**Logs Getter:-**

The main point that I wanted to focus on while building this tool was the time it takes while getting the logs from the file. I already knew about the goroutines and new that it can be helpful in this. Following are my findings and how my solution works:-

* The first thing I knew that if the file is too big then it was impossible to get the whole file and process it so I started looking for efficient solutions on how can I read the data in chunks from a file, here I came across buffered IO reader in go. Using this I was able to read a chunk of data from the file and process it. This I used with goroutine to process multiple chunks of same file at a time and start printing the data as soon as it finds the match
* Then I knew that the dates in the file are in increasing order so the first date in the file is the smallest one. I wrote the code to get the first line and check it’s date, if it is after the end of the date range we can skip the file altogether and if it is in the range process its data.
* Then I wrote code to loop through all the files and pass it to the function which is a go routine that does the first log date check.
* So basically it creates a goroutine for every log file and then for every go routine it creates sub goroutines which will process chunks of data in the file.