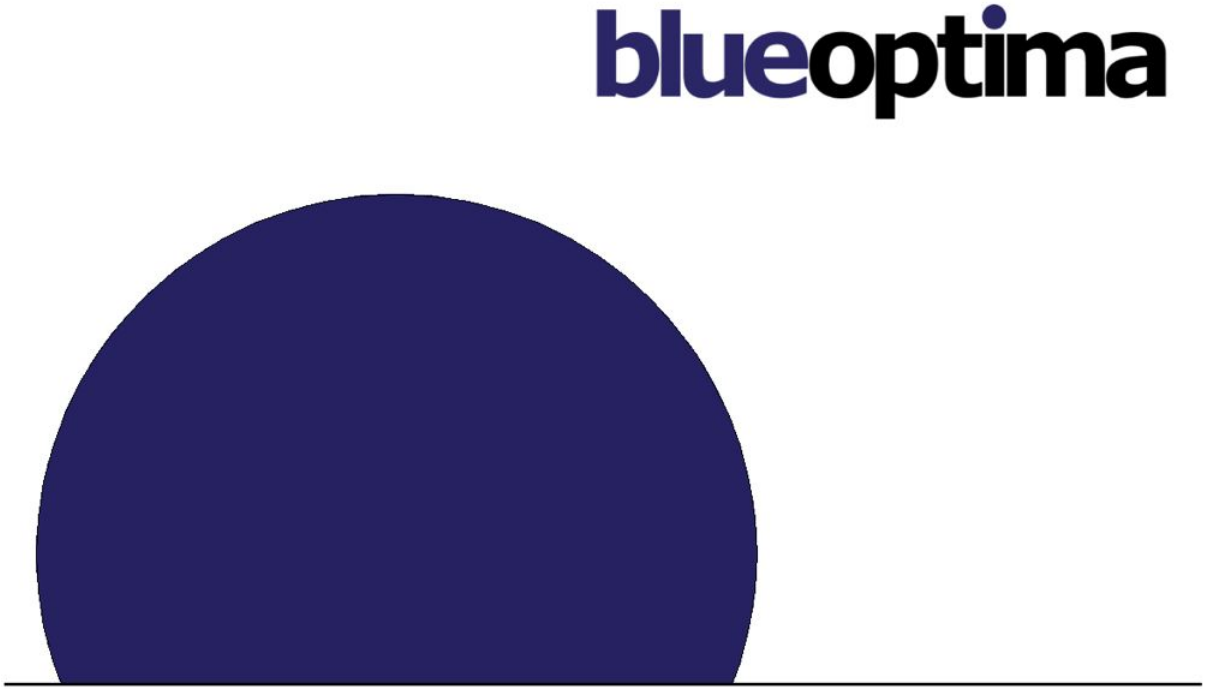

Log Analyzer,

Version: 1.0

Released: Tuesday, Sep 18, 2018

Scenario

Different application have different logging strategies and logs are subsequently generated for various different reasons. While some logs are used to trace an error back to origin, there are logs that are generated that can help understand performance of different parts of applications.

Problem

Following is an extract from a log file that records User's activities on a web application:

```
2018-09-18 04:49:38,215 ERROR (default task-95)
IP-Address=157.49.141.133#,!User-Agent=Mozilla/5.0 (Windows NT 10.0; WOW64;
Trident/7.0; rv:11.0) like
Gecko#,!X-Request-From=UIX#,!Request-Type=POST#,!API=/v1/admin/developers#,!U
ser-Login=test@demo.com#,!User-Name=testUser#,!EnterpriseId=2#,!EnterpriseNam
e=Enterprise-2#,!Auth-Status=#,!Status-Code=200#,!Response-Time=346#,!Request
-Body=

2018-09-18 04:49:39,842 ERROR (default task-21)
IP-Address=157.49.141.133#,!User-Agent=Mozilla/5.0 (Windows NT 10.0; WOW64;
Trident/7.0; rv:11.0) like
Gecko#,!X-Request-From=UIX#,!Request-Type=GET#,!API=/v2/developers#,!User-Log
in=test@demo.com#,!User-Name=testUser#,!EnterpriseId=2#,!EnterpriseName=Enter
prise-2#,!Auth-Status=#,!Status-Code=200#,!Response-Time=1240#,!Request-Body=

2018-09-18 04:49:41,946 ERROR (default task-127)
IP-Address=157.49.141.133#,!User-Agent=Mozilla/5.0 (Windows NT 10.0; WOW64;
Trident/7.0; rv:11.0) like
Gecko#,!X-Request-From=UIX#,!Request-Type=GET#,!API=/v2/developers#,!User-Log
in=test@demo.com#,!User-Name=testUser#,!EnterpriseId=2#,!EnterpriseName=Enter
prise-2#,!Auth-Status=#,!Status-Code=200#,!Response-Time=1117#,!Request-Body=
```

To analyse these logs manually takes significant effort and time. It requires a developer to extract the logs, parse and review them, making it significantly time consuming.

Work Sample Objective

Design a Log analyzer which takes a file as Input, extracts, parses and stores data in a format that can be used to easily filter out activities based on different parameters and the final result should contains atleast following information

1. IP Address
2. User Agent
3. Status Code (200/401/500 etc).
4. Request Type (GET/POST/PUT etc)
5. API
6. User
7. Enterprise Id
8. Enterprise Name

Please note that the input files provided can be big, i.e. with size ranging from few MBs to as big as 1 - 2 GB.

Deliverables

Source code:

1. A comilable and working solution with source code
2. Source code should be zipped and shared either on email or a shareable google drive link. (Do not share the solution or put the code on GitHub or other code sharing system)
3. Readme file detailing steps to configure an Eclipse or Intellij (or relevant IDE) for the source and steps to compile it
4. Any dependency on separate application or a server should be listed.

Project Documentation should include:

1. Explain the approach taken, including (but not limited to) understanding of rate limits, your approach to manage it, information being processed and what it represents.
2. Data storage format and reasons for the approach. If alternatives were explored, provide them as well.
3. Any assumptions made.
4. Suggest any additional parameters that could be additionally taken as input to improve the accuracy of the solution.
5. State any improvements that you would like to make if more time was available for implementation.

Evaluation

Your submission will be evaluated on completing all the items specified in deliverables list. Once the submission is made, we will arrange a follow up discussion to understand from you the solution itself and ask any question that come out of the submission you make.

Important Information:

To make evaluation accurate and less time consuming please take note of following.

1. **Code formatting:** Please indent the code (using tabs or space) and use camel case when defining variables and functions.
2. **Documentation:** Please document the code where necessary and document it just enough. Excessive documentation is worse than no documentation.
3. **Summary:** Please explain on how to build and execute the code. How and what inputs to be passed and where output can be seen. A general execution flow will also be good for the evaluator to understand the solution that is being provided. As mentioned above, this needs to be just enough to make evaluation effective and avoid excessive documentation.