# CovaCare Initial Testing

Brydon Herauf (200454546) & David Kim (200405213)

Faculty of Engineering & Applied Science, University of Regina

ENSE 400

December 6, 2024

## Introduction

The purpose of this testing document is to validate the prototypes that make up a proof of concept for our capstone project. Our project consists of several primary components aimed at creating a comprehensive fall and inactivity detection system. This includes a mobile application, a model configuration API, an SMS alerting feature, camera integration and video processing, fall detection algorithms, and prolonged inactivity detection algorithms.

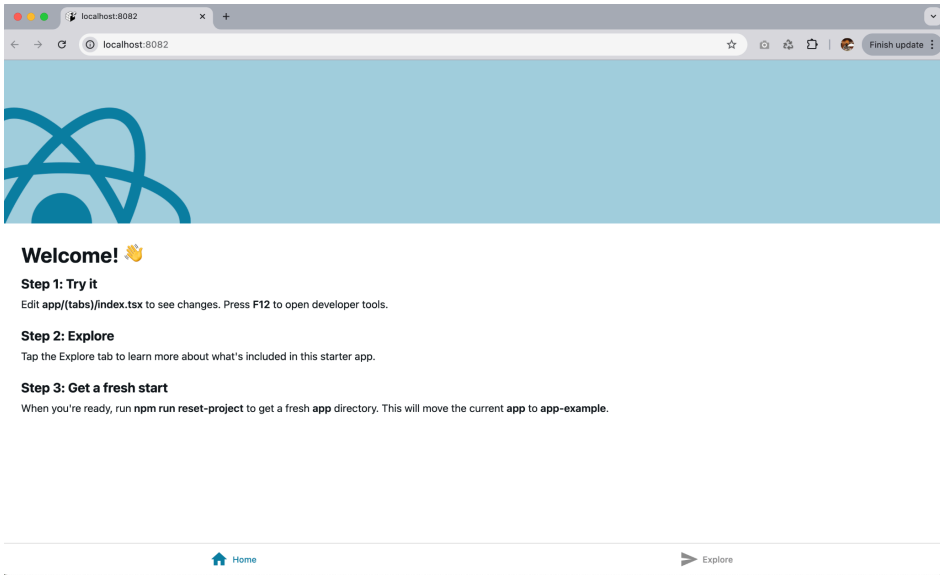
## Test Strategy

At this stage of development, we have a separate slice of every critical part of the system. To validate the functionality of individual components, we have conducted high-level user testing. This testing ensured that each module worked as intended in isolation so we can move forward with integrating them all into a more complete system next.

## The Mobile Application

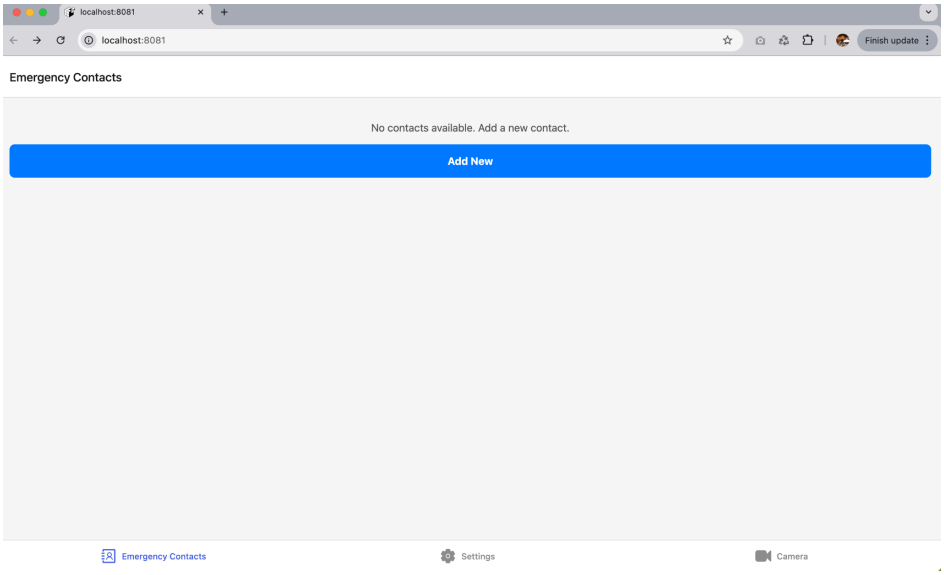
**Test Case 1:** Access a template React Native (Expo) application through a browser.

**Result:** Success. The project was built and successfully loaded on a web browser.



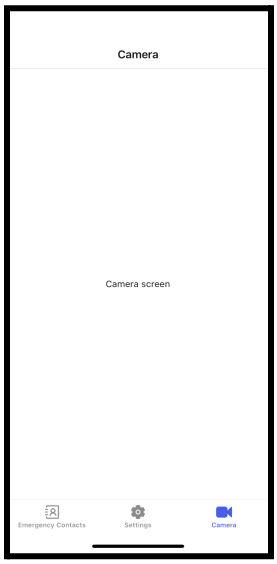
**Test Case 2:** Validate that parts of our Hi-Fi prototype are accurately reflected in the browser.

**Result:** Success. Basic designs were created for each screen according to the high-fidelity prototype, and can be seen on the browser. This design is not final.




**Test Case 3:** View the development application while on a mobile device.

**Result:** Success. The application was viewable by scanning a QR code and loading the app in Expo Go.

**The Model Configuration API**

**Test Case 1:** Access a Flask API using a basic GET route. Send a request from the device that is hosting the API.

**Result:** Success. The API responded with: "This is the GET Endpoint of flask API."


This is the GET Endpoint of flask API.

**Test Case 2:** Access a Flask API using a basic POST route with a simple payload. Send a request from the device that is hosting the API.

**Result:** Success.

A POST request was sent to the / endpoint with a JSON payload:

{"text": "hello"}

The API processed the payload and returned the following JSON response:

{

"cap-text": "HELLO"

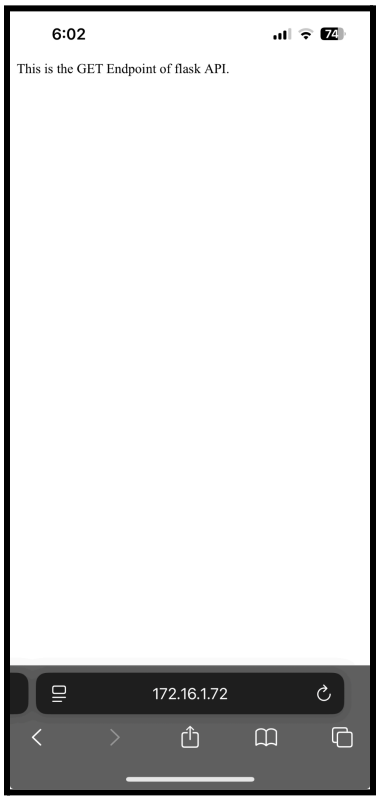
}

```
davidkim@DaviduicBookPro capstone % curl -X POST http://127.0.0.1:4455/ -H "Content-Type: application/json" -
d '{"text": "hello"}'

{
  "cap-text": "HELLO"
}
```

**Test Case 3:** Access the API from a separate device on the same Wifi network using the IP of the hosting device.

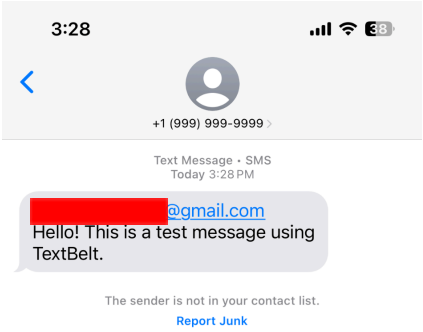
**Result:** Success. The API was accessible from another device on the same network.



**SMS Alerting**

**Test Case:** Send an SMS message from a TextBelt Python script to a developer's mobile phone.

**Result**: The test was successful. After running the Python script, one of our group members received the SMS text message on their phone, confirming the feature's functionality.

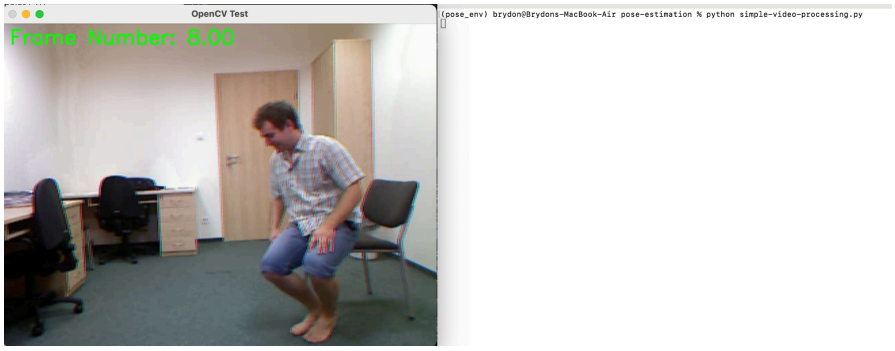



**Camera Integration and Video Processing**

Note - We do not yet have access to a Wifi based camera, so we were unable to fully test integration with an external camera. However, we have scripts that can process videos and live webcam data, as well as a plan for connecting to external devices using an IP.

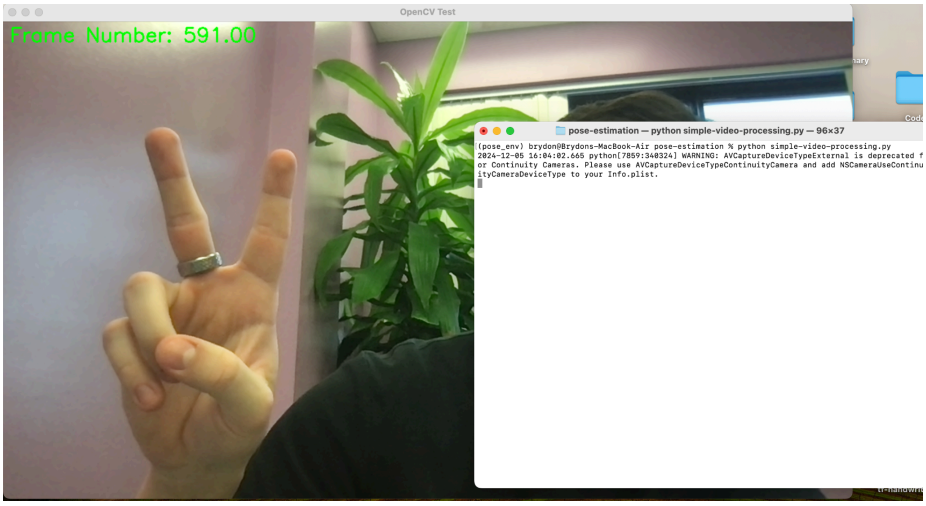
**Test Case 1:** Validate ability to loop through all frames of a sample video

**Result**: Success. Was able to loop through a video frame by frame using OpenCV.




**Test Case 2:** Validate ability to loop through frames from a webcam in real-time

**Result**: Success. Was able to capture live video feed from a webcam and display it in real-time.



**Fall Detection**

**Test Case 1:** Validate ability to detect pose in real-time using various pose estimation models.

**Results:** The following table of results was collected by running various pose estimation models on each frame of a live webcam input, on a MacBook Air.

Processor: 1.1 GHz Quad-Core Intel Core i5

Graphics: Intel Iris Plus Graphics 1536 MB

Memory: 16 GB 3733 MHz LPDDR4X

Operating System: macOS 14.6.1 (23G93)

Execution Method: Simple terminal

| Model | Able to estimate pose? | FPS |
|---|---|---|
| Mediapipe Pose Lite | Yes (Single Pose) | ~ 37 FPS |
| Mediapipe Pose Full | Yes (Single Pose) | ~ 29 FPS |
| MoveNet Lightning Single Pose 32 TFLite | Yes (Single Pose) | ~ 45 FPS |
| MoveNet Lightning Single Pose 16 TFLite | Yes (Single Pose) | ~ 45 FPS |
| MoveNet Lightning Single Pose 8 TFLite | Yes (Single Pose) | ~ 55 FPS |
| MoveNet Thunder Single Pose 32 TFLite | Yes (Single Pose) | ~ 16 FPS |
| MoveNet Thunder Single | Yes (Single Pose) | ~ 16 FPS |

| | | |
|---|---|---|
| Pose 16 TFLite | | |
| MoveNet Thunder Single Pose 8 TFLite | Yes (Single Pose) | ~ 27 FPS |
| MoveNet Lightning Multi Pose 16 TFLite | Yes (Single and Multi Pose) | ~ 25 FPS |
| MoveNet Lightning Single Pose TF2 | Yes (Single Pose) | ~ 17 FPS |
| MoveNet Thunder Single Pose TF2 | Yes (Single Pose) | ~ 9 FPS |
| MoveNet Lightning Multi Pose TF2 | Yes (Single and Multi Pose) | ~ 14 FPS |

Notes:

- There is a heavy Mediapipe model that we did not test yet
- TF2 MoveNet models are heavier, while the TFLite variants are optimized for edge computing
- MoveNet offers simpler and more complex variants of models, specified by 32, 16, and 8.
- All of the TF2 models are 32.
- Lightning is the faster MoveNet model, while Thunder is slightly slower and designed to be more accurate.

**Test Case 2:** Validate that we are able to extract pose data from a video, and store it in a CSV.

**Result:** Success. Pose data was extracted from a video and stored in a CSV file.

pose_results > pose_results.csv

```
0.7697162628173828,0.374052494764328,0.6667017340660095,0.4132630527019501,0.8311837911605835,0.514225959777832,0.679401159286499,0.5514435172080994,0.7477288246154785,0.6280276179
0.7672178745269775,0.3724083304405212 4,0.6632910966873169,0.4097849726676941,0.8308714032173157,0.5083788633346558,0.6735404133796692,0.54517662525177,0.7554324865341187,0.62531888
0.7659324407577515,0.3723973333835602,0.6605724096298218,0.4043271839618683,0.8283933401107788,0.508250355720 52,0.6736353635787964,0.5385125279426575,0.7548739314079285,0.624556541
0.7633916139602661,0.3722271919250488 3,0.6564815044403076,0.4048228561878204 3,0.8282141685485 84,0.5035149455070496,0.6711517572402954,0.5386254787445068,0.7588939666748047,0.617888
0.7603439092636108,0.3727732896804809 6,0.6524065732955933,0.4042922258377075,0.8265830278396606,0.5035861134529114,0.6631089448928833,0.5321993231773376,0.7583555579185486,0.618519
0.7579005956649 78,0.3738617300987247,0.6453536748886108,0.4058863222590295,0.8243671655654907,0.5039961338043213,0.6569420099258423,0.5339690446853638,0.7585969567298889,0.619257
0.7545396089553833,0.3748698830604553,0.6400724053382874,0.4127123951911926,0.8224840164184 57,0.5040122866630554,0.6532329320907593,0.5410525798797607,0.7580704092979431,0.61953806
0.7498693466186523,0.3754444122314453,0.6366143226623535,0.4155451655387878 4,0.8193996548652649,0.5008470416069031,0.6460096836090088,0.5411713719367981,0.7577171325683594,0.619265
0.7456504702568054,0.3775352835652124,0.6310265660828595,0.4209323227405548,0.8147360086441 04,0.5014148354530334,0.6464980244636536,0.5450573563575745,0.7545968294143677,0.62160259
0.7396044135093689,0.3791334033012 39,0.6286823153495789,0.4231581389904022,0.8097667694091797,0.5007231831550598,0.6375235319137573,0.5432373285293579,0.7496429681777954,0.62213319
0.7343149185180664,0.3807985484600067,0.6221488714218 14,0.4270703196525574,0.8072170019149 78,0.5012469887733046,0.6357611417770386,0.5460078120231628,0.7446231842041016,0.6282831430
0.7295669913291931,0.3842456638813019,0.6176213026046753,0.4334003031253814 7,0.8040251731872559,0.5013757944107056,0.6306356191635132,0.5481654405593872,0.7397914528846741,0.631914
0.7232068777084351,0.3875409066677093 5,0.6106087565422058,0.4360488355159759 5,0.7983438372612,0.5022972226142883,0.6288349628448486,0.5514857172966003,0.7338179349899292,0.63767910
0.7173899412155151,0.3918760418891907,0.6032000780105591,0.4418720910124207,0.7949019670484645,0.5045598745346069,0.6221152544021606,0.5603554248809814,0.7285814881324768,0.6434250
0.7064965963363647,0.3955932259559631 3,0.5974536538124084,0.4471893012523651,0.7878219485282898,0.5082287788391113,0.6208750009536743,0.5657109618186951,0.7217167019844055,0.645737
0.6994251012802124,0.4009487032890 32,0.5914601087570 19,0.4571805894374847 4,0.7818140983581543,0.5142431855201721,0.6103779077529907,0.5759122371673584,0.7127759456634521,0.64556336
0.6882337331771851,0.4057243764400482,0.5818498134613037,0.4651188850402832,0.7708995342254639,0.5188452601 4328,0.5987364053726196,0.5872200131416321,0.6941495537757874,0.644511580
0.6763380765914917,0.4127282202243805,0.5742715001106262,0.4735880196094513,0.7524884939193726,0.5335015654563904,0.5901919603347778,0.5968840122222 9,0.6725252866744995,0.656971275
0.6625479459762573,0.4196654260158539,0.5663634538650513,0.4851642549037933 3,0.7392846345901489,0.5449478626251221,0.5856209993362427,0.6036926507949829,0.6499424576759338,0.674444
0.6458936333656311,0.4280068278312683,0.5529961585998535,0.4892361462116245,0.7117991447448 73,0.5618678331375122,0.5580877065658569,0.6147094964981079,0.6232613325119019,0.6814053
0.6313031315803528,0.4403387010097537,0.5396779775619507,0.4989928901195526,0.6848000288009644,0.580266535282 35,0.5468779802322388,0.6278339624404907,0.5954837203025818,0.6930283
0.6176444292068481,0.4495325088500976 6,0.5268930196762085,0.5160688757896423,0.6646688580513,0.5944613814353943,0.5340903997421265,0.6459815502166748,0.5711687803268433,0.706851243
0.5993274450302124,0.4669146239757538,0.5227397084236145,0.5391981005668 64,0.6361349225044 25,0.6151641607284546,0.5302546024322 51,0.6699613332748413,0.5461157560348511,0.7281605005
0.5850576758384705,0.4873674213886261,0.5163710713386536,0.5586596727371216,0.5942603349685669,0.6471625566482544,0.5213050246238708,0.6858961582183838,0.5061412453651428,0.7731219
0.5649903416633606,0.5031500458717346,0.5025242567062378,0.5745033025741577,0.5570750832557678,0.6614214181900024,0.5081886053085327,0.6918089389801025,0.477042019367218,0.77713567
0.5466448664665222,0.5302468538284302,0.4893213212490082,0.5731709003448486,0.5152271389961243,0.6853516101837158,0.4850398600101471,0.6988976597785595,0.4429664611816406,0.79876303
0.5291163325309753,0.5661067366600037,0.4829062521457672,0.5887120962142944,0.4830378236427307,0.7269796133041382,0.4759298264980316,0.7258933782577515,0.39369797706604004,0.83659
0.5090175867080688,0.5948804020881653,0.4767267107963562,0.6064474582672119,0.4702007174491882 3,0.7435182929039001,0.4725295603275299,0.7464736700057983,0.3887942135334015,0.873613
0.4833145141601562 5,0.6193749308586121,0.4630377888679504 4,0.6352757811546326,0.4533337652683258,0.7515379786491394,0.4596714973449707,0.7676368951797485,0.3892164826393127 4,0.8881
0.4623570442199707,0.6452933549880981,0.4441916942596435 5,0.6592488884925842,0.4381094574928283 7,0.7778176069259644,0.4547958970069885,0.7689481973648071,0.3861775994300842 3,0.9090
```

**Test Case 3:** Validate the fall detection LSTM training process by training a sample model on the pose estimated sequence from Test Case 2.

**Result:** Success. Was able to confirm that the model and training process were configured properly by training a model on a single sample video.

Note - Obviously training a model on one video is comically insufficient, but this is just to verify that the groundwork is properly laid for the training process.

| Layer (type) | Output Shape | Param # |
|---|---|---|
| input_layer (InputLayer) | (None, 30, 24) | 0 |
| fall_detection_lstm (FallDetectionLSTM) | (None, 1) | 78,465 |

**Total params:** 78,465 (306.50 KB)
**Trainable params:** 78,465 (306.50 KB)
**Non-trainable params:** 0 (0.00 B)

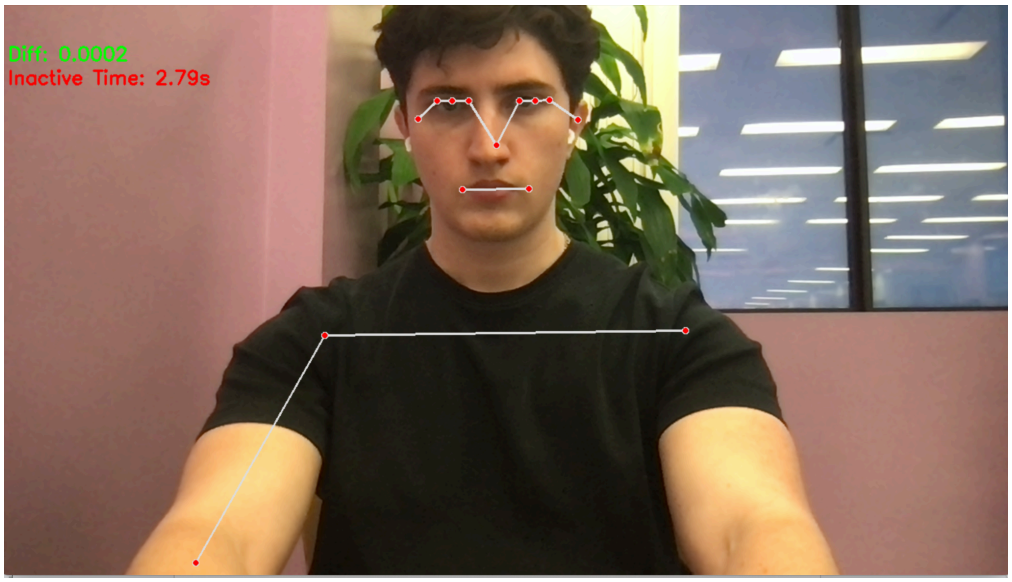
```
Epoch 1/10
1/1 ━━━━━━━━━━━━━━━━━━━━ 1s 1s/step - accuracy: 1.0000 - loss: 0.6367
Epoch 2/10
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 31ms/step - accuracy: 1.0000 - loss: 0.4238
Epoch 3/10
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 32ms/step - accuracy: 1.0000 - loss: 0.2674
Epoch 4/10
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 31ms/step - accuracy: 1.0000 - loss: 0.1601
Epoch 5/10
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 30ms/step - accuracy: 1.0000 - loss: 0.0925
Epoch 6/10
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 31ms/step - accuracy: 1.0000 - loss: 0.0529
Epoch 7/10
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 32ms/step - accuracy: 1.0000 - loss: 0.0307
Epoch 8/10
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 32ms/step - accuracy: 1.0000 - loss: 0.0184
Epoch 9/10
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 36ms/step - accuracy: 1.0000 - loss: 0.0115
Epoch 10/10
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 33ms/step - accuracy: 1.0000 - loss: 0.0075
(pose_env) brydon@Brydons-MacBook-Air pose-estimation %
```

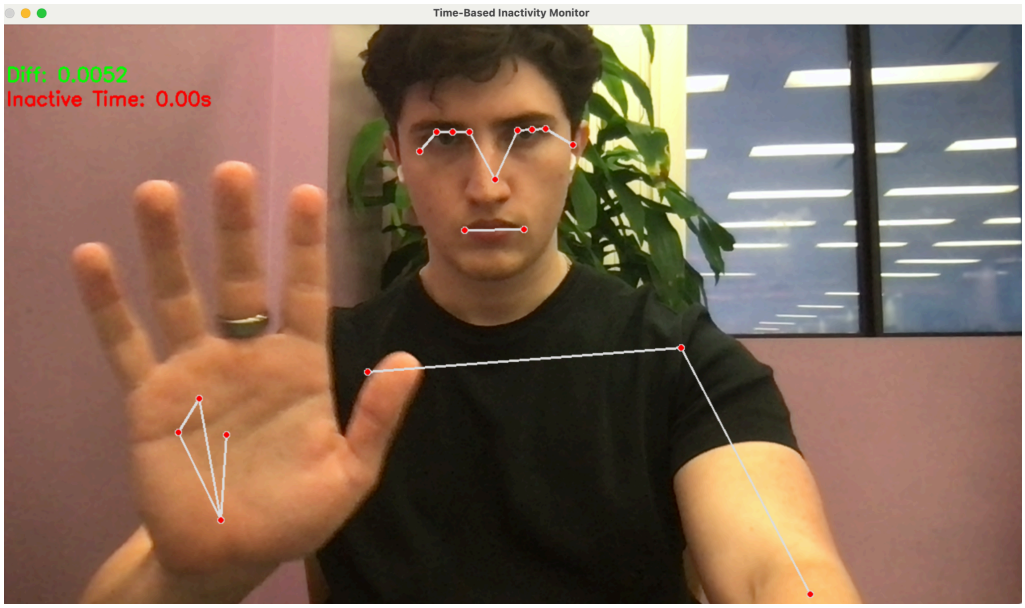
**Prolonged Inactivity Detection**

Prolonged inactivity detection depends on pose estimation, which was already tested in the previous section.

**Test Case 1:** Validate that our pose based inactivity detector will count the amount of time that has passed since last movement from webcam input.

**Results:** Success. The counter increases as I stay still, and resets when I move.




Time-Based Inactivity Monitor
Diff: 0.0052
Inactive Time: 0.00s