Design Document:

1. Used the site : CHI@UC

2. Instance Information-

   Image Name- CC-CentOS7-CUDA10

   CentOS7 was compatible with my GIT version and could run at a lower version of Linux kernel version- 3.10

3. Generated a public key which was later used for authentication in BITVISE.

- Tried using the shared memory in CUDA, but did not see how possibly it is going to effect the efficiency of the program.
- Also could not figure out how many threads were actually working and sitting idle for certain number of N values.

- Also synchronization after the mean and after the standard deviation calculation did not did not have any major effect for small matrix size.

Noticed the maximum time taken was around cudaMemcpy function- around 70%- 75% program run time when executed the nvprof command.

Even cudaMalloc took significant chunk of run time.

In some of the articles it is suggested to use cudaHostAlloc or cudaMallocHost, because it allocates the pin memory which GPU can access faster- something like indexing.