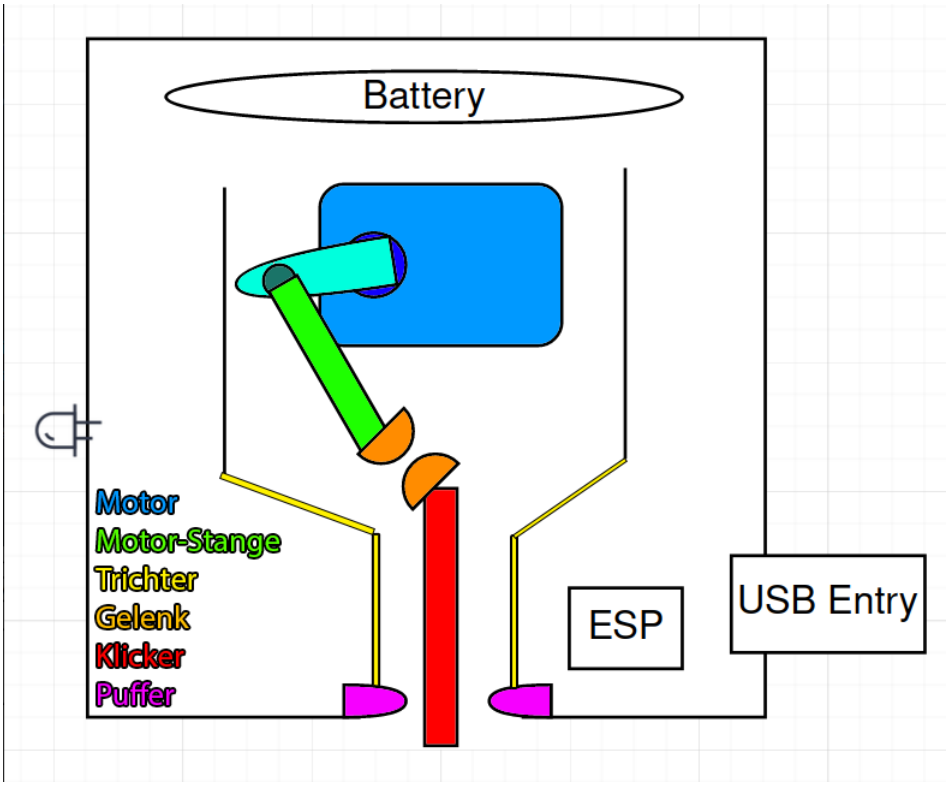

Abbildung 7: Zweiter Entwurf

Der Motor dreht sich durch die Software jeweils 360°. Um an dem Motor unseren Mechanismus zu befestigen, entschied man sich dazu, die mitgelieferten Materialien zu verwenden.

Im Gegensatz zu der ersten Skizze entschieden wir uns dazu, einen anderen Mechanismus zu verwenden. Hierzu wird der Klicker mit einem Gelenk an der Motor-Stange befestigt. Somit wird, wenn sich der Motor nach unten bewegt der Klicker automatisch nach unten gedrückt. Äquivalent funktioniert dies, wenn der Motor sich nach oben dreht.
Um nun den Mechanismus auch in der richtigen Position zu behalten, verwenden wir eine Art Trichter, sodass sich der Klicker nicht nach rechts oder links wegbewegt. Um den Spielraum des Klickers im unteren Bereich zu verringern, nutzen wir Puffer, sodass sich der Klicker nur in einem kleinen Bereich aus dem Gehäuse bewegen kann.

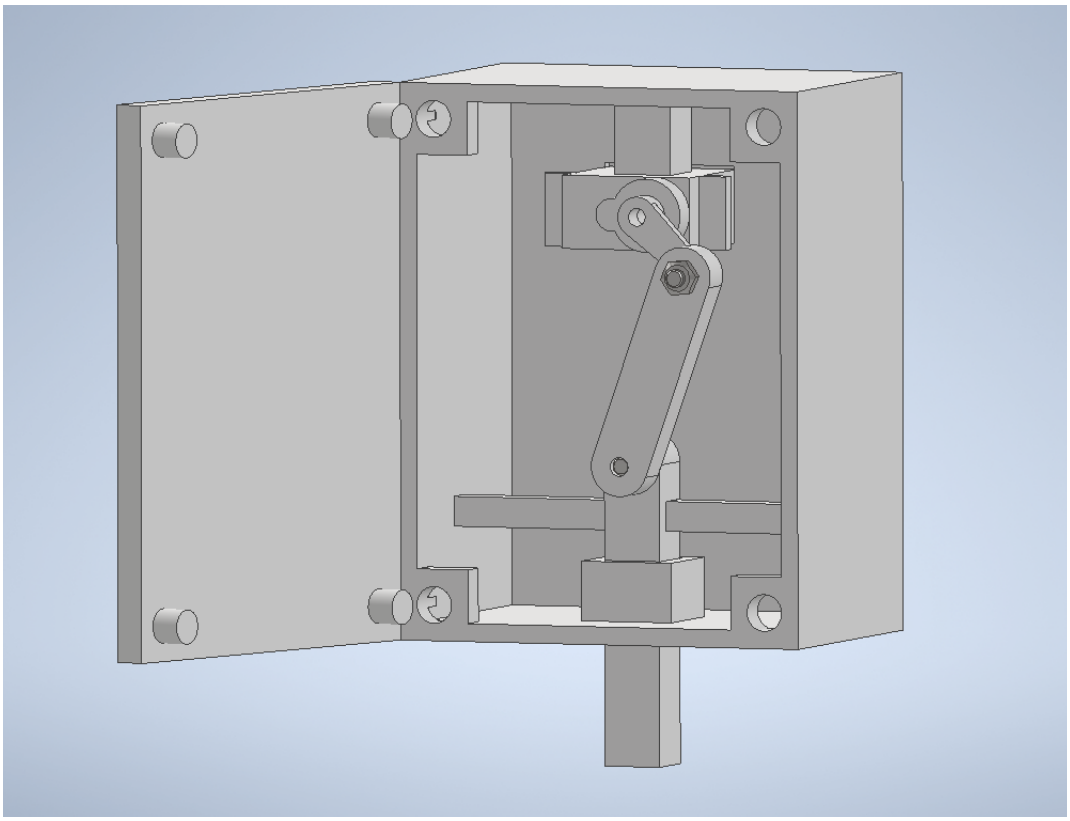

Abbildung 8: Modellierung des Zweiten Entwurfs

Da wir das Modell Drucken mussten, setzten wir die Skizze in einem 3D Modellierungs-Programm um (siehe Abbildung 8). Nun mussten wir das Modell nur noch ausdrucken und zusammenfügen.

3.3 Software

3.3.1 Ablauf / Anforderung an den Mikrocontroller

Die Software für den Mikrocontroller muss für unsere Zwecke folgende Dinge können:

- Öffnen einen Access Points ("AP")
- Eingabe eines WiFi Kennworts
- Steuerung des Servos
- Loop
- Fernsteuerung via. Request-Handling
- Steuerung via. lokaler Website
- Visuelles Feedback via. LED

3.3.2 Genutzte Bibliotheken

Damit wir die Features, welche für die Umsetzung relevant sind, unterstützen können, brauchen wir entsprechende Bibliotheken:

```
1 //Bibliotheken fuer WifiSetup und (Fern-)Steuerung
2 #include <DNSServer.h>
3 #include <ESP8266WebServer.h>
4 #include <WiFiManager.h>
5 #include <ESP8266WiFi.h>
6
7 //Bibliothek zum ansteuern des Servo
8 #include <Servo.h>
```

Diese teilen sich in 2 Kategorien auf

- Wifi Setup und (Fern-)Steuerung
- Servo Steuerung

Die Bibliothek WifiManager.h verwenden wir für allgemeine Netzwerkkommunikation via WiFi.

Damit diese aufgesetzt werden kann, verwenden wir die Bibliotheken DNSServer, ESP8266WebServer, sowie ESP8266WiFi.

Mehr dazu im nächsten Abschnitt.

3.3.3 WiFi Setup

Für das WiFi Setup werden wir einen lokalen AP öffnen, wodurch der Nutzer seinen eigenen AP via. BSSID auswählen kann und danach das entsprechende Passwort eingeben kann.

Kommen wir zuerst zu den globalen Variablen:

```
WiFiServer server(80);  //Server for AP
WiFiClient client;      //Client for communication
bool wifiConnected = false;
```

Ohne weiter auf diese einzugehen kommen wir direkt zu unserer setup()-Methode, da das Konzept anhand dieser verständlicher zu Erläutern ist:

```
void setup()
{
  Serial.begin(115200);

  //Pins
  pinMode(servoPin, OUTPUT);
  pinMode(ledPin, OUTPUT);

  WiFiManager wifiManager;

  wifiManager.autoConnect("SoftSkillsSoSe2022");

  Serial.println("Connected.");
  server.begin();
  Serial.println("Server started");

  // Print the IP address
  Serial.print("Use this URL to connect: ");
  Serial.print("http://");
  Serial.print(WiFi.localIP());
  Serial.println("/");
  wifiConnected = true;

  servo.attach(servoPin); //setzen des Servo auf den D-PIN 9
  digitalWrite(servoPin, LOW);
}
```

Wie wir hier sehen können öffnen wir einen AP mit der BSSID 'SoftSkillsSoSe2022', womit sich der Endnutzer verbinden kann.

Nachdem sich der Nutzer verbunden hat wird dieser folgendermaßen gegrüßt:

! Bild hier einfügen

Nach Auswahl des eigenen AP und Eingabe des entsprechenden Passwortes wird der AP des Mikrocontrollers geschlossen und es stehen 2 Modi zur Steuerung des Gerätes zur Verfügung.

3.3.4 Servo Setup

Um den Servo zu Konfigurieren und Anzusteuern braucht es lediglich die Bibliothek "Servo.h" und einen GPIO Pin (in unserem Fall PIN 9).
Zuerst brauchen wir eine globale Variable um den Servo von Software-Seite aus anzusteuern und einen GPIO PIN für die reale Verbindung zwischen Controller und Servo.

```
1 Servo servo;            //Servo for functionality
2 const int servoPin = 9; //(D4)
```

Blicken wir auf die setup()-Methode zurück fällt folgendes auf:

```
1 servo.attach(servoPin); //setzen des Servo auf den D-PIN 9
2 digitalWrite(servoPin, LOW);
```

Hier ordnen wir unsere Variablen 'servo' vom Typen 'Servo' den entsprechenden Pin zu und setzen diesen auf LOW um jegliche Kontraktion vorerst zu vermeiden.

3.3.5 Loop

Bevor wir weiter auf die Steuerung des Gerätes eingehen, werfen wir kurz einen Blick auf unsere loop()-Methode:

```
1 void loop(){
2   if(!wifiConnected)
3   {
4     hostPage();
5   }
6
7   while(!client.available())
8   {
```

```
    delay(1);
  }

  if(client.available())
  {
    requestHandling();
  }
}
```

Diese unterscheidet zwischen 2 Modi:

- 'Ich habe eine Verbindung und kann ferngesteuert werden'
- 'Ich habe keine Verbindung, aber man kann mich lokal steuern'

Im ersten Fall rufen wir die Methode requestHandling() auf, im zweiten Fall entsprechend die hostPage()-Methode.
Auf beide Methoden werden wir im folgenden eingehen.

3.3.6 Request-Handling für Fernsteuerung

Alles klar, wir haben eine Verbindung! Aber was nun?
Im Endeffekt soll der Mikrocontroller auf einen Befehl warten. Dies realisieren wir über einen lokalen Webserver, wodurch der Endnutzer diesen Port freigeben müsste, oder entsprechend einen Tunnel aufsetzen müsste.
Sehen wir davon ab ist das Konzept sehr simpel und wir unterscheiden zwischen zwei Endpunkten:

- /COMPUTER=ON und
- /COMPUTER=OFF

Werfen wir einen Blick in den Code, bei welchem wir auf eine Anfrage warten:

```
WiFiServer server(80);
WiFiClient client;

void requestHandling()
{
  WiFiClient client = server.available();
  if (!client) {
    return;
  }
   // Read the first line of the request
  String request = client.readStringUntil('\r');
  Serial.println(request);
  client.flush();
```

Insofern der Mikrocontroller den Port 80 noch nicht nutzt und eine Verbindung zu einem WLAN hat, warten wir auf eine Request von außen.
Sobald eine Request (="Anfrage") empfangen wird passiert folgender Abgleich:

```
1   if (request.indexOf("/COMPUTER=ON") != -1)
2   {
3      ....
4   }
5   if (request.indexOf("/COMPUTER=OFF") != -1)
6   {
7      ....
8   }
```

Hierdurch "weiß" der Controller was zu tun ist. Aber was genau ist denn nun zu tun?

- Servo kontrahieren
- LED an/aus machen
- Visuelles Feedback via Response

Für die Kontraktion des Servos haben wir eine push()-Methode angelegt, welche wie folgt aussieht:

```
1  void push()
2  {
3    //For Schleife zum drehen des Servos um 360°
4    for(position = 0; position < 360; position++)
5    {
6      servo.write(position); //Schreiben des aktuellen Wertes der Variable 'position' (1...360)
7      delay(waitTime); //Pause
8    }
9  }
```

Aufgrund unserer Umsetzung eines Gelenks (Siehe Modellierung), müssen wir lediglich eine 360° Rotation ausführen für einen Knopfdruck.

Für das Nutzer-Feedback über die LED lassen wir diese im Anschluss kurz wie folgt blinken:

```
1  void led()
2  {
3    int delayTime = 500;
4    digitalWrite(ledPin, HIGH);
```

```
  delay(delayTime);
  digitalWrite(ledPin, LOW);
  delay(delayTime);
}
```

Hier der komplette Code d. Request-Handling (zuvor nicht gezeigt, aufgrund des hohen Anteils an HTML:

```
void requestHandling()
{
  WiFiClient client = server.available();
  if (!client) {
    return;
  }
  // Read the first line of the request
  String request = client.readStringUntil('\r');
  Serial.println(request);
  client.flush();

  int value = LOW;
  if (request.indexOf("/COMPUTER=ON") != -1)
  {
    push();
    value = HIGH;
  }

  if (request.indexOf("/COMPUTER=OFF") != -1)
  {
    push();
    value = LOW;
  }

  client.println("HTTP/1.1 200 OK");
  client.println("Content-Type: text/html");
  client.println("Origin: SoftSkillsSoSe2022-Gang");
  client.println(""); //  do not forget this one
  client.println("<!DOCTYPE HTML>");
  client.println("<html>");
  client.println("<body>");
  client.println("<h1>Computer is powered now: ");

  if(value == HIGH)
  {
```

```
    client.print("On</h1>");
  } else {
    client.print("Off</h1>");
  }
  client.println("<br><br>");
  client.println("<a href=\"/COMPUTER=ON\"\"><button>Turn On </button></a>");
  client.println("<a href=\"/COMPUTER=OFF\"\"><button>Turn Off </button></a><br />");
  client.println("</body></html>");

  delay(1);
  Serial.println("Client dicsonnected");
  Serial.println("");
}
```

3.3.7 Steuerung via. direkter Interaktion

Wie man zuvor ggf. feststellen konnte unterscheidet sich die beiden Interaktionsmodi im wesentlichen nur darin, ob eine Verbindung zum WiFi besteht, oder nicht.

Da das Konzept somit nahezu identisch ist hier die Implementierung der hostPage()-Methode:

```
void hostPage()
{
  WiFiClient client = server.available();   // Listen for incoming clients

  if (client) {                             // If a new client connects,
    String currentLine = "";                // make a String to hold incoming data from client
    while (client.connected()) {            // loop
      if (client.available()) {             // if there's bytes to read
        char c = client.read();             // read a byte, then
        Serial.write(c);                    // print it out the serial monitor
        header += c;
        if (c == '\n') {                    // if the byte is a newline character and the current
                                            // newline characters which indicated the end of the
          if (currentLine.length() == 0)
          {
              //Response-Header
              client.println("HTTP/1.1 200 OK");
              client.println("Content-Type: text/html");
              client.println("Origin: SoftSkillsSoSe2022-Gang");
```

```
20                client.println(""); //  do not forget this one
21                client.println("<!DOCTYPE HTML>");
22                client.println("<html>");
23
24                //Header
25                client.println("<head><meta name=\"viewport\" content=\"width=device-width, initial
26                client.println("<link rel=\"icon\" href=\"data:,\">");
27                client.println("<style>html { font-family: Helvetica; display: inline-block; margin
28                client.println(".button { background-color: #195B6A; border: none; color: white; pa
29                client.println("text-decoration: none; font-size: 30px; margin: 2px; cursor: pointe
30                client.println(".button2 {background-color: #77878A;}</style></head>");
31
32                //Body
33                client.println("<body>");
34                client.println("<h1>Computer is powered now: ");
35
36                if(value == HIGH)
37                {
38                  client.print("On</h1>");
39                } else {
40                  client.print("Off</h1>");
41                }
42                client.println("<br><br>");
43                client.println("<a href=\"/COMPUTER=ON\"\"><button>Turn On </button></a>");
44                client.println("<a href=\"/COMPUTER=OFF\"\"><button>Turn Off </button></a><br />");
45                client.println("</body></html>");
46
47                delay(1);
48                Serial.println("Client dicsonnected");
49                Serial.println("");
50                break;
51            } else {
52              currentLine = "";
53            }
54          } else if (c != '\r') {
55            currentLine += c;
56          }
57        }
58      }
59      header = "";
60    }
61  }
```

3.4 Nutzerhandbuch

Im Folgenden werden alle Eigenschaften und Funktionen unseres Produktes Smart Start aufgeführt und beschrieben.

3.4.1 Produkt

Smart Start ist ein alltagserleichterndes Produkt, welches die Eigenschaft besitzt, den PC aus der Ferne über einen Knopfdruck an oder auszuschalten. Hierfür muss ein Benutzer nur einen Button über eine App/WebApp betätigen und der Mechanismus im Inneren der Box wird getriggert. Das Produkt ist so ausgelegt, dass es immer und von überall betätigt werden kann (so lange sich der Mikrocontroller im Netzwerk befindet).

3.4.2 Aufbau

Das Produkt, bestehend aus mehreren Einzelteilen, ist wie folgt aufgebaut: Im Inneren sind ein Mikrocontroller, ein Servo-Motor, eine Batterie und ein Hebelgelenk verbaut. Der Mikrocontroller bildet die Steuereinheit, welche dem Servo-Motor das Signal zu drehen (360 Grad) gibt. Der Mikrocontroller wird über einen Button in einer WebApp gesteuert, wird dieser gedrückt, wird ein Signal an den Controller gesendet und dieser leitet das Signal an den Servo-Motor weiter, welcher anfängt sich um 360 Grad zu drehen. Der Servo-Motor ist an einem Hebelgelenk befestigt, dieses ist so designt, dass der Hebel, an der Unterseite der Box, gerade auf den Än-Knopf"des Gehäuses drückt. Dreht sich der Servo-Motor um 360 Grad, bewegt sich der Hebel linear raus und wieder rein. So wird sichergestellt, dass der Knopf sicher gedrückt wird.

Das Produkt kann durch mehrere Möglichkeiten an das PC Gehäuse befestigt werden. Hierfür wird empfohlen Magneten oder eine spezielle Halterung zu verwenden. Ist die Box fest genug an das Gehäuse befestigt, kann der Mikrocontroller mit dem Netzwerk verbunden werden. Sobald das geschehen ist, ist das Gerät einsatzbereit. Nun kann der Knopf nach belieben und von überall betätigt werden.

4 Projektmanagement

4.1 Aufgabenverteilung

Da wir verschiedenen Stärken und Interessen in unserer Gruppe hatten, haben wir uns dazu entschieden, diese für die Aufgabenverteilung zu beleuchten. So kam es, dass sich Philip und Joshua mehr um den Software-Part gekümmert haben und Maximilian, Johannes und Noemi die Modellierung und den Druck des Gerätes angegangen sind. Des Weiteren haben sich alle Gruppenmitglieder regelmäßig im Makerspace getroffen, um die vorher entstandenen Skizzen und Ideen umzusetzen. Dennoch war es keine statische Aufteilung, jedes Gruppenmitglied konnte sich freit entfalten und Aufgaben vom anderen Bereichen übernehmen und lösen.
Die Organisation innerhalb der Gruppe verlief demokratisch und wie in einem Stuhlkreis. So hatte jedes Gruppenmitglied die Möglichkeit, Anmerkungen und Ideen in das Gruppengespräch mit einzubringen.
Neben den verschiedenen Aufgaben wurde ebenfalls für jedes Gruppenmitglied ein Protokollant festgelegt.

5 Tools

Im folgenden werden alle verwendeten Tools aufgelistet und beleuchtet

5.1 BlocksCAD

BlockCAD ist ein Programm für 3D-Modelling im Bauklotz-Format. So können einfache 3D-Strukturen in kurzer Zeit erstellt werden. Wir nutzten BlocksCAD um unsere Skizzen und Entwürfe zu modellieren, damit diese anschließend gesliced und gedruckt werden konnten.

5.2 Discord

Discord ist eine App für Server, Nachrichten und Videoanrufe. Wir nutzten diese, um unsere Meetings abzuhalten, schnell Informationen auszutauschen und Planung zu betreiben.

5.3 Wordpress

WordPress ist eine Webanwendung, mit der Websites erstellt und Inhalte online veröffentlicht werden können. Wir nutzen WordPress, um die Dokumentation auf einer Webseite zu veröffentlichen.

5.4 Arduino

Arduino ist eine Plattform für Elektronik- und Mikrocontroller-Projekte. Diese Plattform umfasst zum einen die Arduino Elektronik-Hardware-Boards und zum

anderen die Arduino Software mit deren Hilfe die Boards programmiert werden können. Wir nutzten die Arduino IDE 2 um unseren Mikrocontroller zu programmieren, damit dieser den Servomotor steuert.

6 Ausblick

Da das Projekt vereinfacht umgesetzt wurde, gibt es weitere Ideen, die es in das Produkt nicht hineingeschafft haben, welche aber dennoch als Ausblick bestehen.

Neben der geplanten Website wäre eine Weiterentwicklung eine Applikation, die neben dem An- und Ausschalten ebenfalls LED-Lichter innerhalb des Produktes ansteuert.

Das Produkt selbst kann mit einem stärkeren Motor und einen kompakteren Gehäuse sowie LED-Lichtern ausgestattet werden. Somit ermöglicht dies, jedes Gerät einschalten zu können und dennoch wenig Platz zu verbrauchen.

Eine Batterie innerhalb des Gehäuses wird das Produkt transportabel und extern machen, sodass man nicht auf eine Stromquelle per USB-Kabel angewiesen ist.

7 Fazit

In diesem Abschnitt wird die Arbeit der letzten Monate reflektiert.

Der Beginn des Projektes wurde durch eine angenehme Arbeitsatmosphäre geschaffen, sodass sich jedes Gruppenmitglied schnell und einfach in die Gruppe eingliedern konnte. Daraus resultierte jeden Mittwochabend ein Treffen auf dem gruppeneigenen Discord-Server per Webcam, um das Projekt zu besprechen.

Aus anfänglichen schwierigkeiten eine Projektidee zu finden wurde durch die Atmosphäre innerhalb der Gruppe demokratisch eine Idee gewählt. Dieses Atmosphäre und das Teamgefühl begleitete unsere Gruppe durch das gesamte Projekt hindurch.
Fragen, die im Laufe einer Sitzung aufgetaucht sind, wurden als Hausaufgabe für die kommende Woche niedergeschrieben, sodass jedes Gruppenmitglied sich mit dem Projekt auseinandergesetzt hat. Diese Methode der Aufgabenstellung hat innerhalb unserer Gruppe eher weniger funktioniert. Der Nachteil war, das viele Aufgaben am Anfang aufgeschoben wurden und somit vor allem am Anfang des Projektes ein geringer Fortschritt zu vermerken war.

Ein Problem war, dass die Idee gut, die Umsetzung jedoch sehr komplex war. Durch diese Komplexität und ein schlechtes Zeitmanagement konnten somit Ideen wie eine Batterie nicht umgesetzt werden.

Lesson learned:

Für zukünftige Projekte wurde gelernt, dass eine geeignete Pufferzeit berücksichtigt werden sollte. Ein weiterer Punkt war das Zeitmanagement, durch welches wir nicht die Möglichkeit hatten, das Projekt wie geplant fertigzustellen. Die Arbeitsteilung war ebenfalls nur grob vorhanden, wodurch das Zusammenfügen der Resultate erschwert wurde. Die Folge daraus war, dass die Resultate in der Theorie jedoch nicht in der Praxis funktioniert haben. Dennoch sind wir an der Herausforderung als Team gewachsen und freuen uns auf weitere Arbeiten.