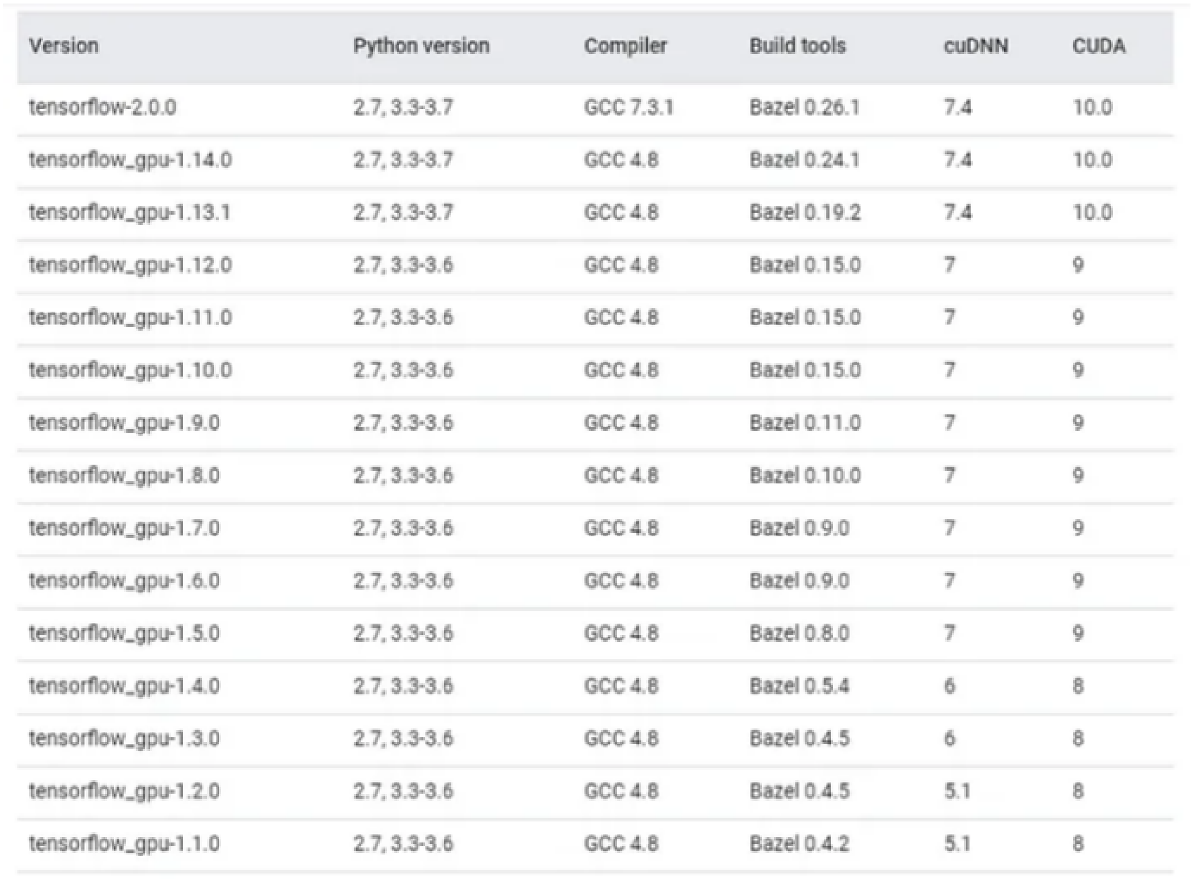
Steps:

1) Check GPU compatibility with tensorflow GPU. Visit this link to find about it. https://developer.nvidia.com/cuda-gpus
2) Switching onto the graphics card. It can be checked through the command nvidia-smi.
3) Check cuda compatibility with the graphic card. Visit this link to find about it. https://docs.nvidia.com/deploy/cuda-compatibility/
4) According to CUDA, find the TF version



Version	Python version	Compiler	Build tools	cuDNN	CUDA
tensorflow-2.0.0	2.7, 3.3-3.7	GCC 7.3.1	Bazel 0.26.1	7.4	10.0
tensorflow_gpu-1.14.0	2.7, 3.3-3.7	GCC 4.8	Bazel 0.24.1	7.4	10.0
tensorflow_gpu-1.13.1	2.7, 3.3-3.7	GCC 4.8	Bazel 0.19.2	7.4	10.0
tensorflow_gpu-1.12.0	2.7, 3.3-3.6	GCC 4.8	Bazel 0.15.0	7	9
tensorflow_gpu-1.11.0	2.7, 3.3-3.6	GCC 4.8	Bazel 0.15.0	7	9
tensorflow_gpu-1.10.0	2.7, 3.3-3.6	GCC 4.8	Bazel 0.15.0	7	9
tensorflow_gpu-1.9.0	2.7, 3.3-3.6	GCC 4.8	Bazel 0.11.0	7	9
tensorflow_gpu-1.8.0	2.7, 3.3-3.6	GCC 4.8	Bazel 0.10.0	7	9
tensorflow_gpu-1.7.0	2.7, 3.3-3.6	GCC 4.8	Bazel 0.9.0	7	9
tensorflow_gpu-1.6.0	2.7, 3.3-3.6	GCC 4.8	Bazel 0.9.0	7	9
tensorflow_gpu-1.5.0	2.7, 3.3-3.6	GCC 4.8	Bazel 0.8.0	7	9
tensorflow_gpu-1.4.0	2.7, 3.3-3.6	GCC 4.8	Bazel 0.5.4	6	8
tensorflow_gpu-1.3.0	2.7, 3.3-3.6	GCC 4.8	Bazel 0.4.5	6	8
tensorflow_gpu-1.2.0	2.7, 3.3-3.6	GCC 4.8	Bazel 0.4.5	5.1	8
tensorflow_gpu-1.1.0	2.7, 3.3-3.6	GCC 4.8	Bazel 0.4.2	5.1	8

5) Download the CUDA toolkit. https://developer.nvidia.com/cuda-toolkit-archive
6) CUDA installation. Add a path to bashrc for Cuda access. Check with nvcc -version.If successful, it will display the CUDA version.

7) Download cudnn compatible with cuda
   https://docs.nvidia.com/deeplearning/cudnn/archives/index.html

8) Install Miniconda and create a separate environment with a specific Python version.

9)Check which devices are available and GPU information.

tf.config.experimental.list_physical_devices()
is_gpu = len(tf.config.experimental.list_physical_devices('GPU')) > 0