I used a Raspberry PI along with sensors, actuators, and accessories to create a website that controls a Raspberry PI Camera and a Servo motor. The Raspberry PI 3B+ runs Raspberry PI OS (previously called Raspbian.) I connected the Raspberry PI to a solderless breadboard using a 26-pin ribbon cable and T-Cobbler breakout, used to ease GPIO pin access. For sensors, I used a Raspberry PI Camera Module V2. For actuators, I used a SG90 Servo motor positioned next to an automatic dice roller. During early iterations of the project, I also used a four-by-four membrane keypad and three LEDs which were all phased out for the final implementation.

**Goal**

To create a networked dice roller with camera capabilities. The idea behind this project comes from one of my favorite hobbies, playing Dungeons and Dragons with my friends. While most of the game is played with pencils, paper, and your imagination, the most fun accessories are the dice. Players buy dice in their favorite colors and grow superstitious about “lucky” dice. Some players even spend $50-$100 dollars buying dice made from hand-carved woods, gemstones, or metals. We bring a few sets of dice with us when we play at friends’ houses, usually our newest sets or a set that thematically matches the character we are playing. If this project goes according to plan, I will always have my favorite dice with me.

**Design and Implementation**

Diagram, schematic

Description automatically generated

The project design centers around controlling a Servo motor and viewing a video stream over the internet. As suggested in class, I implemented this project in three successive stages of complexity.

Stage 1: Controlling the Servo

To begin, I recycled code from my first project to set up the Servo control. I programmed a 4x4 membrane keyboard to control the Servo and LEDs. During this stage, I ran into my first major hurdle in the Servo being capable of pressing the button of the automatic dice roller, more details can be found in the difficulties section at the end of this report.

Stage 2: Video Feed

Once the Servo was set up and capable of activating the automatic dice roller, the next step was the video feed. At this stage, I came across a Python library for streaming a Raspberry Pi Camera over the internet through a Flask server. After downloading Flask and learning the basics, I modified a version of this library to set up my video feed (<https://github.com/miguelgrinberg/flask-video-streaming>.)

Stage 3: Flask Server

This final stage is where the bulk of the project work was done. During stage 2 I had learned how to use Flask to stream my video feed and found that Flask was going to be an ideal utility for my Servo control as well. Flask ended up serving two purposes for my project. First, it served my website which was a convenient feature that saved me from having to use Apache, NGINX, or the less reliable built-in Python SimpleHTTPServer. Second, and more importantly, Flask acted as the interface between by front-end HTML and my back-end Python server scripts. Looking at the last line of the HTML below, we see that a button is defined with href “/roll/70”

Text

Description automatically generated

Next, in the Flask interface code below we see that it deconstructs this route into two variables “/<servo>” which is given in HTML as “/roll” and “/<angle>” which is given as “/70” After pulling these two variables out of the path, the interface called a server side script by concatenating the variables onto an os.system script call, as seen in the last line of diceRoller.py below.

Text

Description automatically generated

Finally, the snippet below of angleServoCtrl.py contains the function being called on the server.

Text

Description automatically generated

This design of HTML -> Flask -> Python script is also used for the video feed, as can be seen in the HTML screenshot above. In this case, the route “/video\_feed” communicates with camera\_pi.py to get a continuous stream of images from the Pi Camera for the video feed.

With these two pieces set up, the project consists of starting the Flask server from diceRoller.py which serves my index.html on the Raspberry Pi’s IP address over port 80. When I visit this site from another computer on my network, I can see a continuous video feed from the PI camera. If I then press the “roll” button on the site, the servo presses down on the automatic dice roller, and I can view the rolling and results live.

Photos of (working) prototype dice roller with camera:  
A toy building blocks with different colored wires

AI-generated content may be incorrect.

A toy building blocks with different dice inside

AI-generated content may be incorrect.