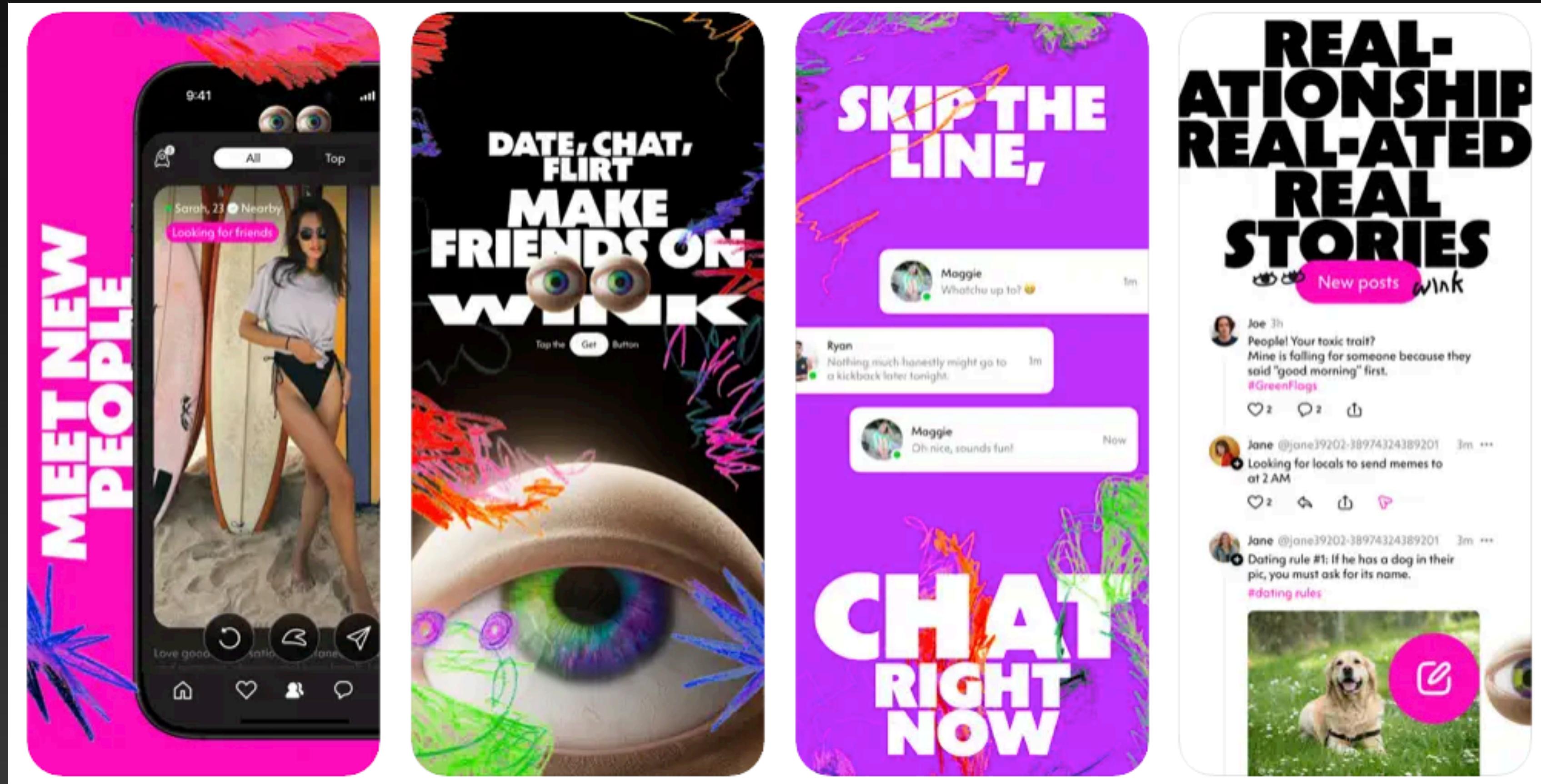


Realtime Face Liveness Detection using TrueDepth Sensor

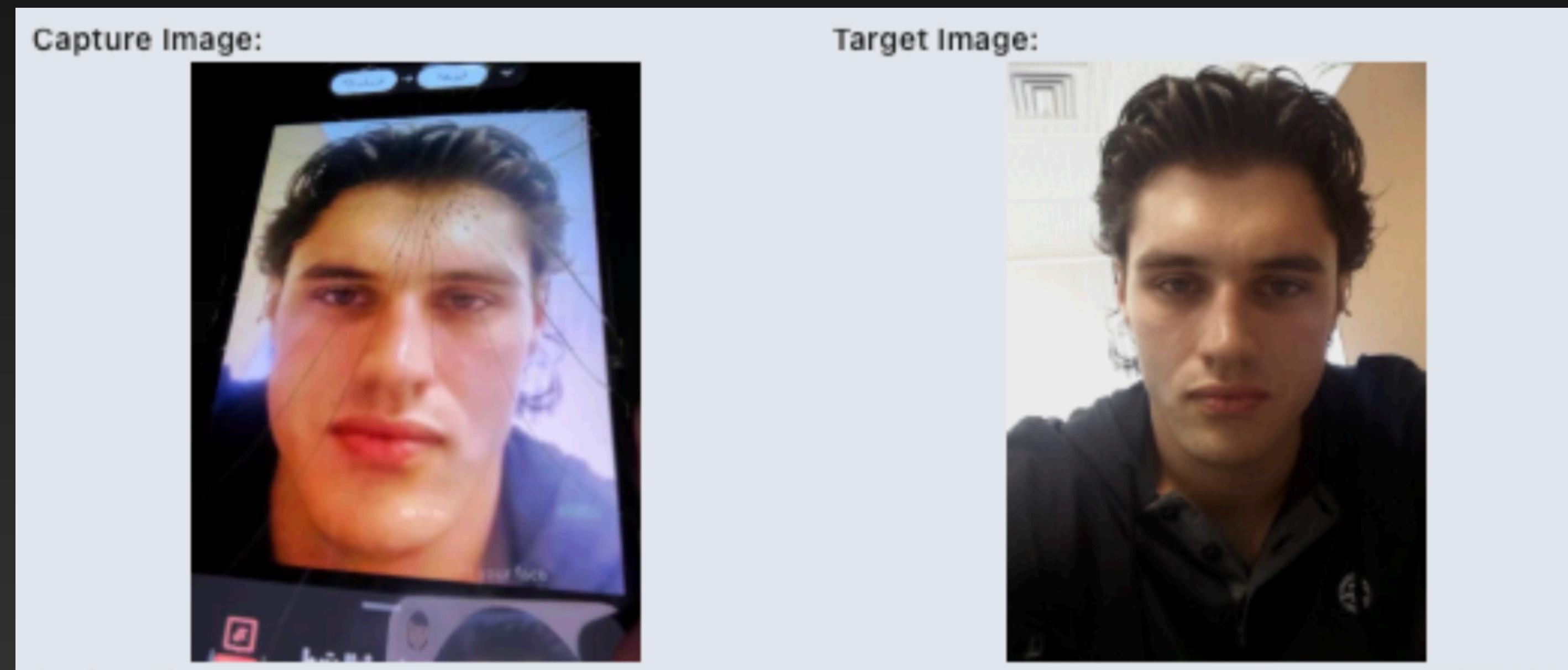
Ethan Fan & Tate Chen

9count

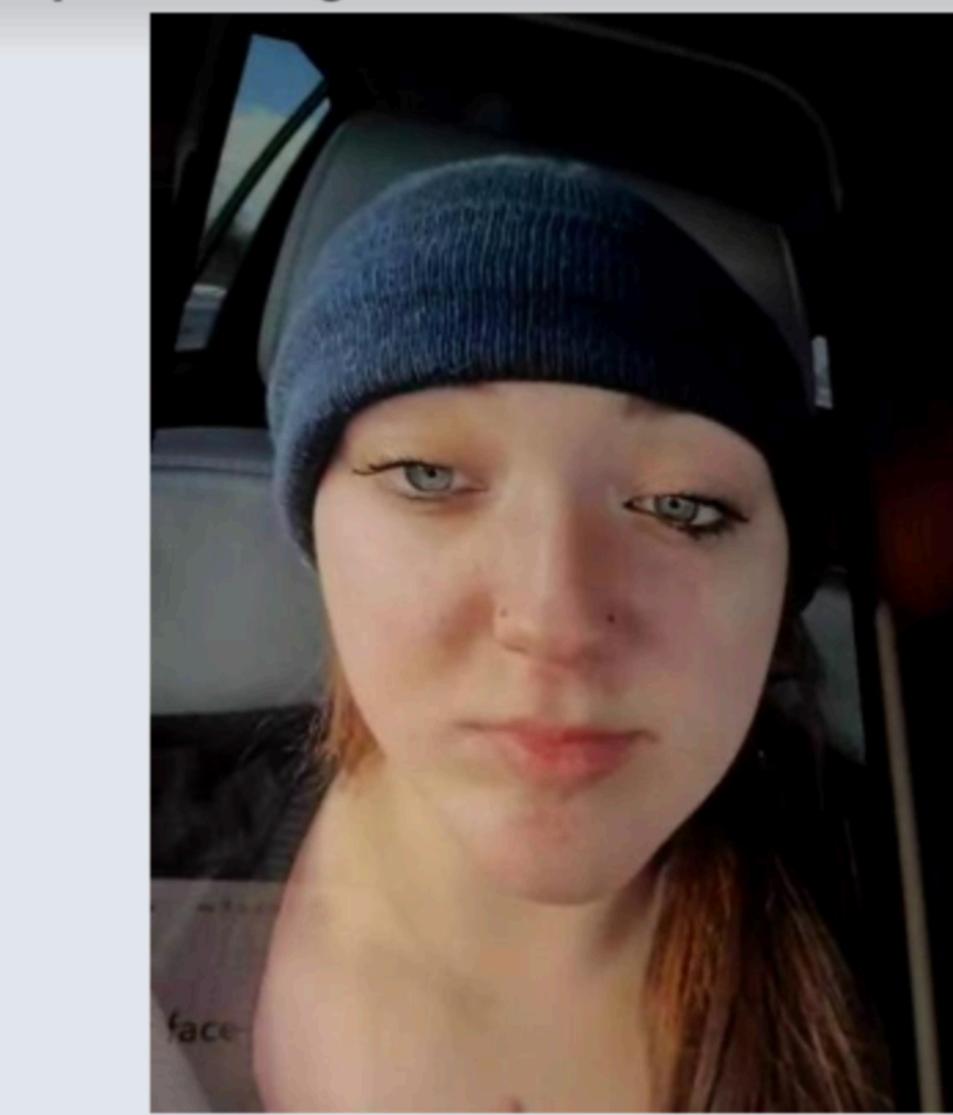


Why we need face liveness?

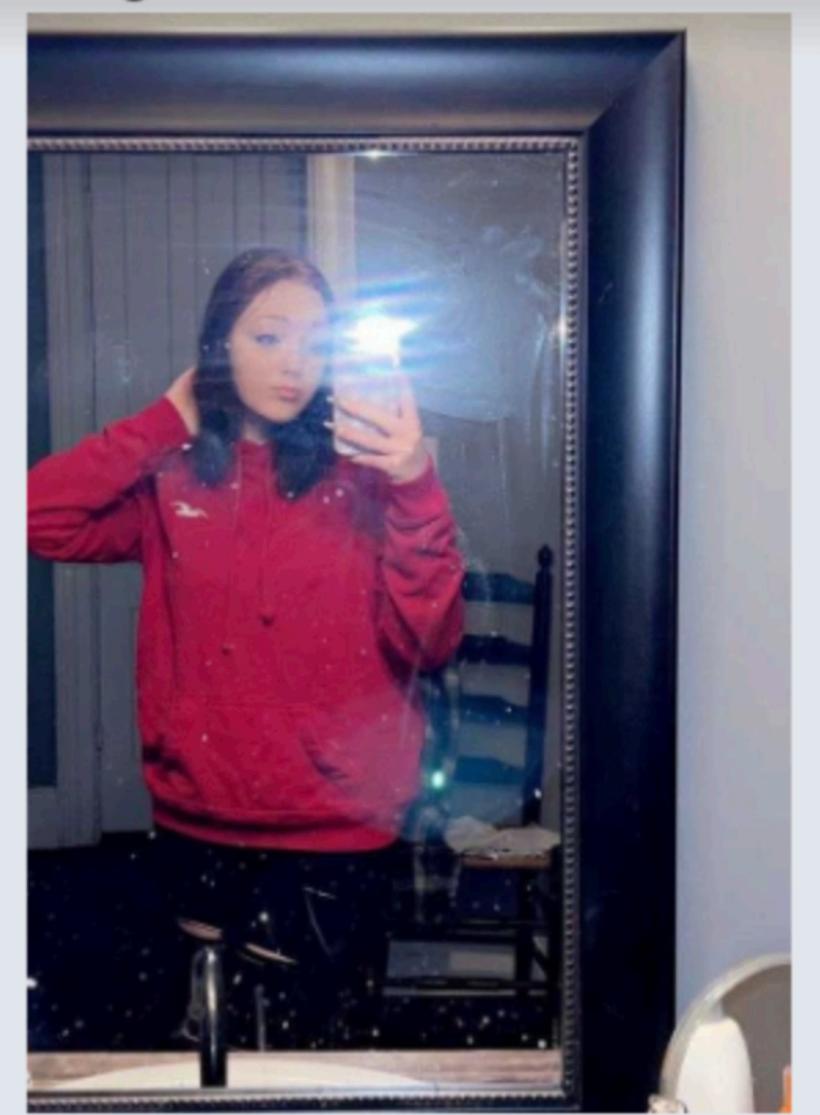
Prevent Spoofing Attacks



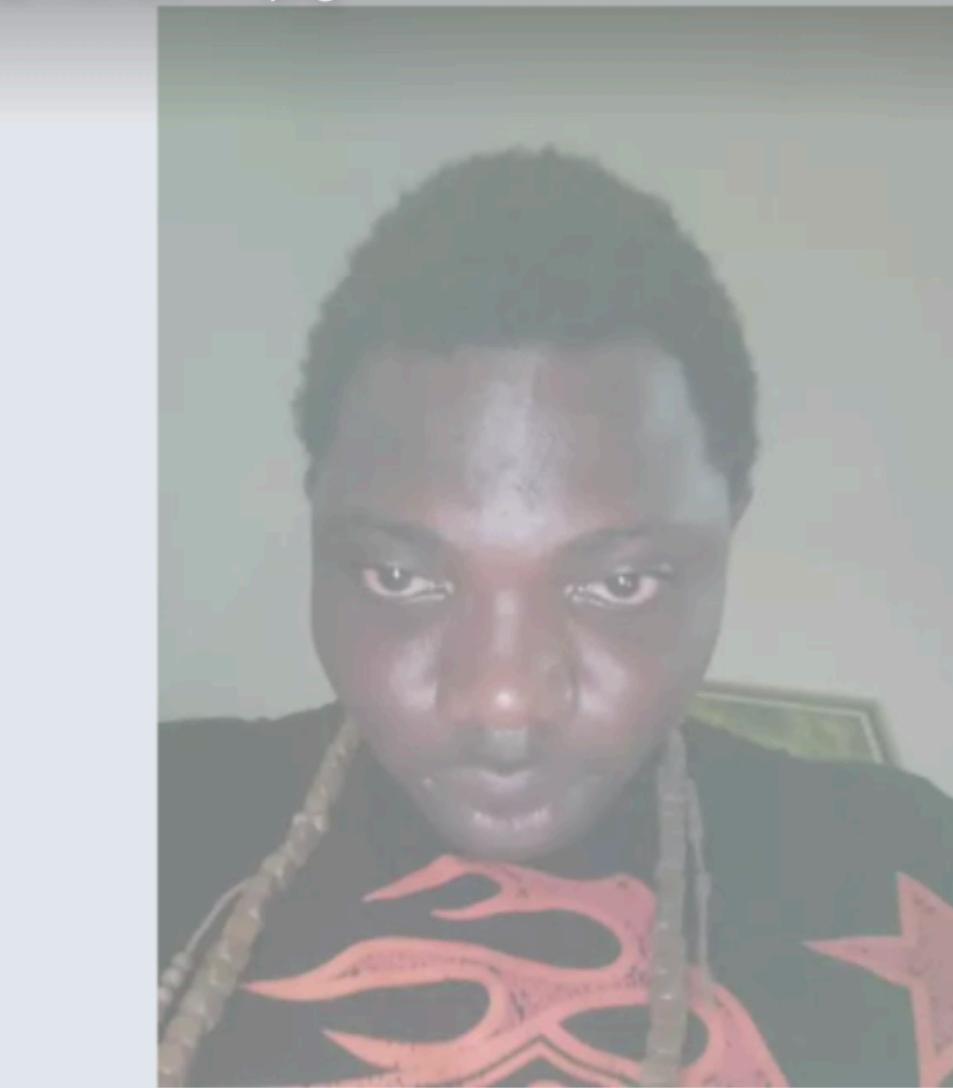
062024_0723 at 11.20.50 PM.png
Capture Image:



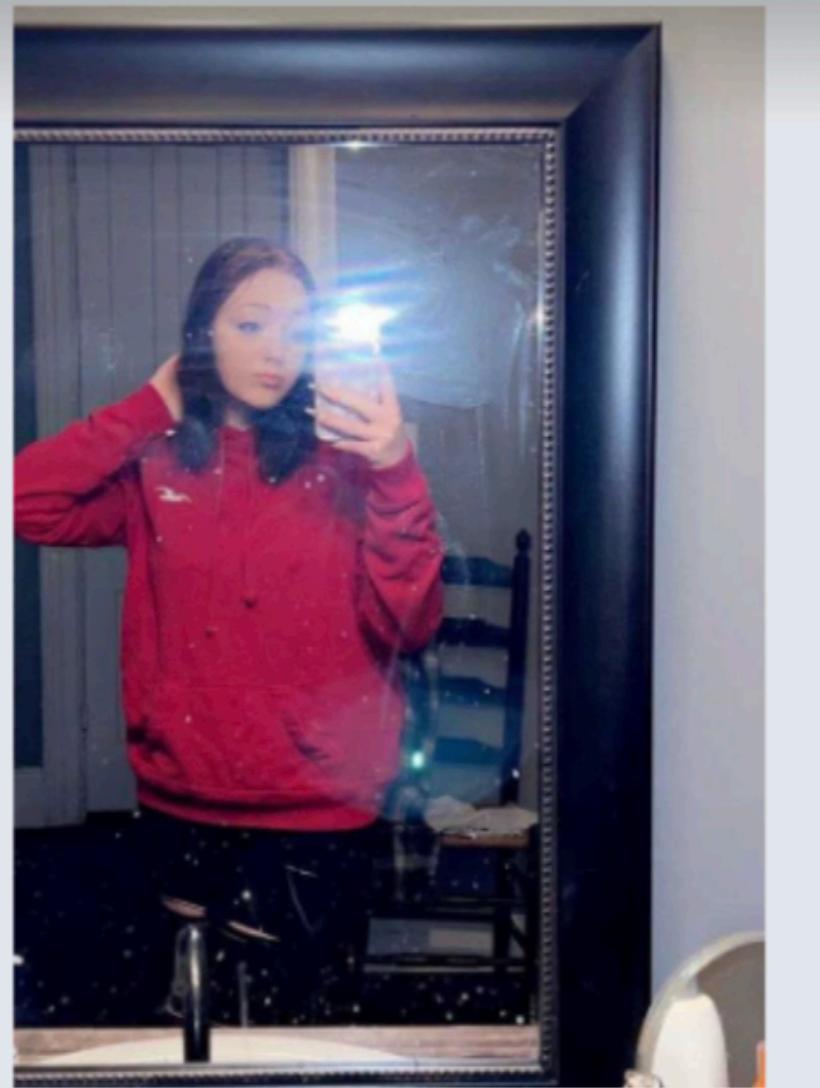
Target Image:



?2 at 7.19.01PM.png
Capture Image:



Target Image:



Why we build depth verification?

Issues we are facing ...

- Apple didn't provide any API for face liveliness**
- Low light environment or under the sun**
- Unable to distinguish 2D facial images**
- It will be a significant expense when scaling up**

AWS Liveness Pricing

Pricing example 1 – Face Liveness check cost for 400,000 checks a month

Let's assume your application performs 400,000 liveness checks in a month and you use the

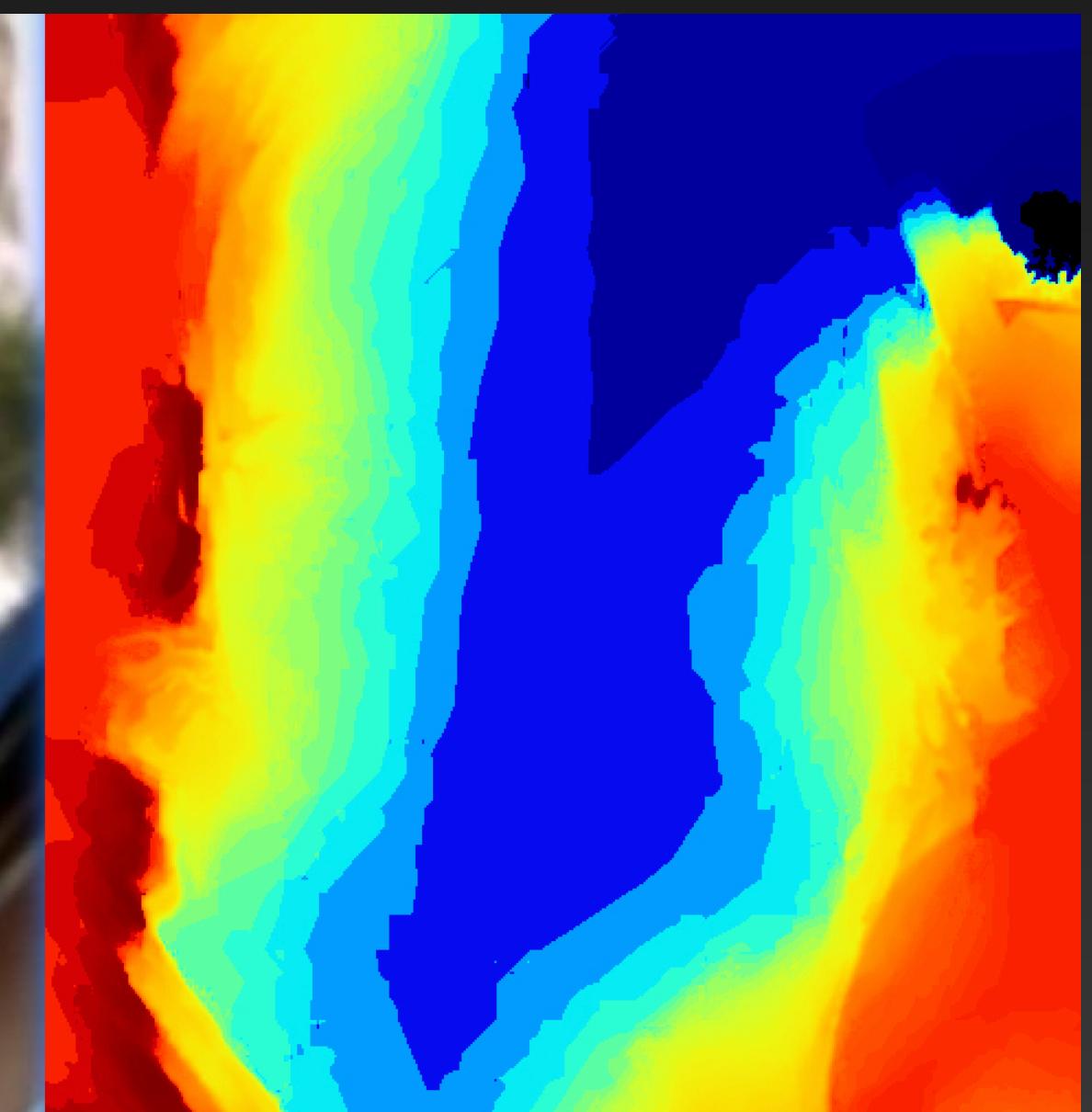
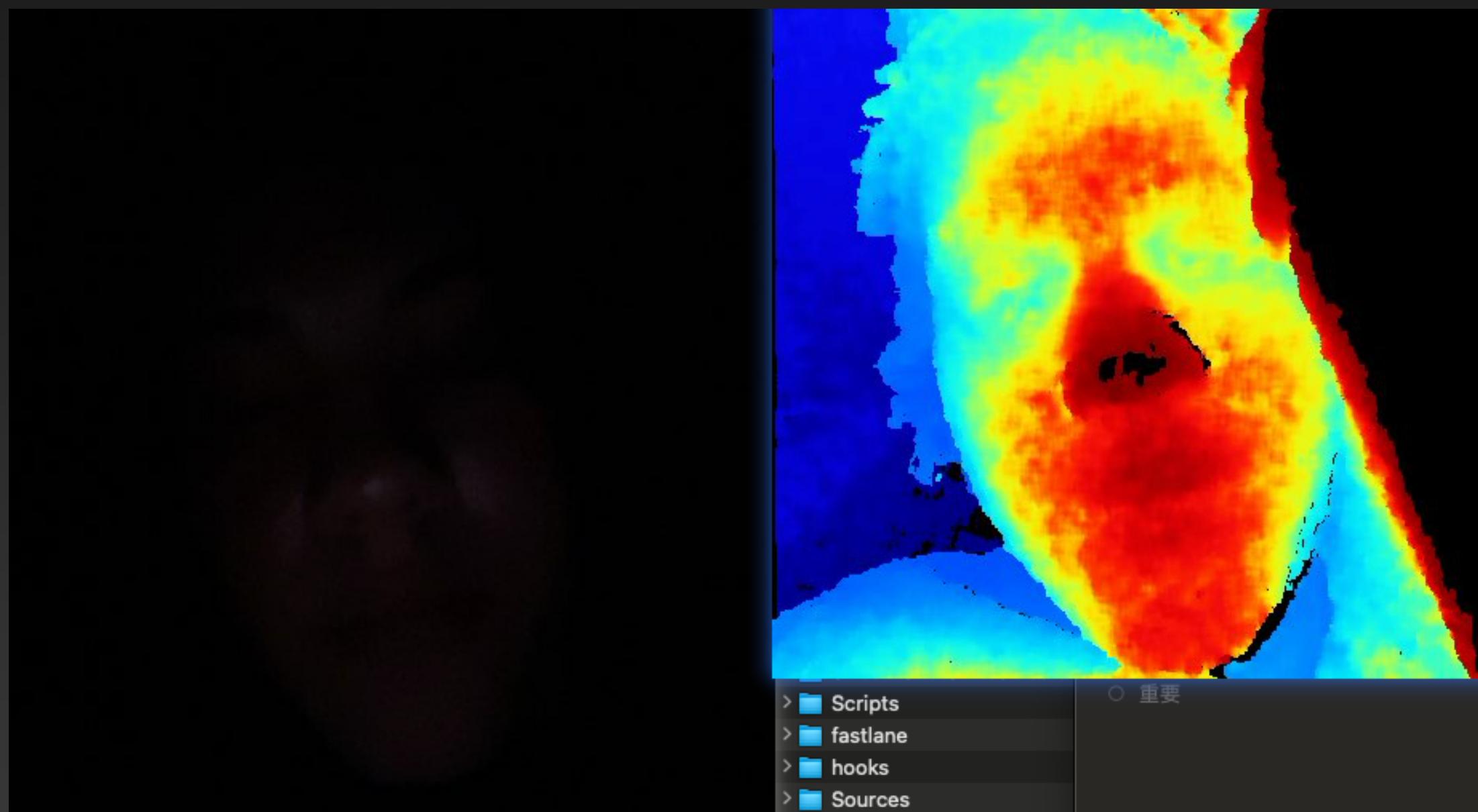
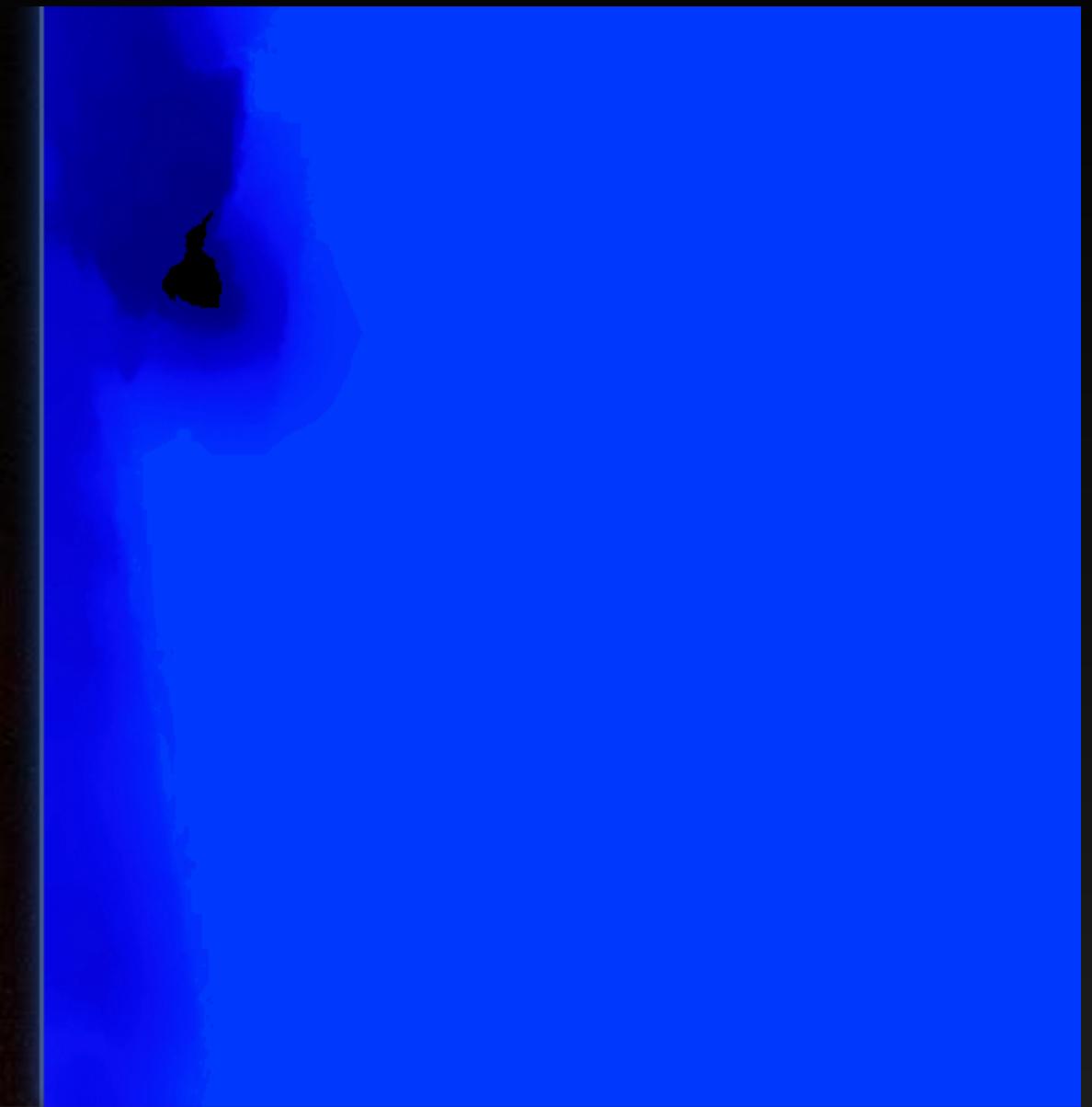
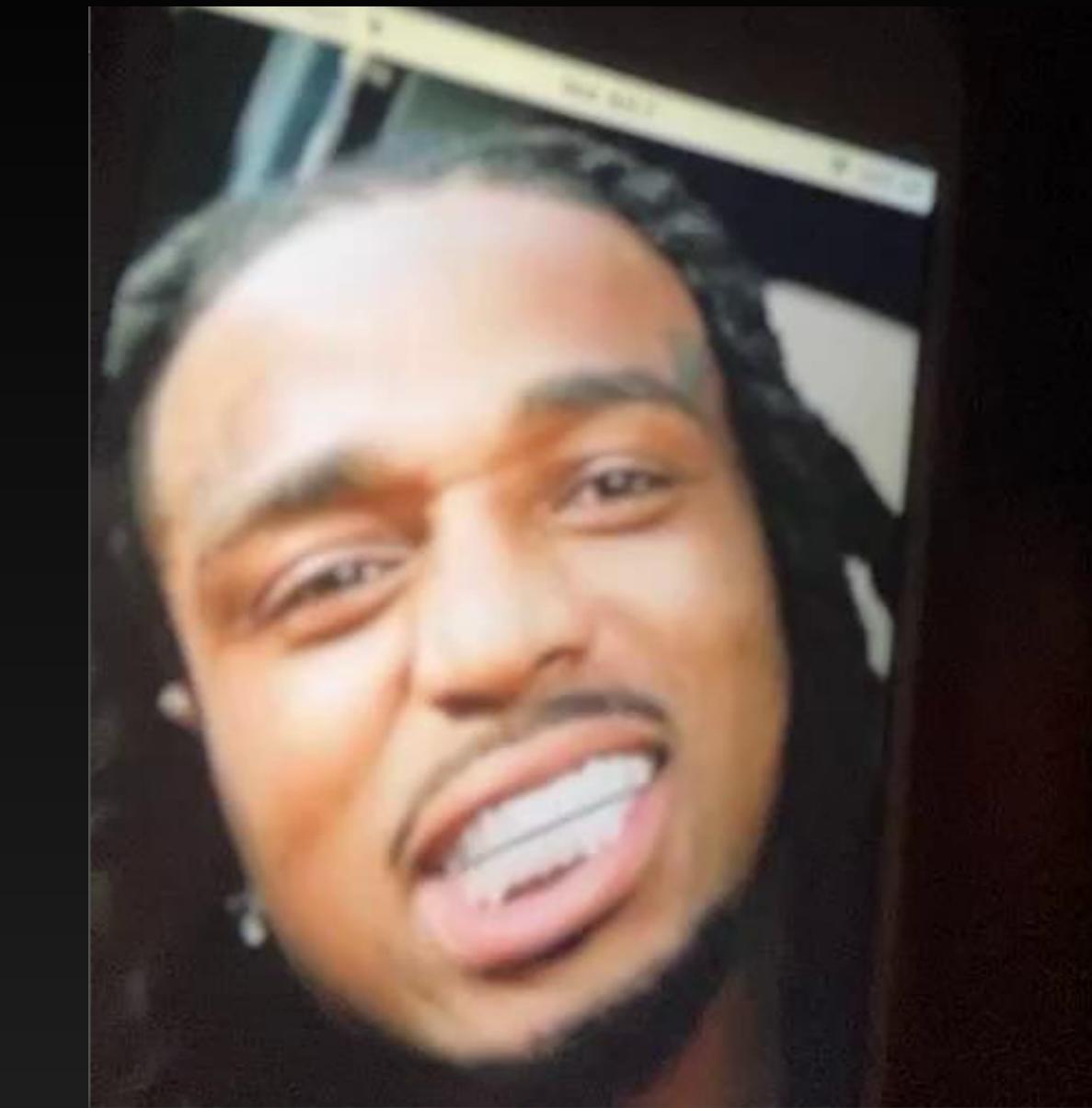
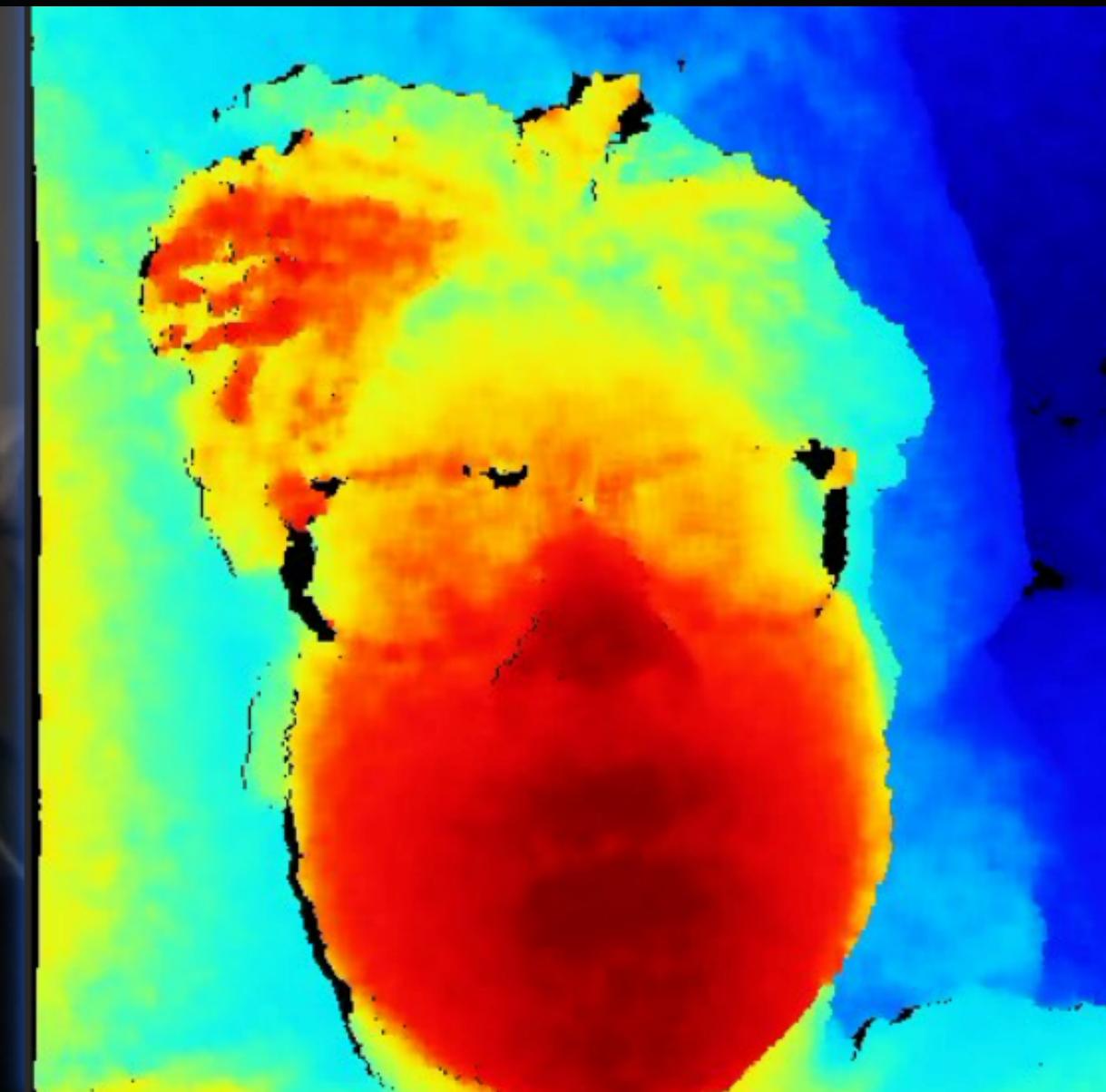
Cost of processing first 500,000 checks in US East (N. Virginia) is \$0.015 per check

Total checks processed = 400,000

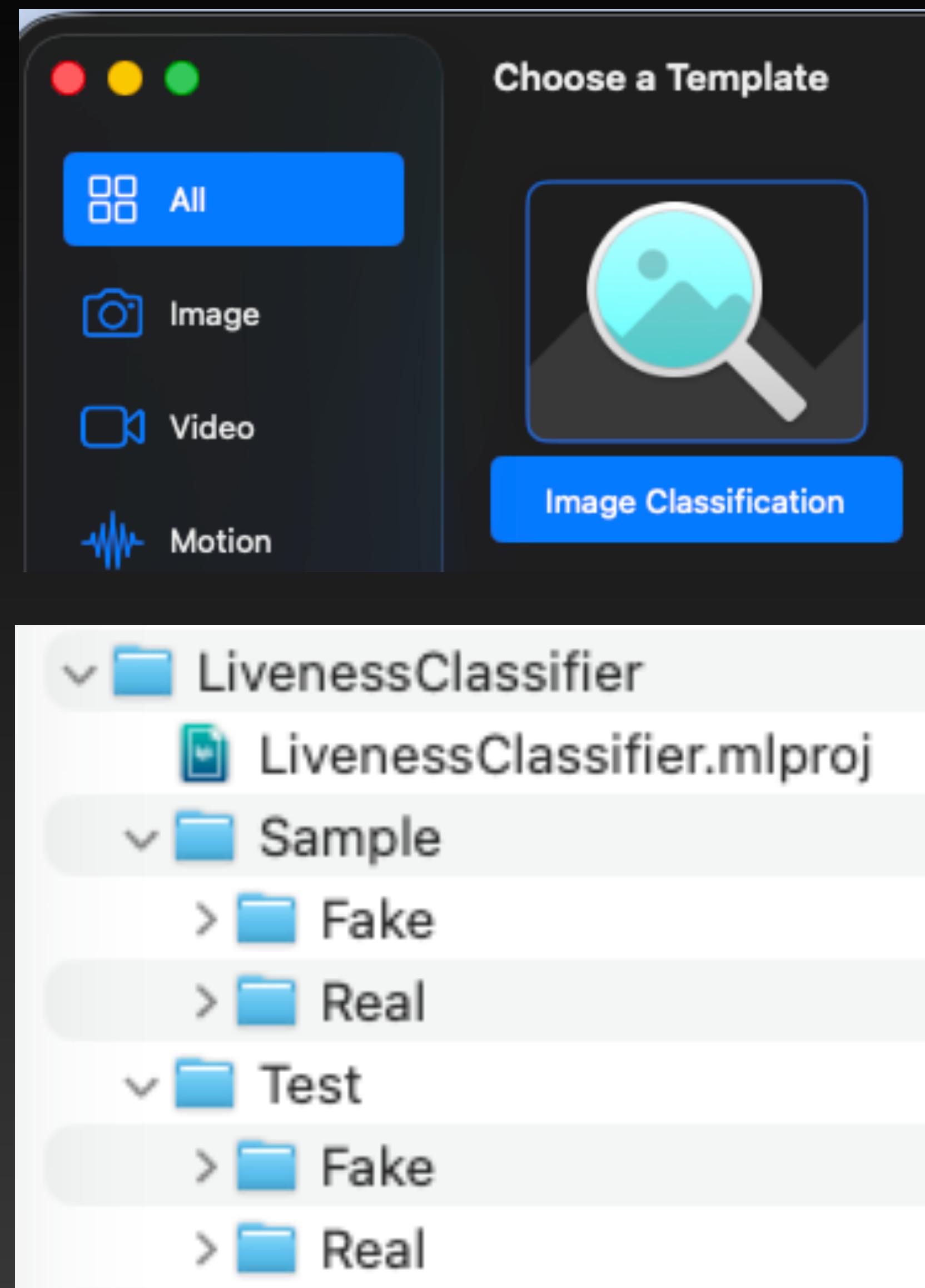
Total cost per month = $\$0.015 * 400,000 = \$6,000$

*<https://aws.amazon.com/rekognition/pricing/>

Train 2D Depth Images with CreateML



Create ML



The right side of the image shows the main configuration screen of the Create ML application.

Data tab (selected):

- Training Data**: 2 Classes, 208 Items. Buttons: View, Sample, Split.
- Validation Data**: Automatically split from training data. Buttons: Customize..., Split.
- Testing Data**: 2 Classes, 51 Items. Buttons: View, Test.

Parameters section:

- Feature Extractor**: Image Feature Print V2. Description: The feature extractor model scales the input image to 360 x 360 and yields a feature embedding size of 768. Model Availability: macOS 14.0+ | iOS 17.0+ | tvOS 17.0+.
- Iterations**: Set to 25.
- Augmentations**:
 - Add Noise
 - Blur
 - Crop
 - Expose
 - Flip
 - Rotate

How it works?

AVFoundation Depth Data from Capture Sessions

```
// Add a video data output
if session.canAddOutput(videoDataOutput) {
    session.addOutput(videoDataOutput)
    videoDataOutput.videoSettings = [kCVPixelBufferPixelFormatTypeKey as String:
        Int(kCVPixelFormatType_32BGRA)]
} else {
    print("Could not add video data output to the session")
    setupResult = .configurationFailed
    session.commitConfiguration()
    return
}

// Add a depth data output
if session.canAddOutput(depthDataOutput) {
    session.addOutput(depthDataOutput)
    depthDataOutput.isFilteringEnabled = false
    if let connection = depthDataOutput.connection(with: .depthData) {
        connection.isEnabled = true
    } else {
        print("No AVCaptureConnection")
    }
} else {
    print("Could not add depth data output to the session")
    setupResult = .configurationFailed
    session.commitConfiguration()
    return
}
```

```
// Use an AVCaptureDataOutputSynchronizer to synchronize the video data and depth data outputs.  
// The first output in the dataOutputs array, in this case the AVCaptureVideoDataOutput, is the  
// "master" output.  
outputSynchronizer = AVCaptureDataOutputSynchronizer(dataOutputs: [videoDataOutput,  
    depthDataOutput])  
outputSynchronizer!.setDelegate(self, queue: dataOutputQueue)  
session.commitConfiguration()
```

```
// MARK: - Video + Depth Frame Processing

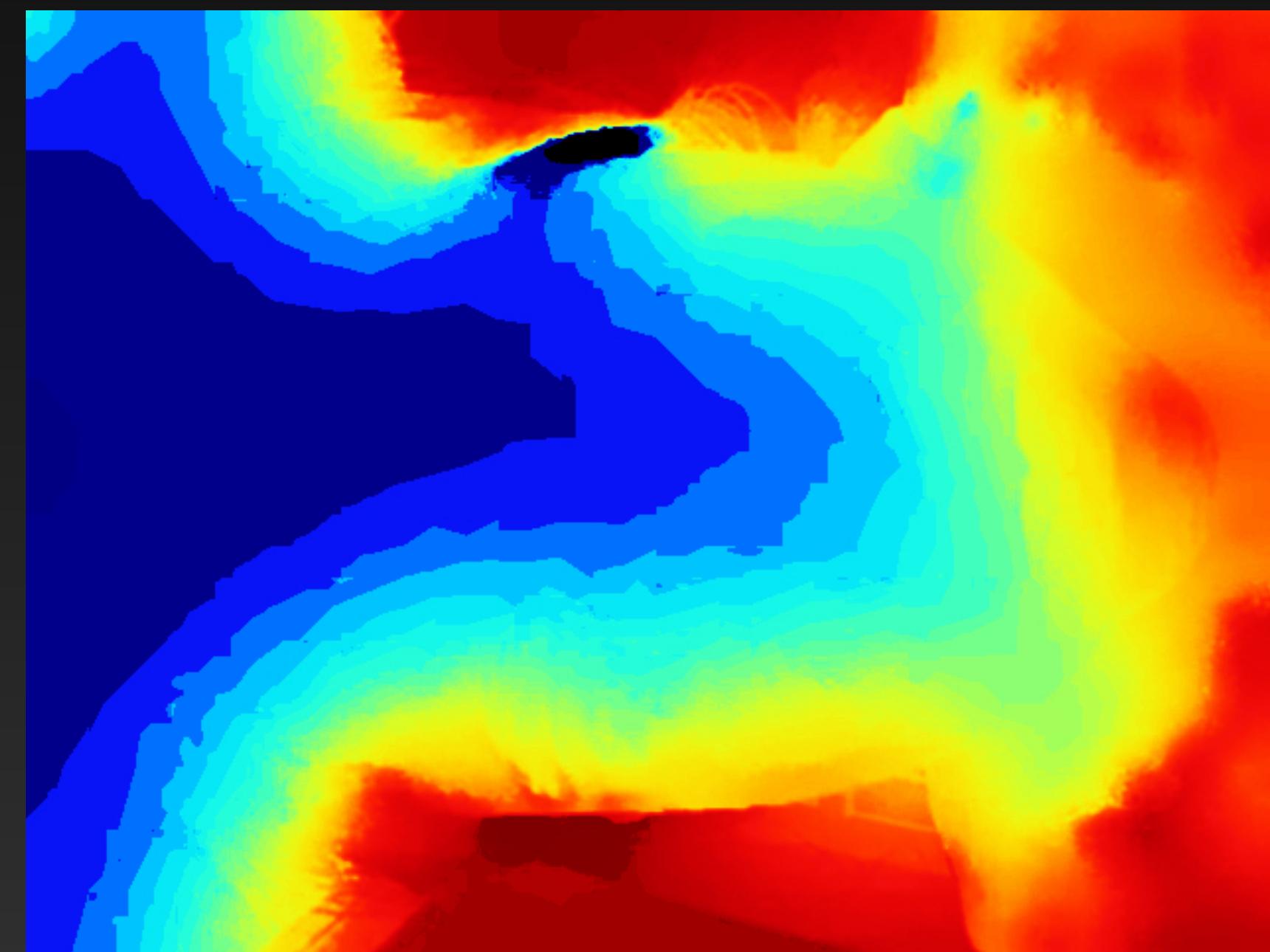
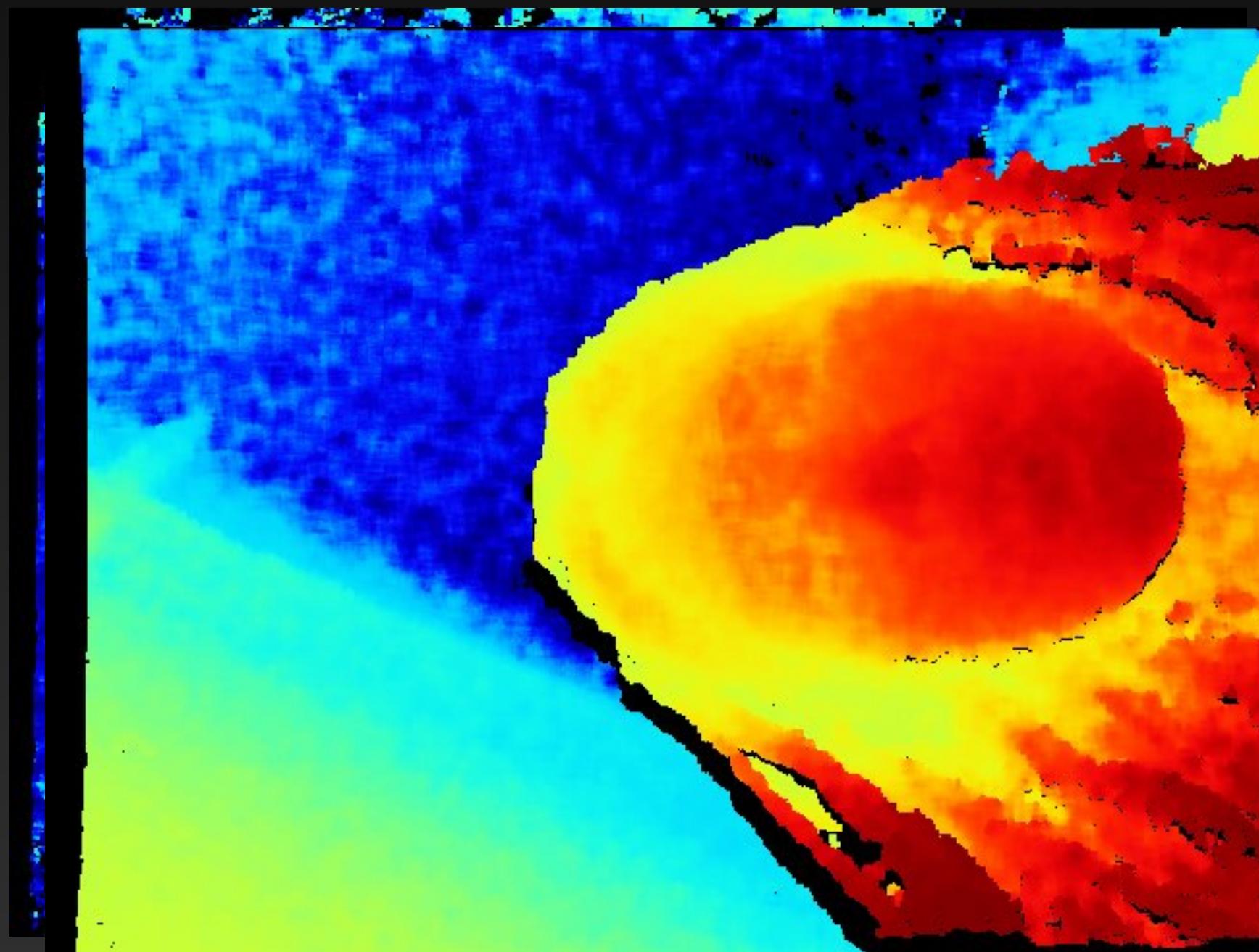
func dataOutputSynchronizer(_ synchronizer: AVCaptureDataOutputSynchronizer,
                           didOutput synchronizedDataCollection: AVCaptureSynchronizedDataCollection)
{
    if !renderingEnabled {
        return
    }

    // Read all outputs
    guard renderingEnabled,
          let syncedDepthData: AVCaptureSynchronizedDepthData =
            synchronizedDataCollection.synchronizedData(for: depthDataOutput) as?
            AVCaptureSynchronizedDepthData,
          let syncedVideoData: AVCaptureSynchronizedSampleBufferData =
            synchronizedDataCollection.synchronizedData(for: videoDataOutput) as?
            AVCaptureSynchronizedSampleBufferData else {
        // only work on synced pairs
        return
    }

    if syncedDepthData.depthDataWasDropped || syncedVideoData.sampleBufferWasDropped {
        return
    }

    let depthData = syncedDepthData.depthData
    let depthPixelBuffer = depthData.depthDataMap
    let sampleBuffer = syncedVideoData.sampleBuffer
```

Convert Depth Data to 2D using Metal Shader

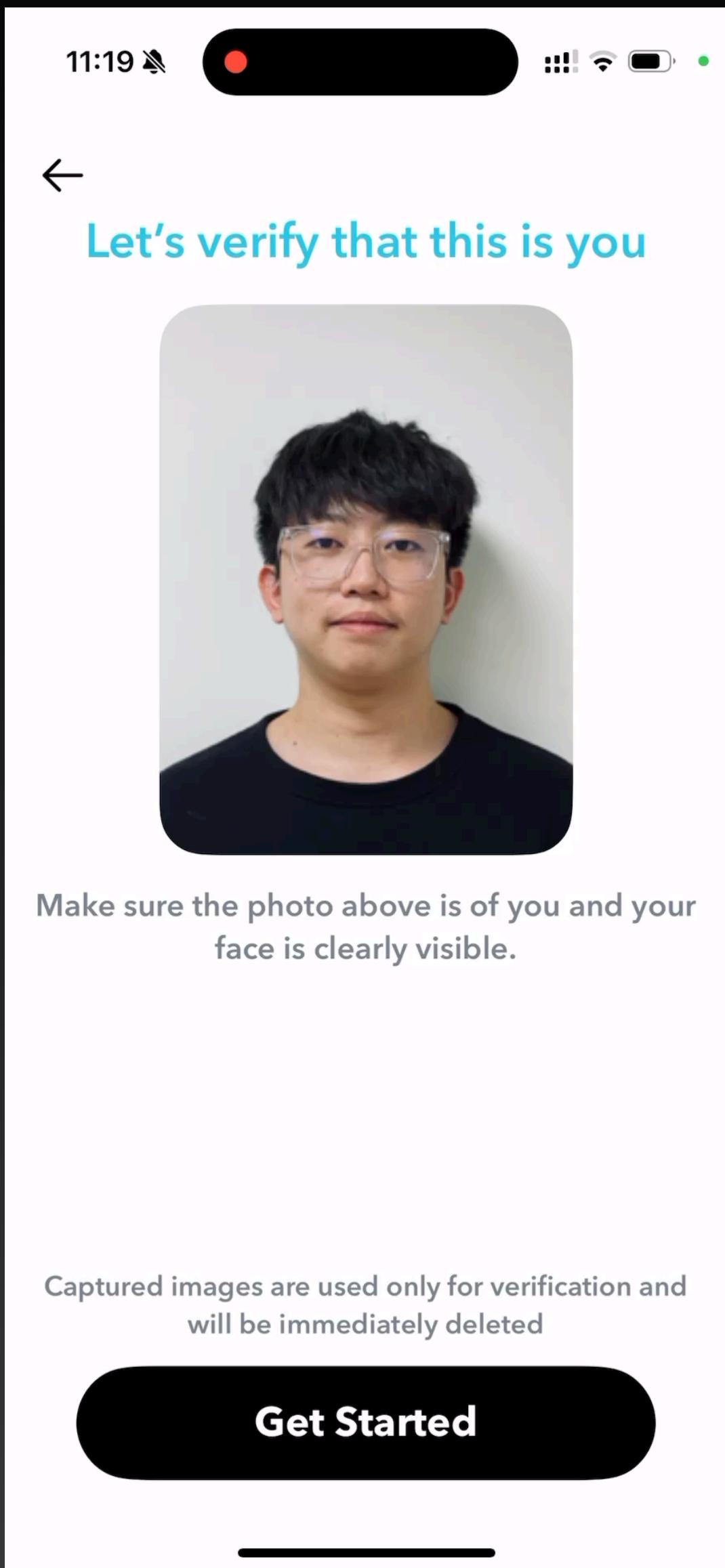


JET color coding to distinguish depth values, ranging from red (close) to blue (far)

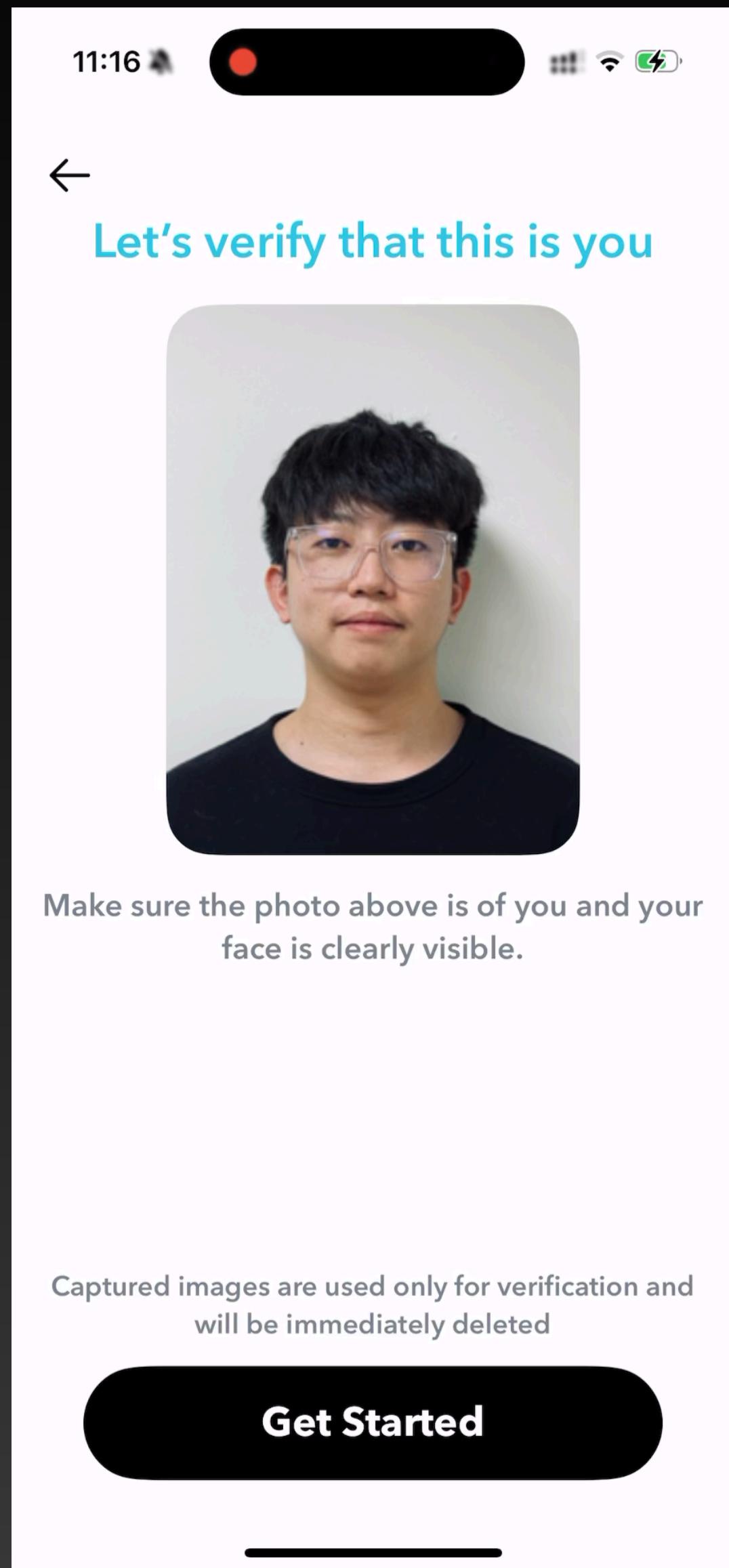
Classifying Images with Vision and Core ML

<https://developer.apple.com/documentation/coreml/classifying-images-with-vision-and-core-ml>

AWS Rekognition



TrueDepth Sensor



Optimization

- Only run inferences when detecting face
- + When the face is within a certain distance

We save around **75%** cost after
implementing depth verification

Twitter/X: coolioxlr