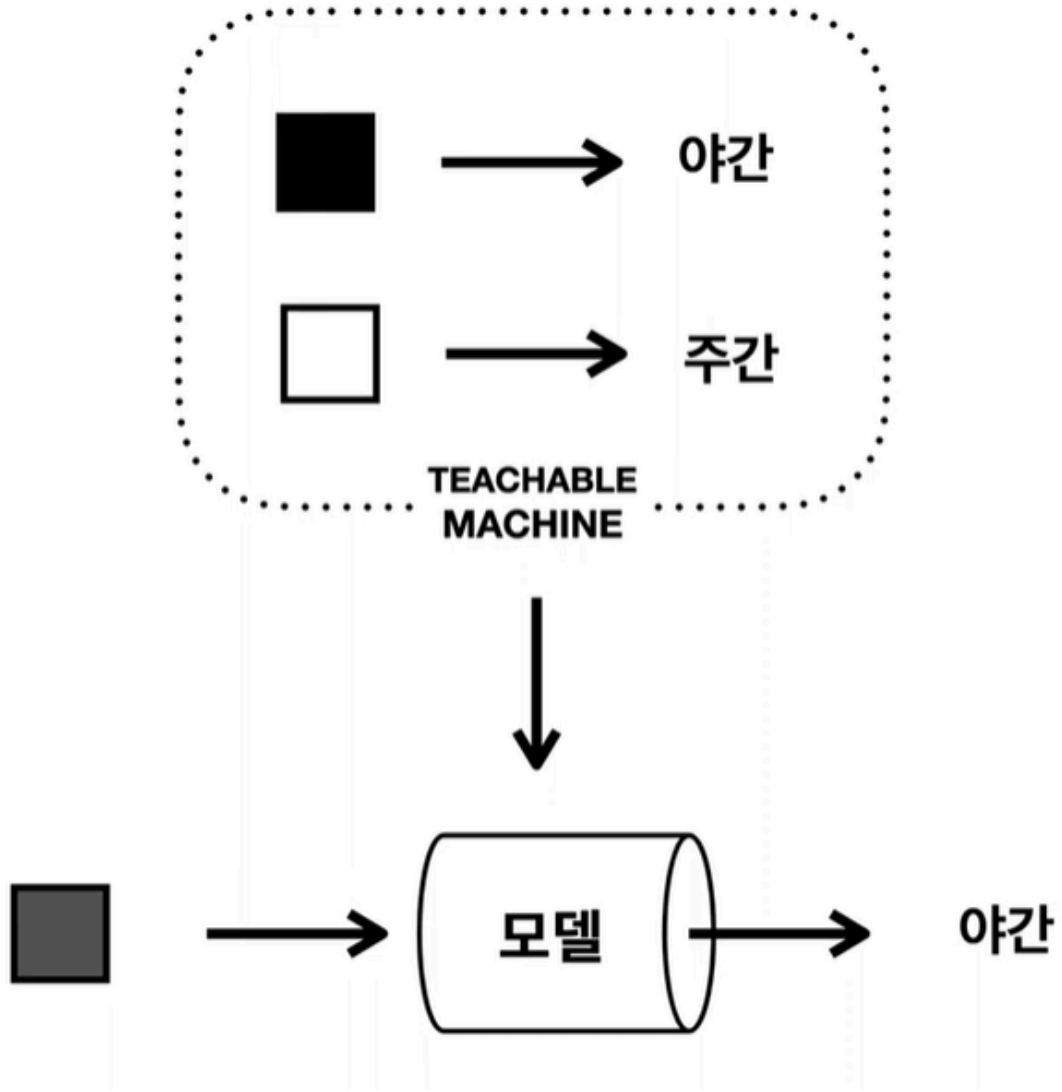
Teachable machine & WEB

목표 : 주변이 밝으면 주간 모드, 주변이 어두우면 야간 모드로 동작하는 웹 사이트



https://teachablemachine.withgoogle.com/models/Dw3KJjAXn/

```
<div>Teachable Machine Image Model</div>
<button type="button" onclick="init()">Start</button>
<div id="webcam-container"></div>
<div id="label-container"></div>
<script src="https://cdn.jsdelivr.net/npm/@tensorflow/tfjs@latest/dist/tf.min.js
<script src="https://cdn.jsdelivr.net/npm/@teachablemachine/image@latest/d
```

```
<script type="text/javascript">
    // More API functions here:
    // https://github.com/googlecreativelab/teachablemachine-community/tree/

    // the link to your model provided by Teachable Machine export panel
    const URL = "https://teachablemachine.withgoogle.com/models/Dw3KJjAX

    let model, webcam, labelContainer, maxPredictions;

    // Load the image model and setup the webcam
    async function init() {
        const modelURL = URL + "model.json";
        const metadataURL = URL + "metadata.json";

        // load the model and metadata
        // Refer to tmImage.loadFromFiles() in the API to support files from a file p
        // or files from your local hard drive
        // Note: the pose library adds "tmImage" object to your window (window.
        model = await tmImage.load(modelURL, metadataURL);
        maxPredictions = model.getTotalClasses();

        // Convenience function to setup a webcam
        const flip = true; // whether to flip the webcam
        webcam = new tmImage.Webcam(200, 200, flip); // width, height, flip
        await webcam.setup(); // request access to the webcam
        await webcam.play();
        window.requestAnimationFrame(loop);

        // append elements to the DOM
        document.getElementById("webcam-container").appendChild(webcam.c
        labelContainer = document.getElementById("label-container");
        for (let i = 0; i < maxPredictions; i++) { // and class labels
            labelContainer.appendChild(document.createElement("div"));
        }
    }

    async function loop() {
        webcam.update(); // update the webcam frame
```

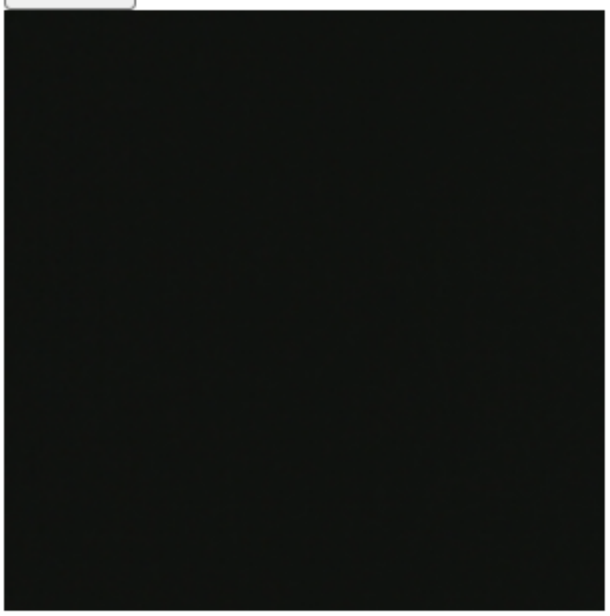
```
        await predict();
        window.requestAnimationFrame(loop);
    }

    // run the webcam image through the image model
    async function predict() {
        // predict can take in an image, video or canvas html element
        const prediction = await model.predict(webcam.canvas);
        for (let i = 0; i < maxPredictions; i++) {
            const classPrediction =
                prediction[i].className + ": " + prediction[i].probability.toFixed(2);
            labelContainer.childNodes[i].innerHTML = classPrediction;
        }
    }
</script>
```

Teachable machine의 이미지 인식 기능은 https 통신 환경에서만 활성화

웹 캠을 사용하기 때문에 F5나 F11로 접근이 가능하면 안됨

Teachable Machine Image Model

Start

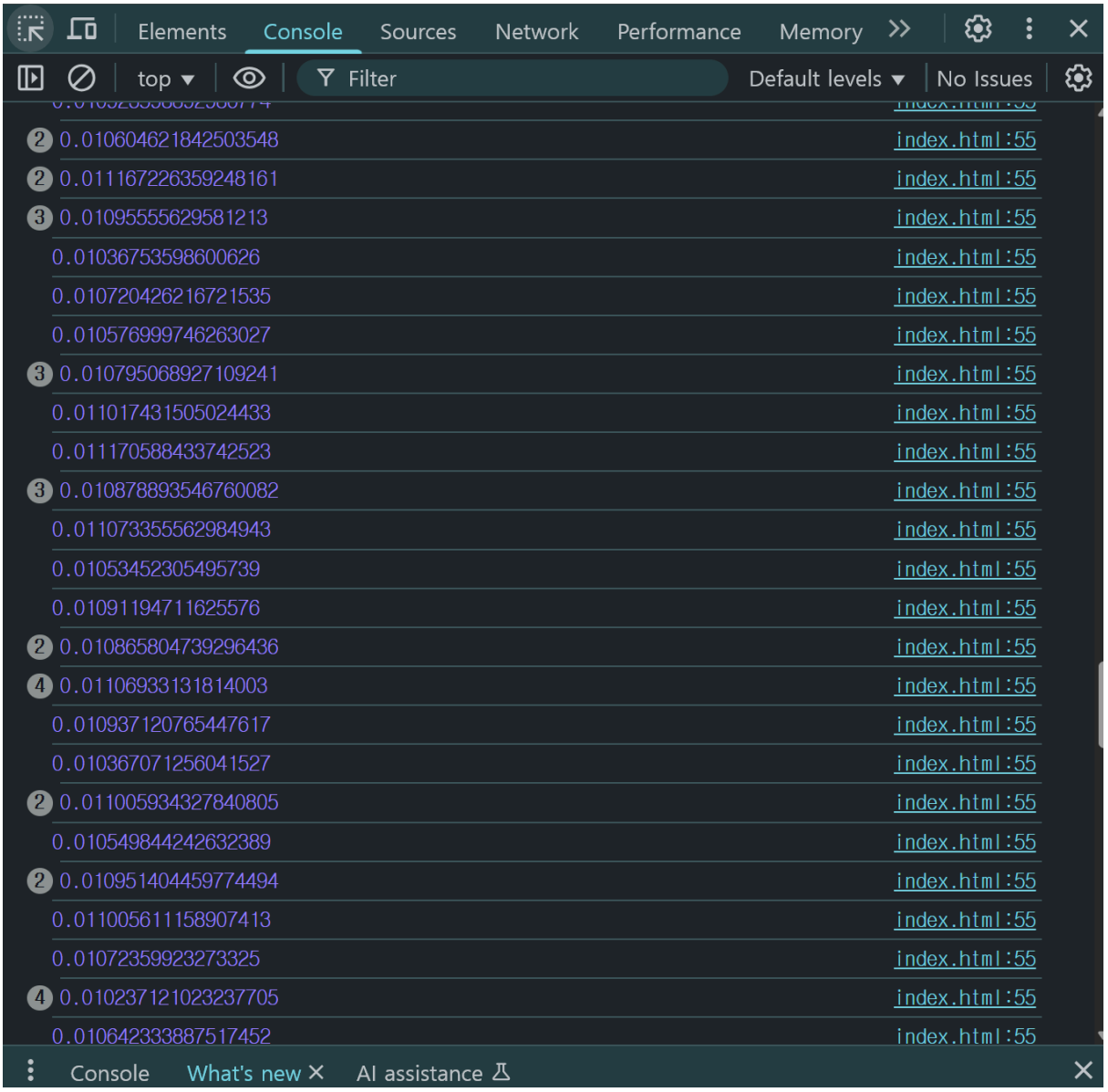


Bright: 0.00

Dark: 1.00

Teachable Machine Image Model

Start



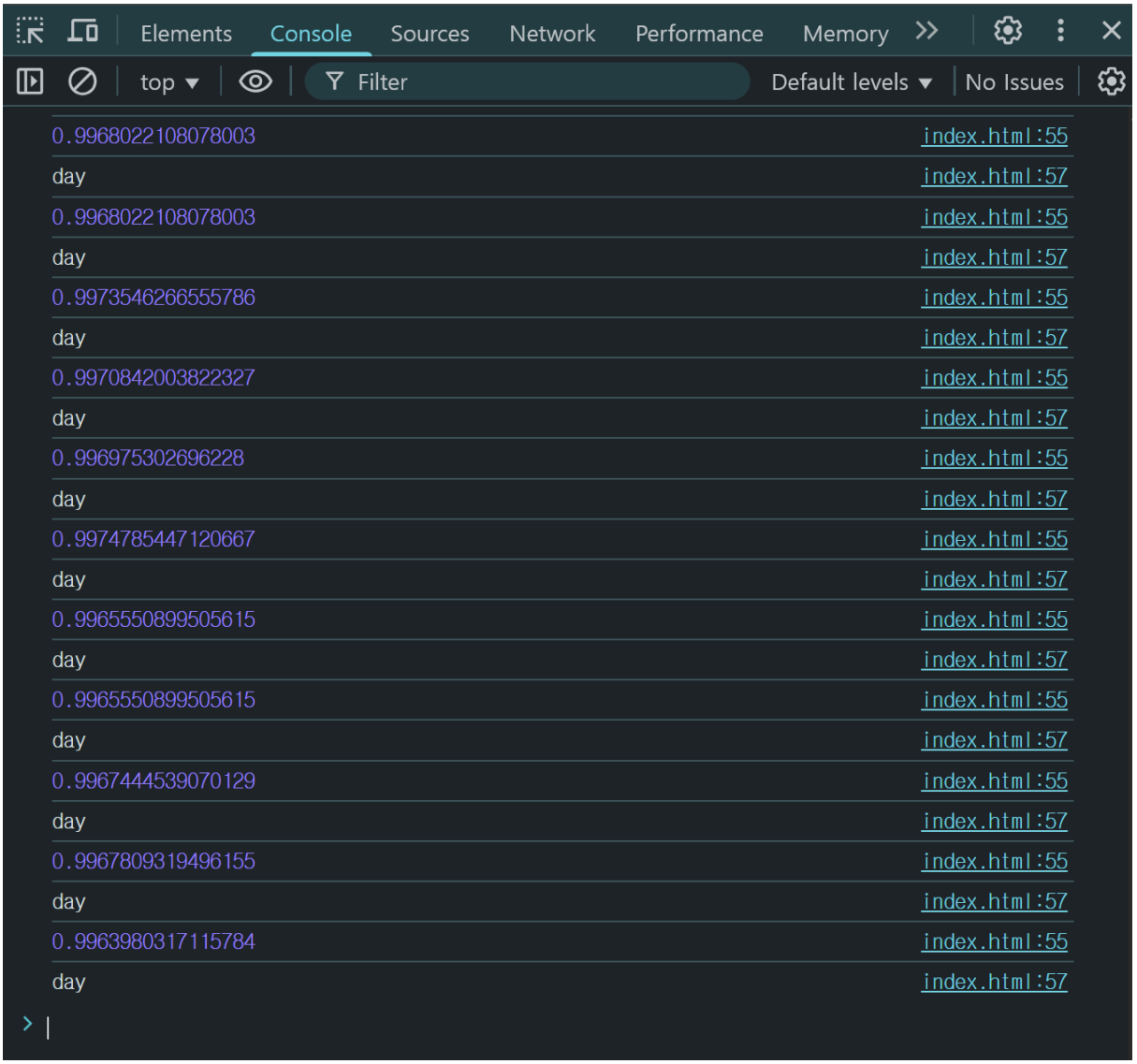
console.log(prediction[0].probability); 0번째 원소의 probability를 console에 찍는다

```
<!-- index.html -->
<!DOCTYPE html>
<div>Teachable Machine Image Model</div>
<button type="button" onclick="init()">Start</button>
<div id="webcam-container"></div>
<div id="label-container"></div>
<script src="https://cdn.jsdelivr.net/npm/@tensorflow/tfjs@latest/dist/tf.min.js
<script src="https://cdn.jsdelivr.net/npm/@teachablemachine/image@latest/d
```

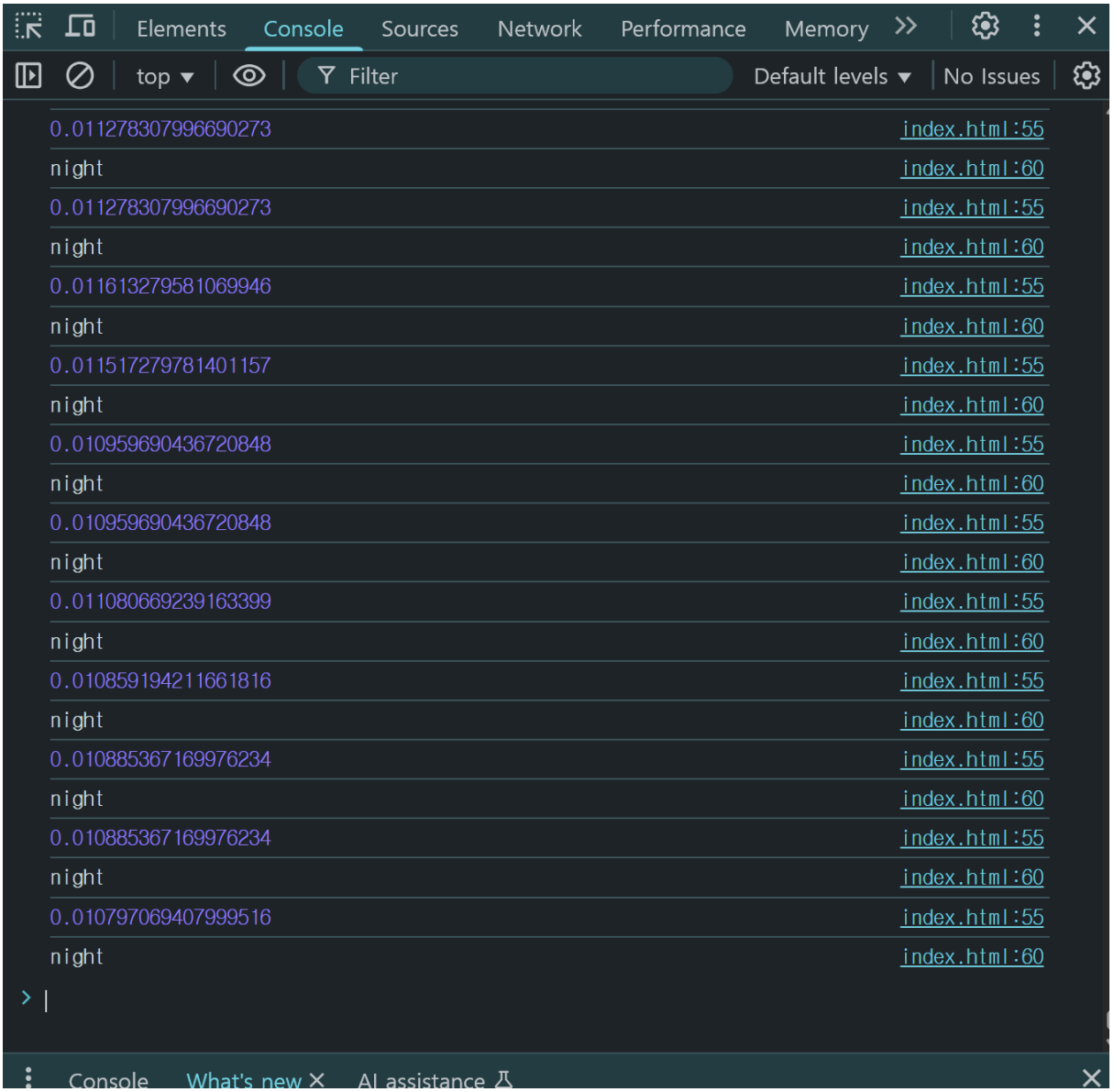
```
<script type="text/javascript">
    // More API functions here:
    // https://github.com/googlecreativelab/teachablemachine-community/tree/

    // the link to your model provided by Teachable Machine export panel
    const URL = "https://teachablemachine.withgoogle.com/models/Dw3KJjAX

    let model, webcam, labelContainer, maxPredictions;

    // Load the image model and setup the webcam
    async function init() {
        const modelURL = URL + "model.json";
        const metadataURL = URL + "metadata.json";

        // load the model and metadata
        // Refer to tmImage.loadFromFiles() in the API to support files from a file p
        // or files from your local hard drive
        // Note: the pose library adds "tmImage" object to your window (window.
        model = await tmImage.load(modelURL, metadataURL);
        maxPredictions = model.getTotalClasses();

        // Convenience function to setup a webcam
        const flip = true; // whether to flip the webcam
        webcam = new tmImage.Webcam(200, 200, flip); // width, height, flip
        await webcam.setup(); // request access to the webcam
        await webcam.play();
        window.requestAnimationFrame(loop);

        // append elements to the DOM
        document.getElementById("webcam-container").appendChild(webcam.c
        labelContainer = document.getElementById("label-container");
        for (let i = 0; i < maxPredictions; i++) { // and class labels
            labelContainer.appendChild(document.createElement("div"));
        }
    }

    async function loop() {
        webcam.update(); // update the webcam frame
```

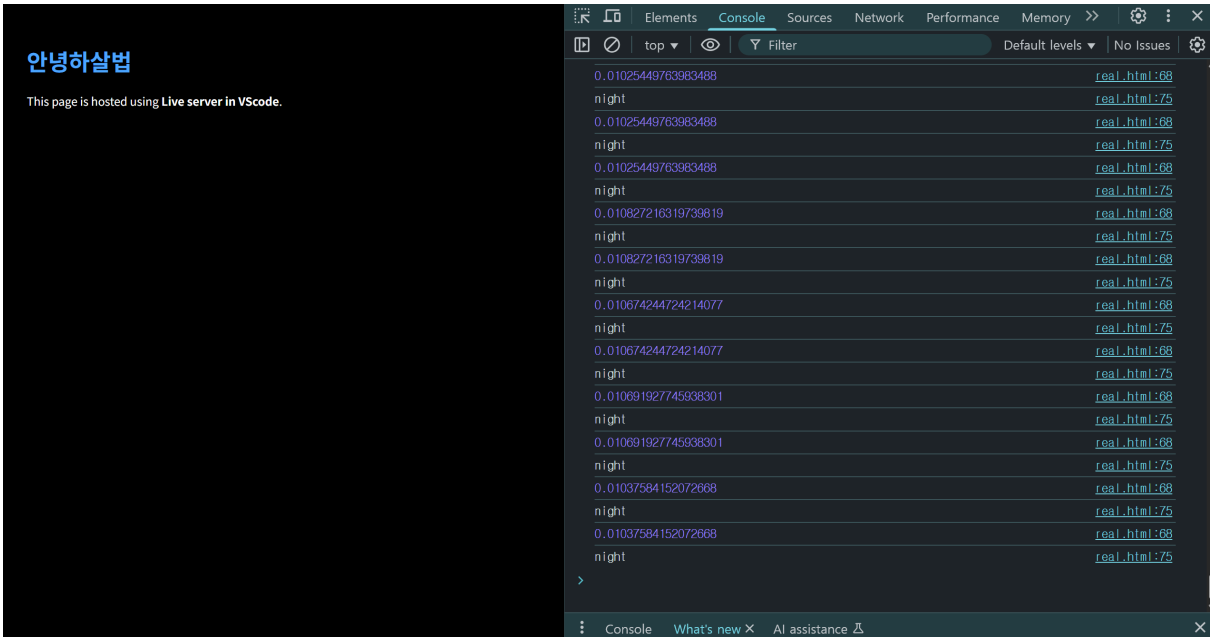
```
        await predict();
        window.requestAnimationFrame(loop);
    }

    // run the webcam image through the image model
    async function predict() {
        // predict can take in an image, video or canvas html element
        const prediction = await model.predict(webcam.canvas);
        console.log(prediction[0].probability);
        if(prediction[0].probability>0.5) {
          console.log('day');
        }
        else {
          console.log('night');
        }
        for (let i = 0; i < maxPredictions; i++) {
            const classPrediction =
                prediction[i].className + ": " + prediction[i].probability.toFixed(2);
            labelContainer.childNodes[i].innerHTML = classPrediction;
        }
    }
</script>
```

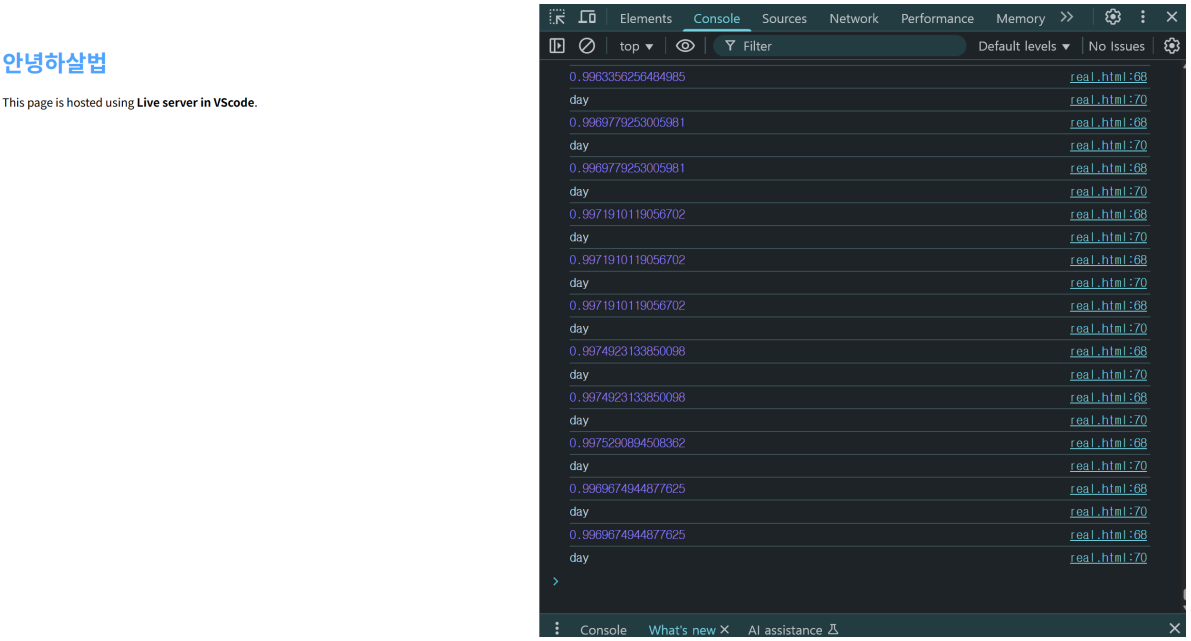
밝을 때


Elements
Console
Sources
Network
Performance
Memory
top
Filter
Default levels
No Issues
0.9968022108078003 index.html:55
day index.html:57
0.9968022108078003 index.html:55
day index.html:57
0.9973546266555786 index.html:55
day index.html:57
0.9970842003822327 index.html:55
day index.html:57
0.996975302696228 index.html:55
day index.html:57
0.9974785447120667 index.html:55
day index.html:57
0.9965550899505615 index.html:55
day index.html:57
0.9965550899505615 index.html:55
day index.html:57
0.9967444539070129 index.html:55
day index.html:57
0.9967809319496155 index.html:55
day index.html:57
0.9963980317115784 index.html:55
day index.html:57

어두울 때


Elements
Console
Sources
Network
Performance
Memory
top
Filter
Default levels
No Issues
0.011278307996690273 index.html:55
night index.html:60
0.011278307996690273 index.html:55
night index.html:60
0.011613279581069946 index.html:55
night index.html:60
0.011517279781401157 index.html:55
night index.html:60
0.010959690436720848 index.html:55
night index.html:60
0.010959690436720848 index.html:55
night index.html:60
0.011080669239163399 index.html:55
night index.html:60
0.010859194211661816 index.html:55
night index.html:60
0.010885367169976234 index.html:55
night index.html:60
0.010885367169976234 index.html:55
night index.html:60
0.010797069407999516 index.html:55
night index.html:60
Console
What's new
AI assistance






```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>하이</title>
  <style>
    body {
      background-color: #111;
      color: #eee;
```

```
    font-family: sans-serif;
    padding: 30px;
   }
   h1 {
    color: #4da6ff;
   }
  </style>
 </head>
 <body>
  <h1>안녕하살법</h1>
  <p>This page is hosted using <strong>Live server in VScode</strong>.</p>

  <div id="webcam-container" style="visibility:hidden; position: absolute;"></
  <div id="label-container"></div>
  <script src="https://cdn.jsdelivr.net/npm/@tensorflow/tfjs@latest/dist/tf.min.
  <script src="https://cdn.jsdelivr.net/npm/@teachablemachine/image@latest/
  <script type="text/javascript">
   // More API functions here:
   // https://github.com/googlecreativelab/teachablemachine-community/tree/

   // the link to your model provided by Teachable Machine export panel
   const URL = "https://teachablemachine.withgoogle.com/models/Dw3KJjAX

   let model, webcam, labelContainer, maxPredictions;

   // Load the image model and setup the webcam
   async function init() {
      const modelURL = URL + "model.json";
      const metadataURL = URL + "metadata.json";

      // load the model and metadata
      // Refer to tmImage.loadFromFiles() in the API to support files from a file p
      // or files from your local hard drive
      // Note: the pose library adds "tmImage" object to your window (window.
      model = await tmImage.load(modelURL, metadataURL);
      maxPredictions = model.getTotalClasses();

      // Convenience function to setup a webcam
```

```
        const flip = true; // whether to flip the webcam
        webcam = new tmImage.Webcam(200, 200, flip); // width, height, flip
        await webcam.setup(); // request access to the webcam
        await webcam.play();
        window.requestAnimationFrame(loop);

        // append elements to the DOM
        document.getElementById("webcam-container").appendChild(webcam.c
    }

    async function loop() {
        webcam.update(); // update the webcam frame
        await predict();
        window.requestAnimationFrame(loop);
    }

    // run the webcam image through the image model
    async function predict() {
        // predict can take in an image, video or canvas html element
        const prediction = await model.predict(webcam.canvas);
        console.log(prediction[0].probability);
        if(prediction[0].probability>0.5) {
          document.querySelector('body').style.backgroundColor = 'white';
          document.querySelector('body').style.color = 'black';
        }
        else {
          document.querySelector('body').style.backgroundColor = 'black';
          document.querySelector('body').style.color = 'white';
        }
    }
    init();
  </script>


</body>
</html>
```

최종 코드 끝!