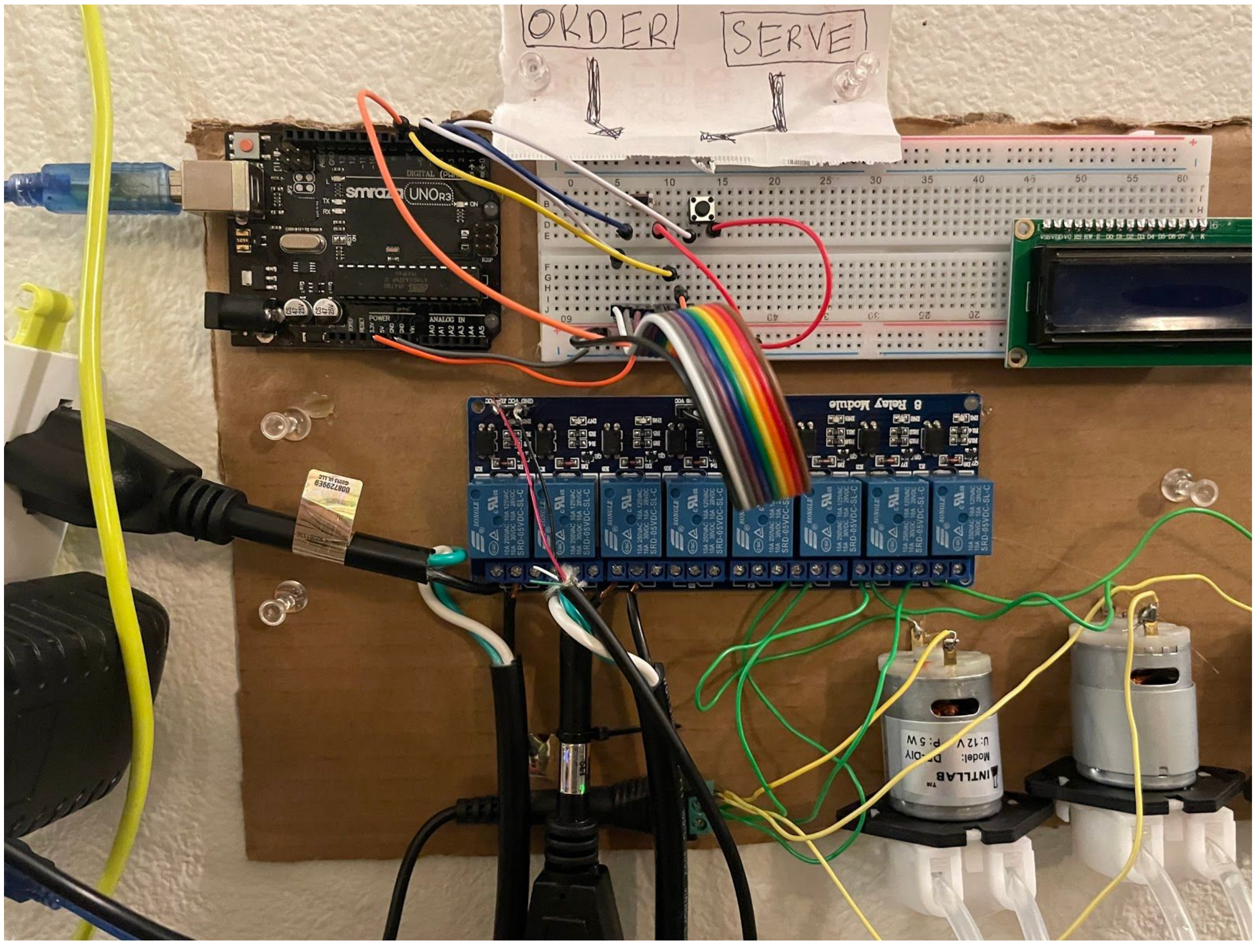
ME 102B Project: Automatic Tea Maker 3000

Project Description

During the day I drink loads of tea because it has caffeine so it often gives me a little bit of a jolt. However, the process of making tea is time consuming and breaks my concentration. If only I could have tea with the press of a button. Well, with this device I can. The Automatic Tea Maker 3000 is able to hold a stock of 3 cups of tea and automatically boils water and creates more stock once the 3 cups are finished. One order button keeps track of how many cups of tea I wanted, and one serve button is used to serve me the tea once I am ready to receive it. I added the serve button because I didnt want the machine squirting tea without me first placing the cup in the respective position.

Setup



Components

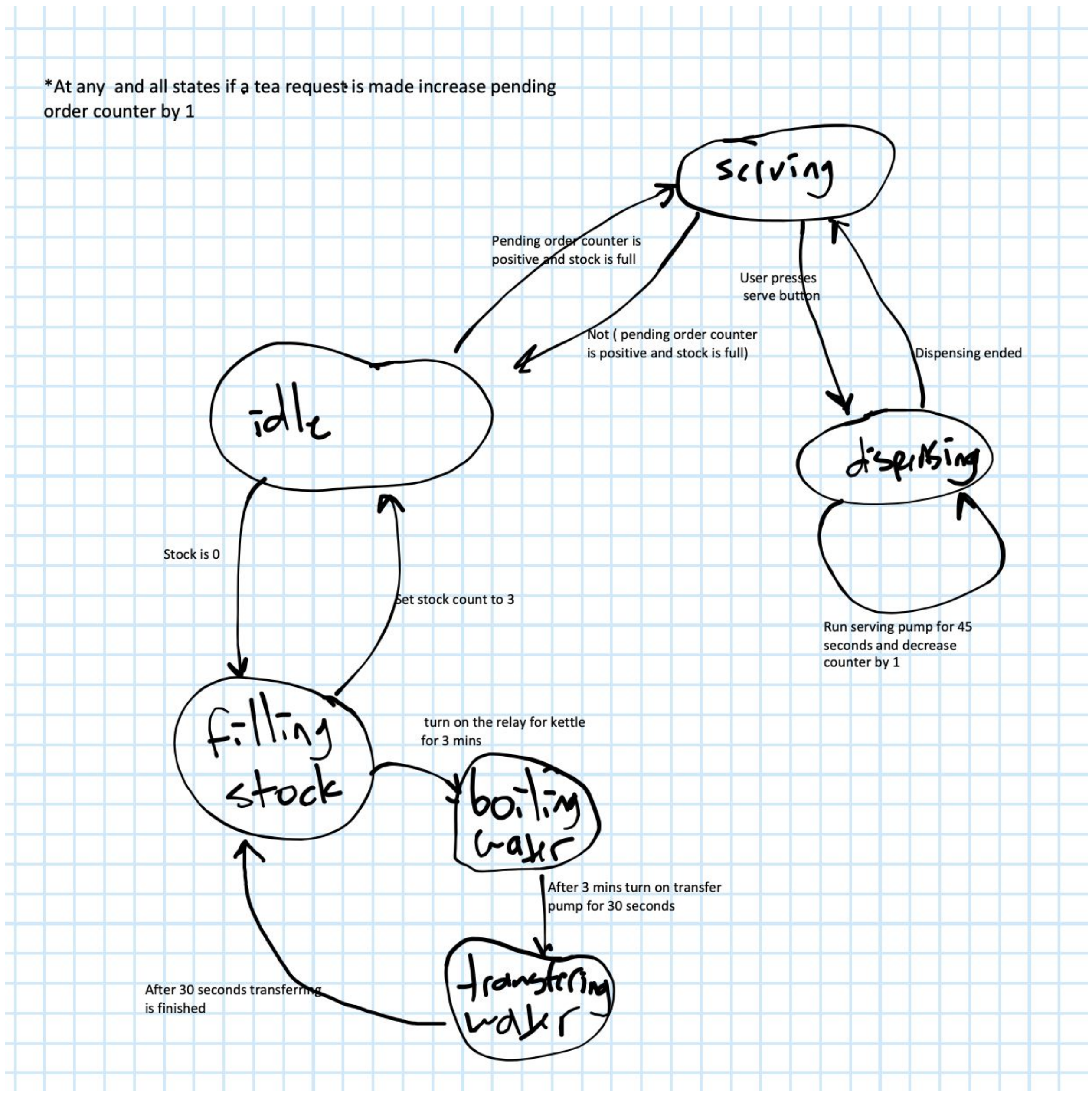
The two non standard devices I'm using here are the peristaltic pump, magnetic stirrer, and the relay device. The screen is also in the picture but I couldn't figure out how to use it so I didn't include it in the final prototype.

Relay: This is a device used to control other devices that run on higher power than the arduino’s max. I'm using it to control the 2 peristaltic pumps and the kettle. They run on 12V and 110V respectively. The relay is essentially a switch so I had to split the hot wire of the kettle and connect it to the 2 ends of the relay, and for the pumps I have the same set up but I was able to just use loose wires. When the relay receives a low signal on one of the pins located on the upper side of it, it connects the wires of the respective device to be turned on. You will also notice that I have a separate 5V supply for the relay device. I was reading online that this is good practice in case something goes wrong with the relay mechanism. This way your arduino and computer will not be subjected to the high voltage and currents that the relay controls.

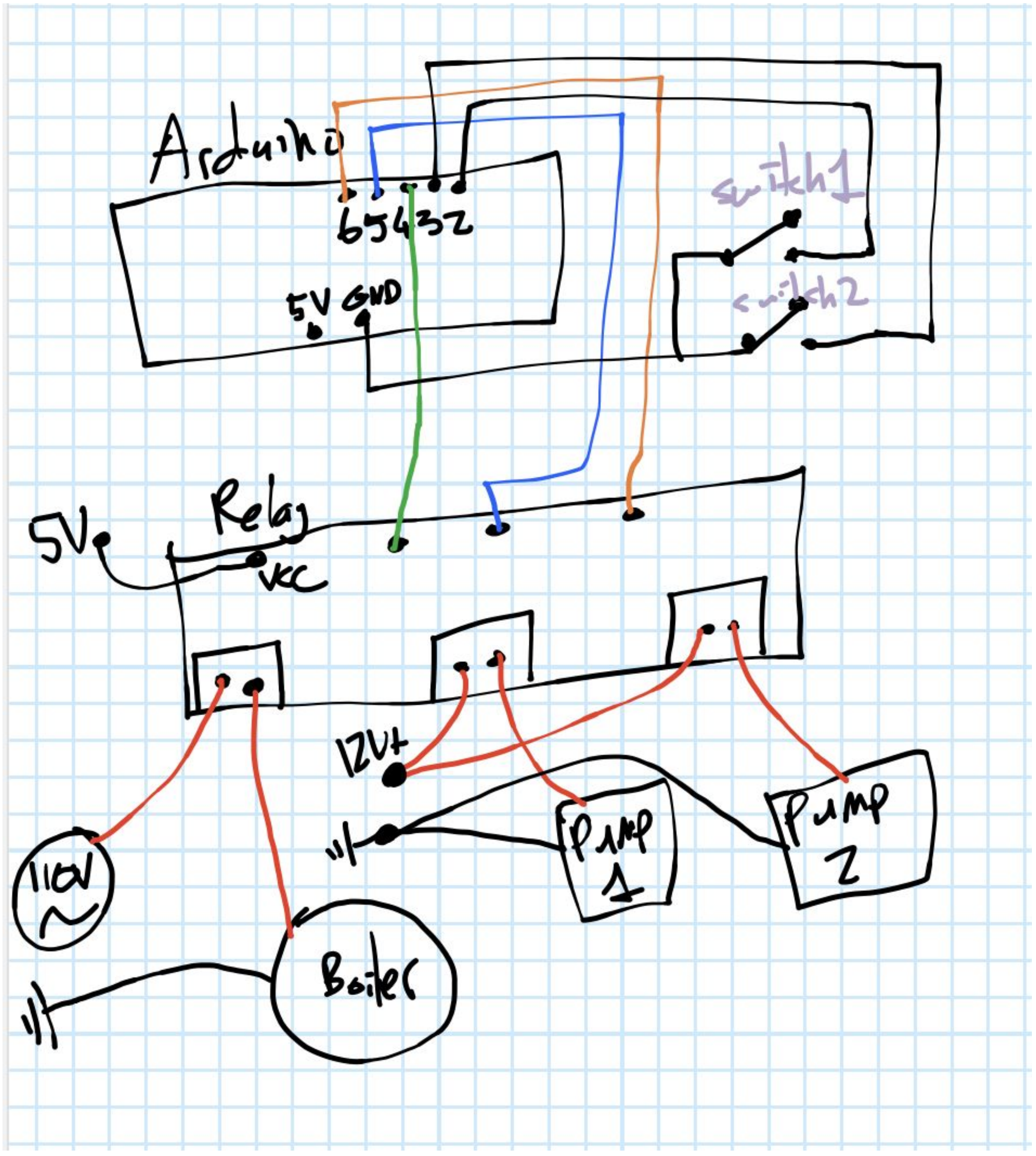
Peristaltic Pumps: These pumps are really no special than any other pump, they just carry fluid from one place to another. The advantage of these is that they do not need priming. Priming of the pump is the process of getting rid of all the air bubbles and cavities. Traditional pumps cannot run this way because they cannot pump air. Peristaltic pump works using peristalsis so it is able to also pump air, although not very good at it. This ability makes it possible for the pump to pump out all the air before getting to the fluid, therefore self priming. It makes it the ideal pump to use for little fluid transfers like this. One disadvantage is that it is much slower than a traditional pump.

Magnetic Stirrer: I am not really controlling this device but it is still a part of my system. I saw this device first in chemistry class and never thought I would use it. It basically creates a rotating magnetic field and there is a spinner that comes with it which you can put into the container you are intending to stir. The spinner is magnetic so it rotates with the field, stirring the liquid.

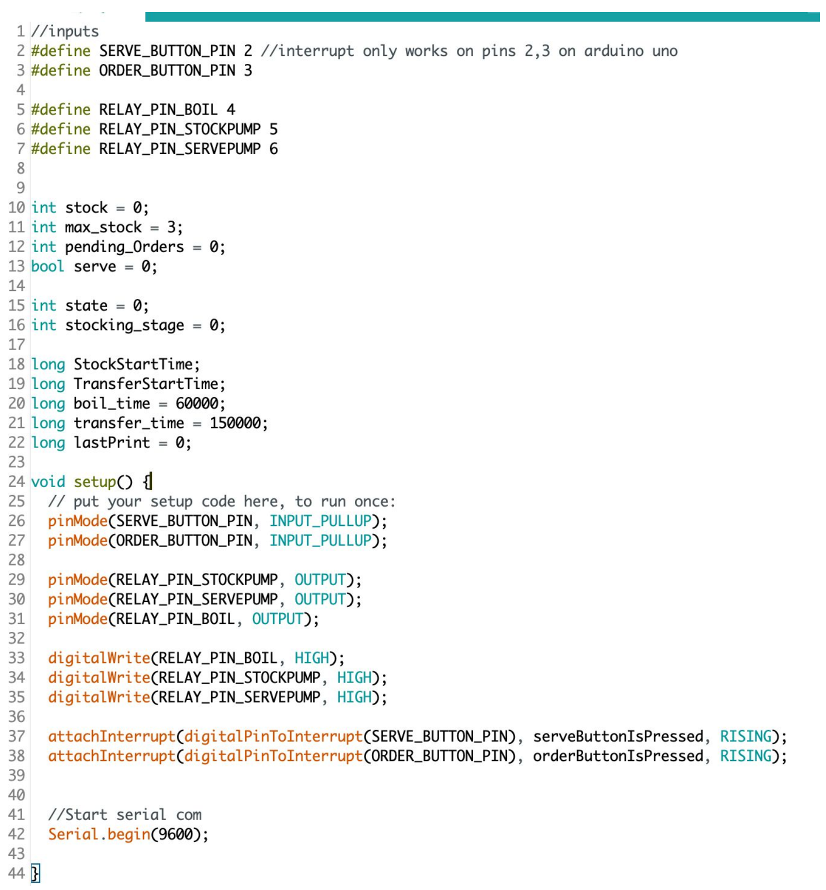
Finite State Diagram


*At any and all states if a tea request is made increase pending order counter by 1
serving
Pending order counter is positive and stock is full
Not (pending order counter is positive and stock is full)
User presses serve button
Dispensing ended
idle
dispensing
Run serving pump for 45 seconds and decrease counter by 1
Stock is 0
Set stock count to 3
filling stock
turn on the relay for kettle for 3 mins
boiling water
After 3 mins turn on transfer pump for 30 seconds
transferring water
After 30 seconds transferring is finished

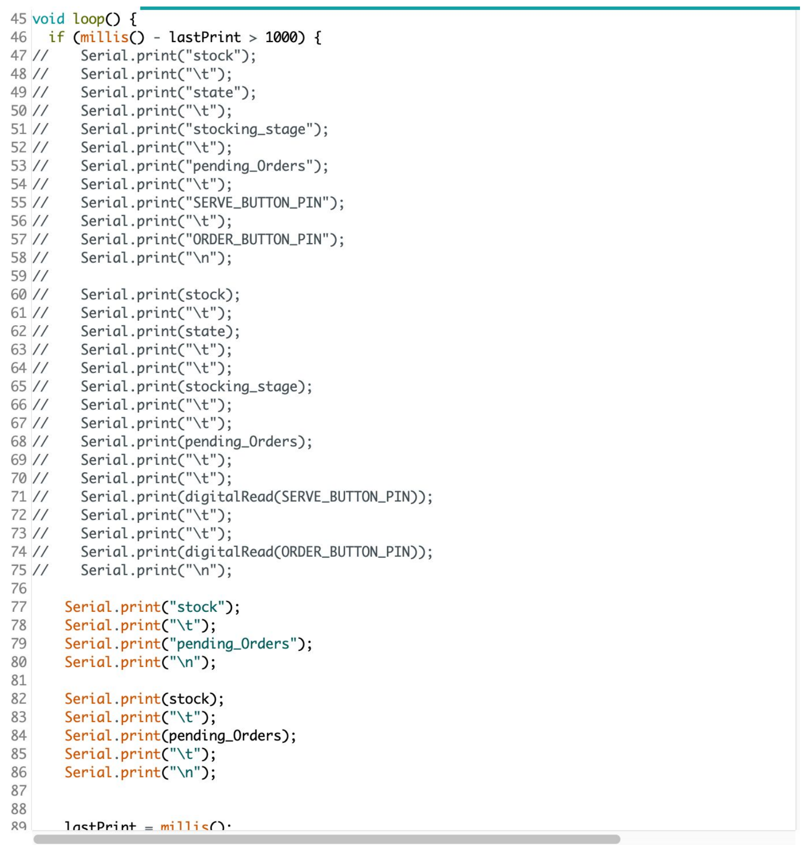
Circuit Diagram



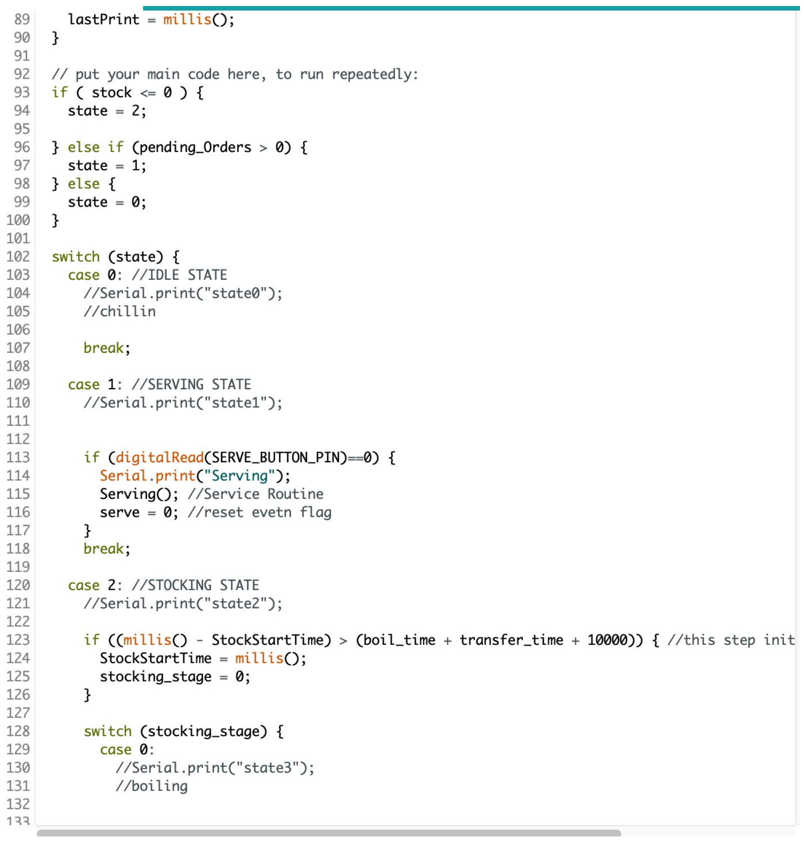
Appendix



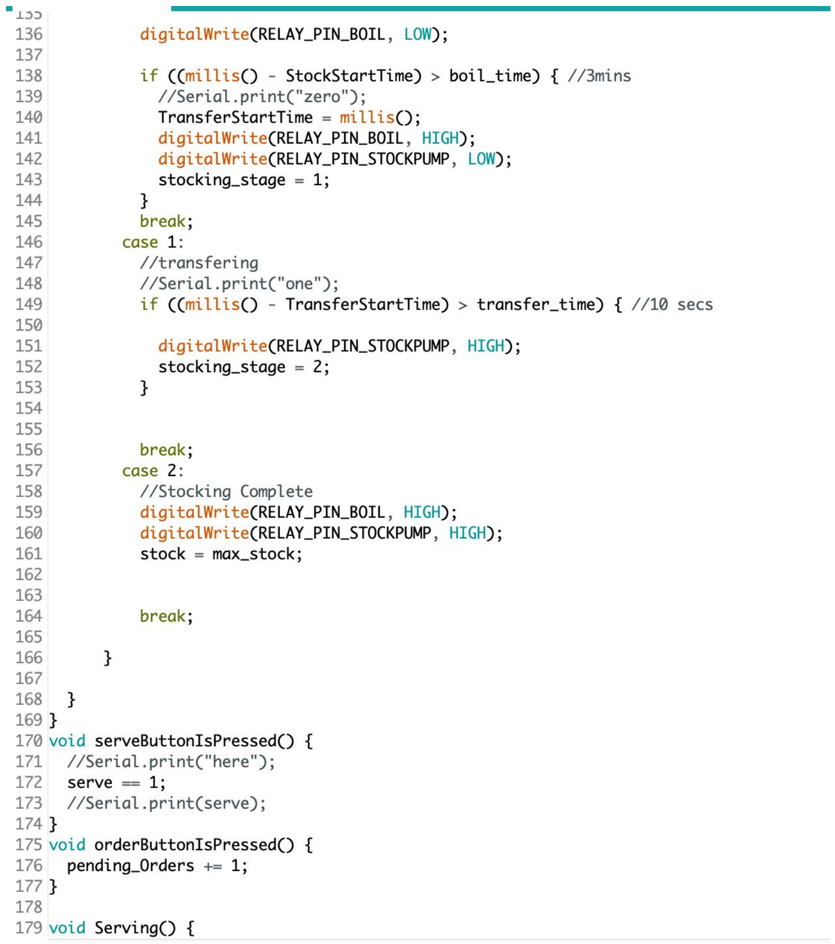
```
//inputs
#define SERVE_BUTTON_PIN 2 //interrupt only works on pins 2,3 on arduino uno
#define ORDER_BUTTON_PIN 3

#define RELAY_PIN_BOIL 4
#define RELAY_PIN_STOCKPUMP 5
#define RELAY_PIN_SERVEPUMP 6


int stock = 0;
int max_stock = 3;
int pending_Orders = 0;
bool serve = 0;

int state = 0;
int stocking_stage = 0;

long StockStartTime;
long TransferStartTime;
long boil_time = 60000;
long transfer_time = 150000;
long lastPrint = 0;

void setup() {
  // put your setup code here, to run once:
  pinMode(SERVE_BUTTON_PIN, INPUT_PULLUP);
  pinMode(ORDER_BUTTON_PIN, INPUT_PULLUP);

  pinMode(RELAY_PIN_STOCKPUMP, OUTPUT);
  pinMode(RELAY_PIN_SERVEPUMP, OUTPUT);
  pinMode(RELAY_PIN_BOIL, OUTPUT);

  digitalWrite(RELAY_PIN_BOIL, HIGH);
  digitalWrite(RELAY_PIN_STOCKPUMP, HIGH);
  digitalWrite(RELAY_PIN_SERVEPUMP, HIGH);

  attachInterrupt(digitalPinToInterrupt(SERVE_BUTTON_PIN), serveButtonIsPressed, RISING);
  attachInterrupt(digitalPinToInterrupt(ORDER_BUTTON_PIN), orderButtonIsPressed, RISING);


  //Start serial com
  Serial.begin(9600);

}
```



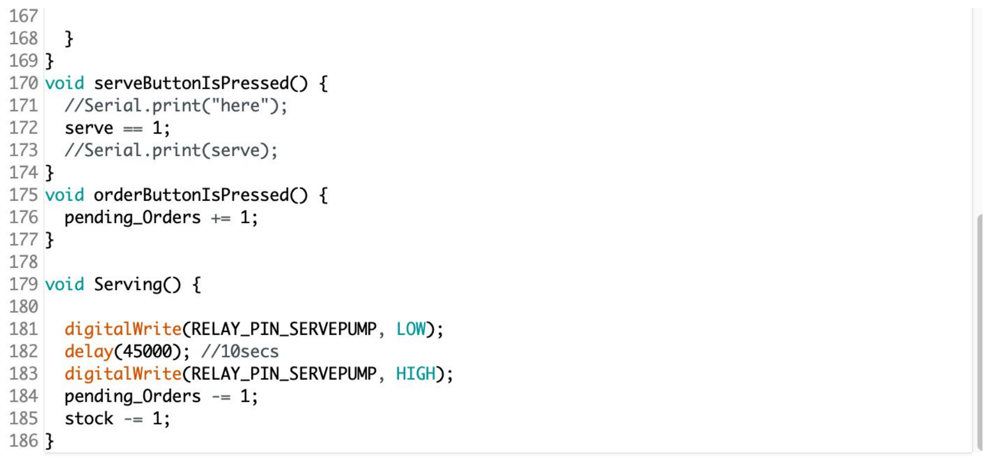
```
void loop() {
  if (millis() - lastPrint > 1000) {
//    Serial.print("stock");
//    Serial.print("\t");
//    Serial.print("state");
//    Serial.print("\t");
//    Serial.print("stocking_stage");
//    Serial.print("\t");
//    Serial.print("pending_Orders");
//    Serial.print("\t");
//    Serial.print("SERVE_BUTTON_PIN");
//    Serial.print("\t");
//    Serial.print("ORDER_BUTTON_PIN");
//    Serial.print("\n");
//
//    Serial.print(stock);
//    Serial.print("\t");
//    Serial.print(state);
//    Serial.print("\t");
//    Serial.print("\t");
//    Serial.print(stocking_stage);
//    Serial.print("\t");
//    Serial.print("\t");
//    Serial.print(pending_Orders);
//    Serial.print("\t");
//    Serial.print("\t");
//    Serial.print(digitalRead(SERVE_BUTTON_PIN));
//    Serial.print("\t");
//    Serial.print("\t");
//    Serial.print(digitalRead(ORDER_BUTTON_PIN));
//    Serial.print("\n");

    Serial.print("stock");
    Serial.print("\t");
    Serial.print("pending_Orders");
    Serial.print("\n");

    Serial.print(stock);
    Serial.print("\t");
    Serial.print(pending_Orders);
    Serial.print("\t");
    Serial.print("\n");


```



```
    lastPrint = millis();
  }

  // put your main code here, to run repeatedly:
  if ( stock <= 0 ) {
    state = 2;

  } else if (pending_Orders > 0) {
    state = 1;
  } else {
    state = 0;
  }

  switch (state) {
    case 0: //IDLE STATE
      //Serial.print("state0");
      //chillin

      break;

    case 1: //SERVING STATE
      //Serial.print("state1");


      if (digitalRead(SERVE_BUTTON_PIN)==0) {
        Serial.print("Serving");
        Serving(); //Service Routine
        serve = 0; //reset evetn flag
      }
      break;

    case 2: //STOCKING STATE
      //Serial.print("state2");

      if ((millis() - StockStartTime) > (boil_time + transfer_time + 10000)) { //this step init
        StockStartTime = millis();
        stocking_stage = 0;
      }

      switch (stocking_stage) {
        case 0:
          //Serial.print("state3");
          //boiling

```



```
          digitalWrite(RELAY_PIN_BOIL, LOW);

          if ((millis() - StockStartTime) > boil_time) { //3mins
            //Serial.print("zero");
            TransferStartTime = millis();
            digitalWrite(RELAY_PIN_BOIL, HIGH);
            digitalWrite(RELAY_PIN_STOCKPUMP, LOW);
            stocking_stage = 1;
          }
          break;
        case 1:
          //transfering
          //Serial.print("one");
          if ((millis() - TransferStartTime) > transfer_time) { //10 secs

            digitalWrite(RELAY_PIN_STOCKPUMP, HIGH);
            stocking_stage = 2;
          }


          break;
        case 2:
          //Stocking Complete
          digitalWrite(RELAY_PIN_BOIL, HIGH);
          digitalWrite(RELAY_PIN_STOCKPUMP, HIGH);
          stock = max_stock;


          break;

      }

  }
}
void serveButtonIsPressed() {
  //Serial.print("here");
  serve == 1;
  //Serial.print(serve);
}
void orderButtonIsPressed() {
  pending_Orders += 1;
}

void Serving() {
```



```

  }
}
void serveButtonIsPressed() {
  //Serial.print("here");
  serve == 1;
  //Serial.print(serve);
}
void orderButtonIsPressed() {
  pending_Orders += 1;
}

void Serving() {
```

```

  digitalWrite(RELAY_PIN_SERVEPUMP, LOW);
  delay(45000); //10secs
  digitalWrite(RELAY_PIN_SERVEPUMP, HIGH);
  pending_Orders -= 1;
  stock -= 1;
}
```