# LOG\_PARSING

Design and Executed an Log Parsing atomization system to Parse HTML log format to Excel format. Converted logs to most readable format for user so that different log can be segregated based on log level such as Debug/Exception/Error/Warning. This application reduces the manual efforts of categorizing the log based on its severity level.

**Log File Description:**

Log file content is coming in Html format, Actual log content is Fixed HTML Table Format. Log file may consist of multiple Tables in different HTML tags, Number of log Table may Vary according to log file size.

**Log Tables Format**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Transaction\_Id | TD\_SessionId | Thread\_Name | Level | Message | Stack\_Trace | CallTraceResponse |

Transaction ID: System Generated Number

TD Session ID: Time stamp when log is traced

Thread Name: Tread Level

Level: Log Level [Error/Exception/Debug/SQL/Warning]

Message: File of workflow description where error/exception occurred

Stack Trace: Details of Log

Call Trace Response: Detail Response capture

**Log Parsing Process**

* Log file is loaded to python environment.
* Log file has been parsing through html Parser.
* Each row from each html tag has been parse through html parser and formatted in python Dataframe.
* Each row has been identifying from HTML format and stored in Pandas Dataframe
* Once Complete table Parse and Stored in Dataframe, segregation process Can be started to store different log level (ex. Error log stored in different object)
* Different log object Saved to Excel file.

**Input to Program:**

file\_path: Location of Log File

Ex: C:/Desktop /log\_automation/DCR\_Service.log

target\_path: Location of Directory where target file should be created.

Ex: C:/Desktop/log\_automation

**Output:**

Excel file will be generated same as log file name in Target Directory with Different Excel Tab such as Error/Exception/Debug/Warning