Choose the Right Hardware

Proposal Template

Scenario 1: Manufacturing

Client Requirements and Potential Hardware Solution

Looking through the scenario and finding the client requirements, we suggest a potential hardware type by explaining how this hardware would satisfy each of the requirements.

Which hardware might be most appropriate for this scenario? (CPU / IGPU / VPU / FPGA)
HETERO:FPGA+CPU

Requirement Observed (Include at least two.)	**How does the chosen hardware meet this requirement?**
the floor running **24 hours a day** so that packaging continues nonstop ➔ **Robust**	FPGAs are designed to have *100% on-time performance*, meaning they can be continuously running *24 hours a day, 7 days a week, 365 days a year.* They are also able to function over a wide range of temperatures, from 0° C to 60° C. This means that FPGAs can be deployed in harsh environments like *factory floors and still perform optimally.*
Each camera records video at **30-35 FPS** Mr. Vishwas would like the image processing task to **be completed five times per second.** ➔ **High performance, low latency**	Once programmed with a suitable bitstream, FPGAs can execute neural networks with *high performance and very little latency*. The high performance comes from the ability to *run many sections of the FPGA in parallel*. FPGAs also flow the data from one layer to the next, while *keeping the data from one output to the next input layer on the same FPGA device.* When running a neural network, *we run the whole thing on the FPGA so the FPGAs don't go off-chip for the memory*. This is faster than sending the output back to the CPU over the PCIe bus.

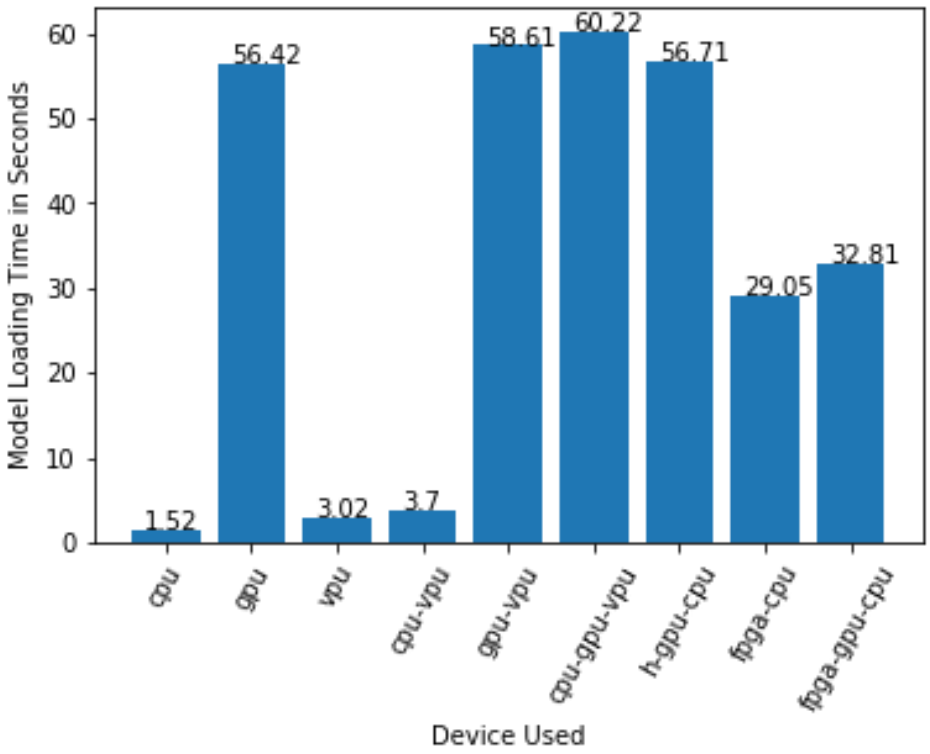
Once this productivity problem has been addressed,Mr. Vishwas would like to be able to **repurpose the system to address a second issue** ➔ **Flexibility**	FPGAs are *field-programmable; they can be reprogrammed to adapt to new, evolving, and custom networks*.
To be able to detect chip flaws without slowing down the packaging process, the system would need to be able to **run inference on the video stream very quickly.** ➔ **Large Networks**	One feature of FPGAs that makes them especially useful in deep learning is that they can support large networks, with a capacity to handle networks that have *more than 2 million parameters*.
Additionally, because **there are multiple chip designs—and new designs are created regularly**—the system would also **need to be flexible so that it can be reprogrammed** and **optimized to quickly** detect flaws in **different** chip designs. ➔ **Flexibility**	Various precision options (FP16, 11 and 9 bit) are supported—allowing developers a *balance between speed and accuracy*. The bitstreams being used can be *updated without changing the hardware*. This allows you to *improve the performance* of your system *without replacing the FPGA*.
While Naomi Semiconductors has **plenty of revenue** to **install a quality system,** this is still a significant investment ➔ **Budget for hardware: Any**	FPGAs costs a lot (approx.~$1,600.00) + CPU
they would ideally like it to last for at least **5-10 years.** ➔ **Long Lifespan**	FPGAs have a *long lifespan*. For example, FPGAs that use devices from Intel's Internet of Things Group have a *guaranteed availability of 10 years*, from start of production.

Queue Monitoring Requirements

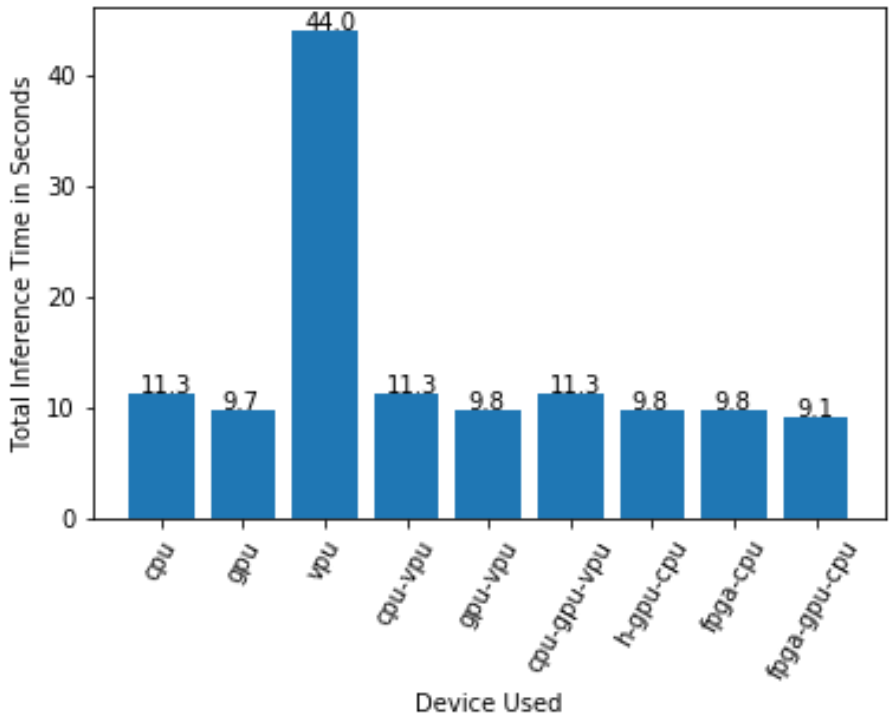
Maximum number of people in the queue	*5*
Model precision chosen (FP32, FP16, or Int8)	*FP16*

Test Results

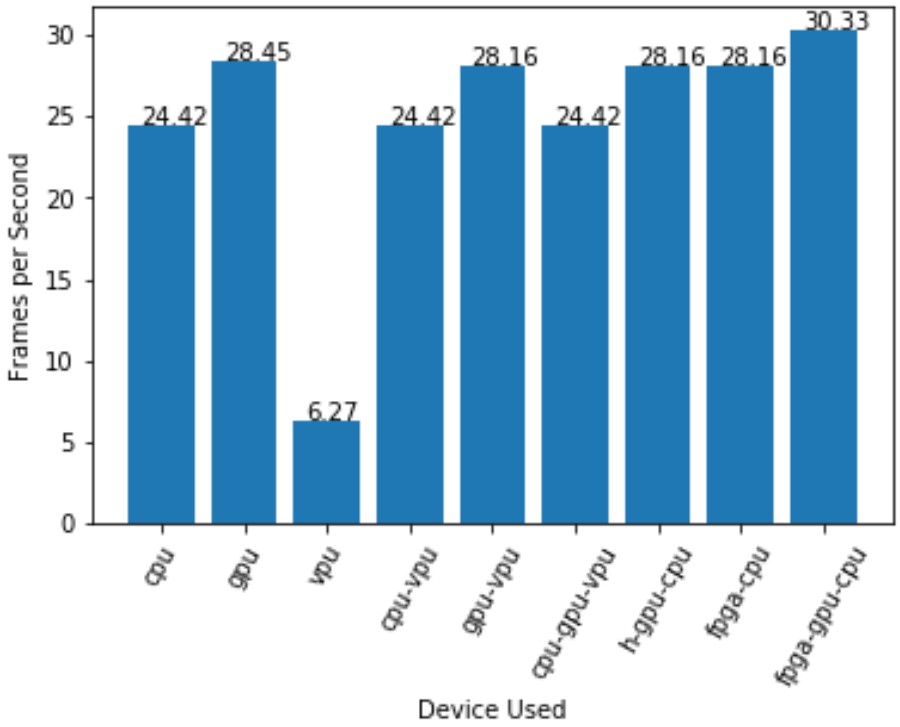
After the testing of the application on all four hardware types (CPU, IGPU, VPU, and FPGA), with the matplotlib output we can observe and compare the results below. There are three graphs (for model load time, inference time, and FPS).



Model Load Time



Inference Time



FPS

Final Hardware Recommendation

Synthesizing the points from above and we claim that the chosen hardware is the best choice for this scenario. The client's requirements, the test results, and how these relate to one another are summarized below.

Write-up: Final Hardware Recommendation
The budget has no limitations, so we can get the combination with the best performance. Considering the client's requirements and the small gap of ~3sec model load time of HETERO: FPGA, CPU vs HETERO: FPGA, IGPU, CPU and the best performance of the latter one in FPS and inference time, we will suggest the HETERO: FPGA, IGPU, CPU solution.

Scenario 2: Retail

Client Requirements and Potential Hardware Solution

Looking through the scenario and finding the client requirements, we suggest a potential hardware type by explaining how this hardware would satisfy each of the requirements.

Which hardware might be most appropriate for this scenario? (CPU / IGPU / VPU / FPGA)
CPU+IGPU

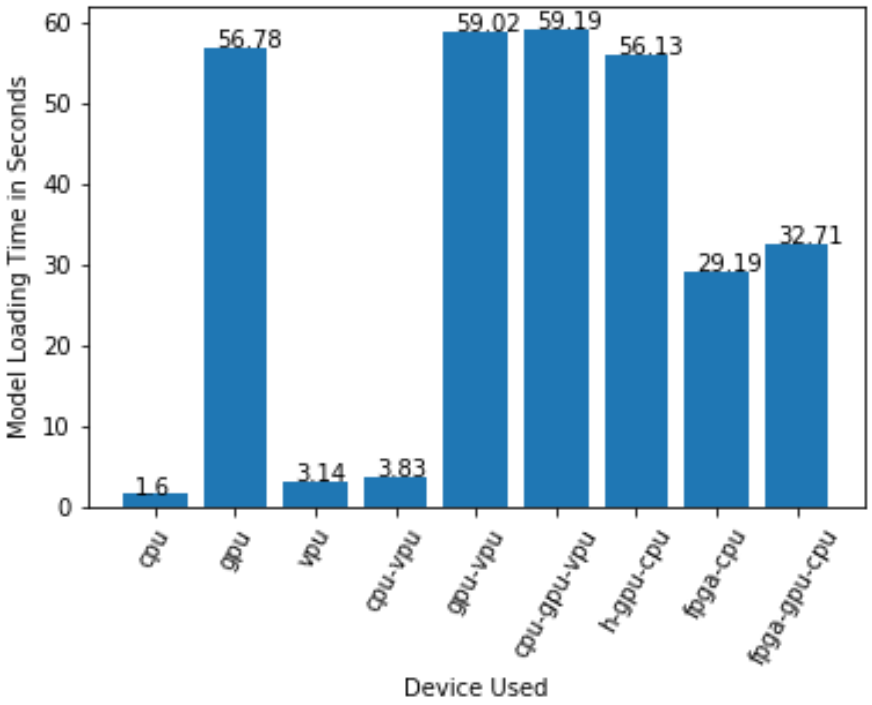
Requirement Observed (Include at least two.)	How does the chosen hardware meet this requirement?
average wait time: 230 seconds on the weekends: 350-400 seconds ➔ **Average performance (speed)**	On IGPUs, the Execution Unit instruction set and hardware are optimized for 16bit floating point data types. This improves inference speed, as we can process twice as many 16bit operands per clock cycle as we can when using 32 bit operands.
Most of the store's checkout counters already have a modern computer, each of which has an **Intel i7 core processor**. Currently these processors are only used to **carry out some minimal tasks that are not computationally expensive.** ➔ **Shared Components (CPU+IGPU)**	The CPU and IGPU share the same system memory, higher-level caches, and memory controller. This reduces memory latency by speeding up data transfer between the two devices.
Mr. Lin **does not have much money to invest in additional hardware** ➔ **No budget**	The CPU and IGPU are present on the same die.
and also would like to **save** as much as possible on his **electric bill**. ➔ **Configurable Power Consumption**	The clock rate for the slice and unslice can be controlled separately. This means that unused sections in a GPU can be powered down to reduce power consumption.

Queue Monitoring Requirements

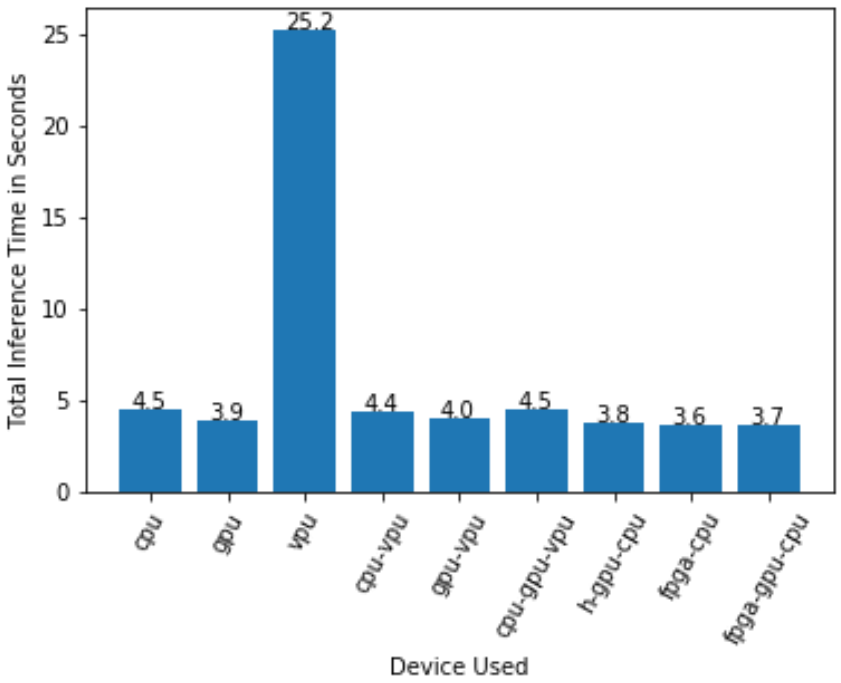
Maximum number of people in the queue	5
Model precision chosen (FP32, FP16, or Int8)	FP16

Test Results

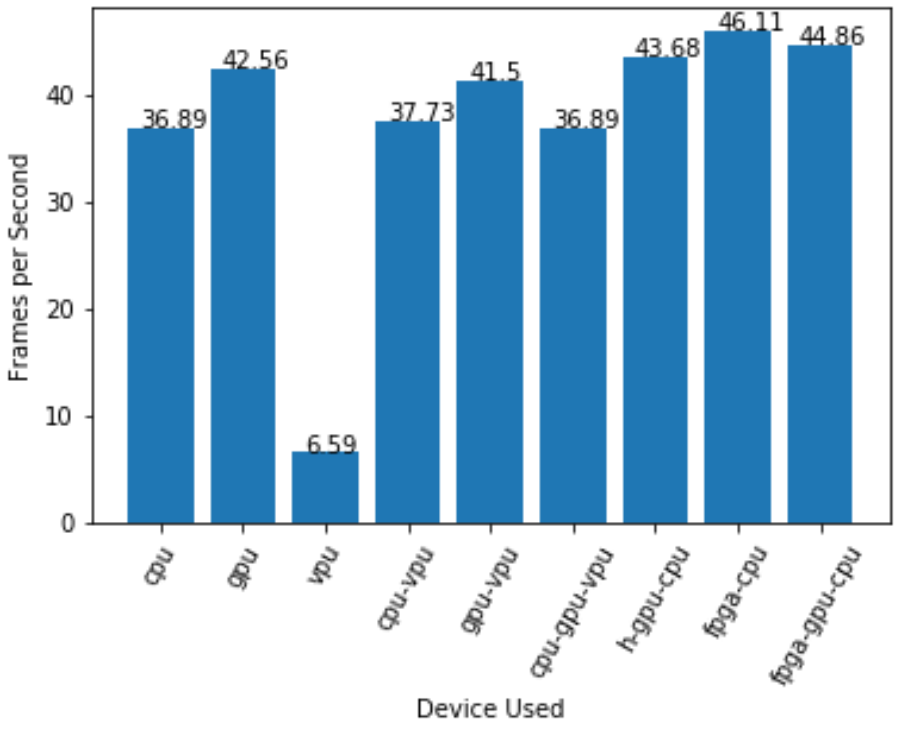
After the testing of the application on all four hardware types (CPU, IGPU, VPU, and FPGA), with the matplotlib output we can observe and compare the results below. There are three graphs (for model load time, inference time, and FPS).



Model Load Time



Inference Time



FPS

Final Hardware Recommendation

Synthesizing the points from above and we claim that the chosen hardware is the best choice for this scenario. The client's requirements, the test results, and how these relate to one another are summarized below.

Write-up: Final Hardware Recommendation
The client needs an inexpensive solution without new equipment and with low energy consumption. The HETERO:GPU, CPU seems to be the best no-new-equipment solution. However, it has high model load time, but it is also the best no-new-equipment solution.

Scenario 3: Transportation

Client Requirements and Potential Hardware Solution

Look through the scenario and find any relevant client requirements. Then, suggest a potential hardware type and explain how this hardware would satisfy each of the requirements.

Which hardware might be most appropriate for this scenario? (CPU / IGPU / VPU / FPGA)
MULTI: IGPU, VPU *(VPU:The processor in the NCS2 is the Myriad X VPU.)*

Requirement Observed (Include at least two.)	**How does the chosen hardware meet this requirement?**
Ms. Leah would like to automate this using an Edge AI system that would monitor the queues in **real-time** and **quickly** direct the crowd in the right manner. ➔ **On-chip memory for fast results**	The Myriad X has 2.5 Mbytes of on-chip memory. This is key to minimizing off-chip data movement, which in turn reduces latency and power consumption.
They monitor the entire situation with **7 CCTV** cameras on the platform. ➔ **Imaging/Hardware Accelerators to release-balance resources** ➔ **Vector processors (7 CCTV)**	VPUs have specialized accelerators for image processing. The Myriad X includes accelerators for functions like hardware encode and decode of H.264 and *Motion JPEG*, as well as a warp engine for handling fisheye lens, dense *optical flow*, and stereo depth perception. The Myriad X has sixteen proprietary vector processors known as *Streaming Hybrid Architecture Vector Engine* (*SHAVE*) processors. SHAVE processors use 128bit VLIW (Very long instruction word) architecture and are optimized for *computer vision workloads.*
The **CPUs** in these machines are currently being used to process and view CCTV footage for security purposes and **no significant additional processing power is available to run inference.** ➔ **Imaging accelerators to release-balance resources** ➔ **Scalability**	VPUs are specialized for image processing. One example of this specialization is found in the *imaging accelerators*, which have specific kernels that are used for *image processing operations*. These operations range from simple techniques for denoising an image, to algorithms for edge detection. Multiple NCS2s (or other Myriad X devices) will allow multiple inferences to run in parallel.
Ms. Leah's budget allows for a **maximum of $300 per machine** ➔ **Total cost for multiple NCS2**	Adding multiple NCS2s (or other Myriad X devices) will allow multiple inferences to run in parallel costing $70-$100 each (~3pcs).

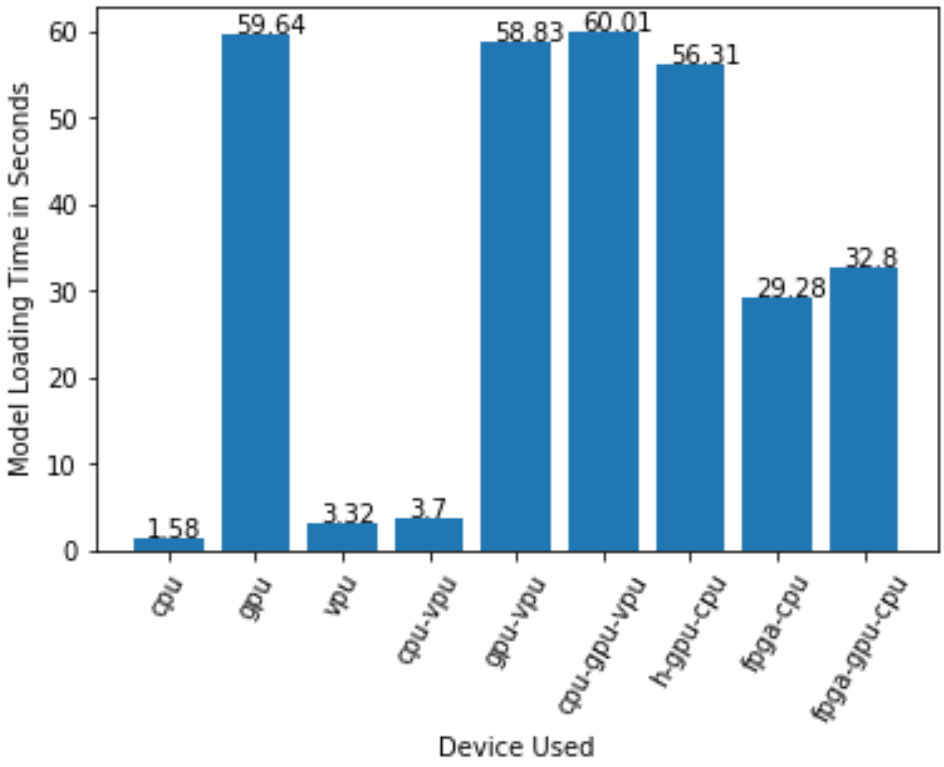
save as much as possible both **on hardware** ➔ **Cost**	Compared to other AI accelerators, the NCS2 is an inexpensive option, typically costing around $70 to $100.
future power requirements. ➔ **Neural compute engine with low energy consumption**	The Myriad X features a *neural compute engine*, which is a dedicated hardware accelerator optimized for running deep learning neural networks at *low power* without any loss in accuracy. Also, the Myriad X has 2.5 Mbytes of on-chip memory. This is key to minimizing off-chip data movement, which in turn reduces latency and *power consumption.* *The Myriad X has a very low power consumption of only 1-2 watts.*

Queue Monitoring Requirements

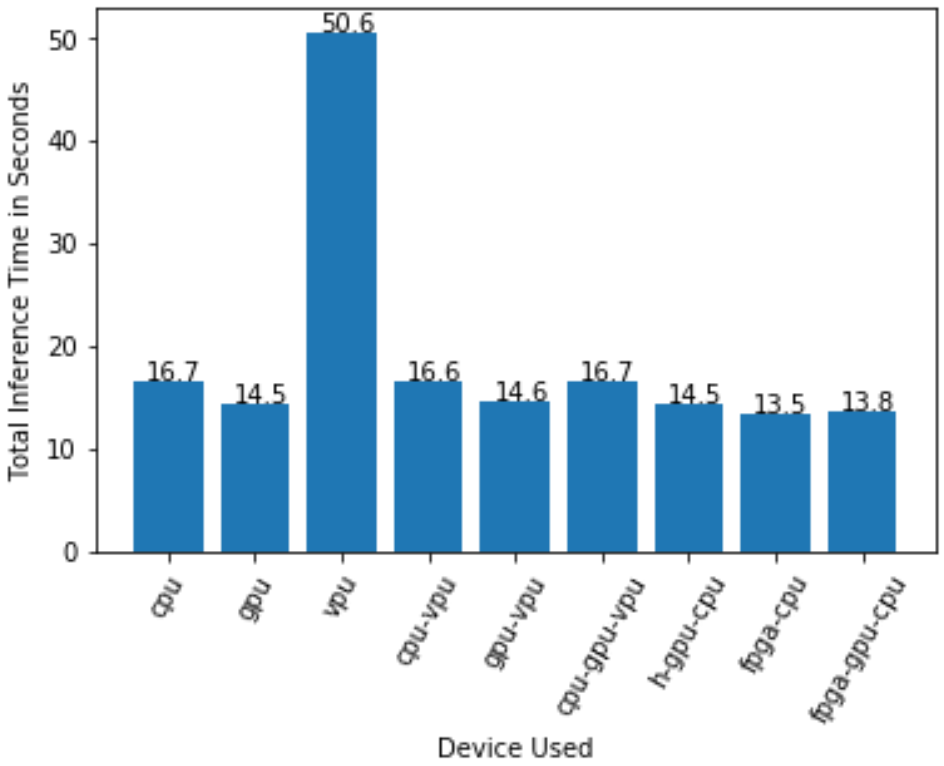
Maximum number of people in the queue	7
Model precision chosen (FP32, FP16, or Int8)	FP16

Test Results

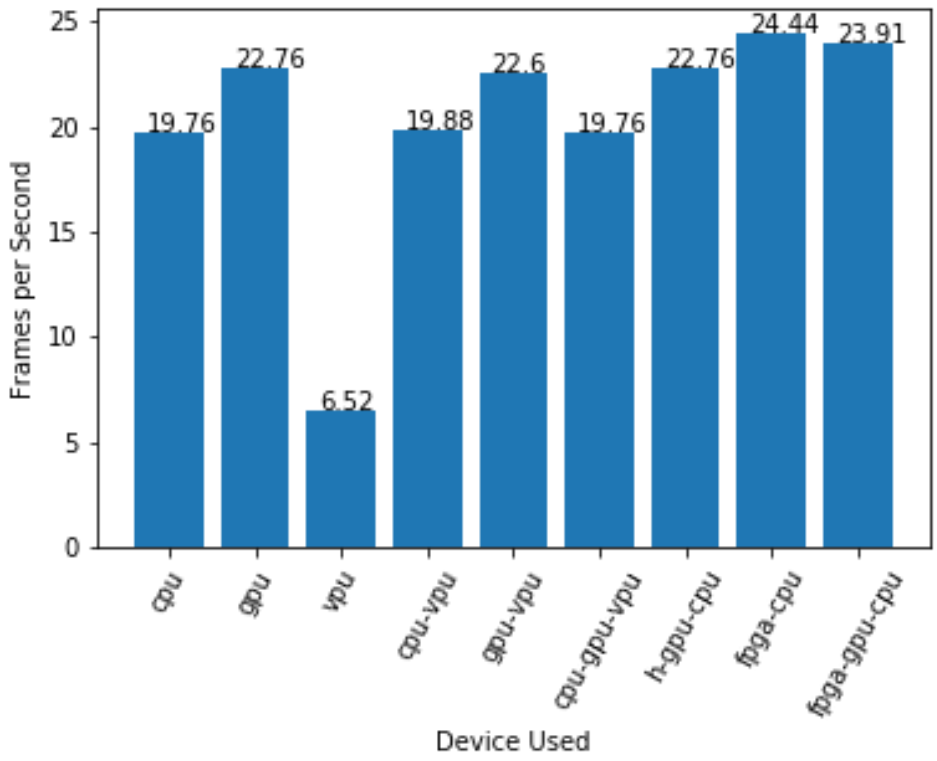
After you've tested your application on all four hardware types (CPU, IGPU, VPU, and FPGA), copy the matplotlib output showing the comparison into the spaces below. You should have three graphs (for model load time, inference time, and FPS).



Model Load Time



Inference Time



FPS

Final Hardware Recommendation

Now synthesize your points from above and provide a brief write-up describing why the chosen hardware is the best choice for this scenario. Be sure to discuss the client's requirements, the test results, and how these relate to one another (e.g., perhaps one of the devices performed better than the rest, but does not meet one of the client's requirements).

Write-up: Final Hardware Recommendation
The CPU can share limited resources for image processing. The majority of options condemned due to cost and power limitations. The best option is a MULTI: IGPU, VPU combination or a HETERO: GPU, CPU compination. However, CPU is used for security purposes and no significant additional processing power is available to run inference. To sum up, the MULTI: IGPU, VPU combination is the best solution.