**Project Overview:**

**Problem Statement**: To accelerate the inference process of a CNN model

**What we have done:**

* We did literature study about the different methods of accelerating the inference process.
* We decided to apply the method of data quantization for accelerating the inference process.
* We have chosen the LeNet1 Architecture for this purpose.
* Firstly we have used the Half Precision method to express the floating point numbers using 16 bits - (1 sign bit + 5 exponent bits + 10 mantissa bits) and truncated the 5 mantissa bits to make it 11 bits - (1 sign bit + 5 exponent bits + 5 mantissa bits) for the purpose of acceleration
* In the first week, we have implemented the LeNet1 Architecture in Python using Pytorch and extracted weights and biases after training and validating it.
* Then we started implementing the inference model of LeNet1 with Half Precision Floating Point data.
* The first layer is the convolution layer:
  + It takes 28 x 28 x 16 bits as input image and uses four 5 x 5 x 16 bits kernels to give four tensors having 24 x 24 x 16 bits as output after convolution.
* The convolution output is then passed through the ReLu activation function.
* The second layer is the average pooling layer:
  + It takes one tensor having 24 x 24 x 16 bits as input and gives one tensor 12 x 12 x 16 bits as output.
  + This process is repeated for the rest 3 tensors.
* The third layer is the convolution layer:
  + It takes four tensors having 12 x 12 x 16 bits and gives one tensor having 8 x 8 x 16 bits after convolution by four kernels having 5 x 5 x 16 bits.
  + This process is repeated 12 times to generate 12 feature maps having 8 x 8 x 16 bits as output.
* The output us then passed through the ReLu activation function
* The fourth layer is the average pooling layer:
  + It takes 12 tensors having 8 x 8 x 16 bits and gives 12 tensors having 4 x 4 x 16 bits as output.
* These 12 tensors are reshaped to 1 x 192 x 16 shape.
* The last layer is the fully connected layer:
  + It has 192 nodes each having 16 bits.
* It gives ten 16 bit values which are compared using comparator and gives the maximum value as output and the detected digit.

At last, we have converted the 16 bits input to 11 bits input for the purpose of inference acceleration.