Assignment: ML-Compiler Optimizations

1. Write a PyTorch model or download from HuggingFace (Ex, `resnet18`, `bert`, `gpt2`, etc), consisting of basic layers (e.g., linear, ReLU, matmul, convolution). Use Torch's **`torch.fx`** toolkit to trace and print the computational graph. Comment on redundant or chainable operations. Write a short explanation (5–6 sentences) describing its structure.
2. Generate torch-IR with the fx frontend for the above model using the torch-mlir compiler. Use stable or nightly Python wheels to install `torch-mlir` (https://github.com/llvm/torch-mlir) in a Python environment. Propose some optimizations that can improve the inference performance for any target hardware.
3. In your FX or MLIR output, suggest *one* or *two* pairs of operations that could be fused. Explain what effect this fusion would have on memory access and kernel launch overhead, discussing potential fusion opportunities and their benefits. Also, mention any additional memory optimizations applicable. Specify one hardware backend (CPU, GPU, or TPU) that would be most suitable for achieving optimal performance and why.