In this document I have penned down my approach towards this problem (4.2 CV):

LeNet is a simple convolutional network which can help in large scale image processing. The lenet-5 which I have decided to use have 7 layers to it, which include 3 convolutional layers, 2 sub-sampling (pooling) layers and 1 fully connected layer, that are followed by the output layer. The convolution layer with six kernels (I have tried changing the number of kernels in my code) of 5*5. I have set the padding to "same" so that the input and output volume size is the same. I have used the relu activation function.

Task 1:

The cifar10 dataset was acquired from keras.dataset . Then I had to analyse the dataset, its shape, its labels and what the data looked like. Then the data was normalised by using the min max scaler (dividing it with max pixel value i.e. 255). I made the model described above using keras. Since the labels were numbers from 0-9 they had to be converted categorical. The results were -

- 61 % Accuracy on the test set which used as the validation data with LeNet without data Augmentation, using Adam optimiser.
- 55.24% Accuracy using SGD (Stochastic gradient descent) on the test set which used as the validation data with LeNet without data Augmentation, using Adam optimiser.

For the rest of the project the SGD model was used.

Task 2:

I learned about DCGAN (Deep convolutional Generative Adversial Network) from https://www.tensorflow.org/tutorials/generative/dcgan. Hence, I have used the keras Sequential API to implement it. The main logic behind the dcgan is the Generator and discriminator. The dcgan was used on cifar 10 to create new images. The dcgan generator model is stored in generator_model.h5 file.

Note: This task isn't completely done since the images generated by dcgan is not labelled.

Task 3:

To understand the importance of Data augmentation on the accuracy, I augmented the dataset with the flipped images so that the model becomes better in generalising.
Keras has a feature called ImageDataGenerator which easily helps in preprocessing data, using this I flipped the images and concatenated it with the training datset. The accuracy increased to 58.77% (about 3% increase).

BONUS:

Task 1:

By tweaking the parameters in the ImageDataGenerator (i.e. by zooming, rotating the images randomly etc) the model attained an accuracy of 62.45% (about 8% increase from initial model accuracy).

Task 2 and 3:

The LeNet architecture is implemented on the MNIST dataset. For this dataset the pre-processing part was very similar to cifar10.

The exact same model (the one used on cifar 10) was used on the MNIST dataset, and the accuracy was found to be significantly higher, 98.72%.

But, after augmenting with the horizontally flipped images the accuracy decreased to 97.11%. Hence, the effect of data augmentation seems to depend upon the data itself. But after a better augmentation after rotating, zoom, and shifting height and width; the accuracy increased to 98.94% (minor increase).

The accuracy after flipping increased of the cifar 10 dataset but decreased for the MNIST dataset. Even after a further augmentation the accuracy increase was marginal for MNIST.