Part Two: Offload the DNN-based inference

Implementation
My implementation has a local server rust file and a remote server rust file – the local server captures the image from the video stream in the App struct, and then sends that image over a TCP stream to the remote server using the ServerFacing struct. The remote server reads the image from the stream, applies the model to the image, and then sends back the keypoints as the result. The local server will then render the keypoints as well as the image on the screen.

Difficulties
I had a lot of difficulties setting up the stream – I was incorrectly serializing and deserializing my image and I didn't understand how to properly read the length of the buffer. I also had issues connecting my servers using TCP – I was using UTM and had difficulty setting up the port forwarding to allow TCP on the correct port.

AI
AI wrote most of my baseline code – it wrote all of my App and Server struct as well as the handle_client method. However, most of the details (such as what address to use, how to deserialize, how to flatten) were incorrect, so I had to make adjustments to that.

In your report, please describe any noteworthy efforts by you toward the above objectives.
The structs are quite easy to understand and use, so they are effective for use by app developers. The second interface has only one method and it only reads and analyzes the image, so it achieves high performance. Since this is all happening quickly, the app is able to correctly mark the results on the image in a way indistinguishable from part one.