# Logging For Testers

As a tester have you ever worked on testing a feature or application and thought to yourself, "I wonder who uses this?"?

I've had these thoughts in previous roles. I think this particularly applicable to testers who work on enterprise systems meant for specific business domains. In a previous role, I worked on project management application for accountants. As someone who's not an accountant, I didn't have any experience or even business knowledge of this product. How accountants use the product on a daily basis was a mystery to me. I wondered often, "Who uses this?"

There are several ways testers can answer the questions of who is using an application and how a user actually uses an application. One approach that is becoming more commonplace is to analyze production logs of an application. Logs are lists of lines of text output from an application in a given environment such a test server or production server. These can be helpful for testing purposes because they provide real feedback and information from an application as its being used. In addition to giving insight for how an application is being used, logs can provide information that can describe or even help solve bugs that arise in the application.

Each line of a log corresponds to some event or occurrence in the application. A log line could be informational ("A user successfully logged in at "1:00 PM EST"), a warning ("The current number of users is 90% of the total allowed concurrent users"), or an error ("A valid user login failed unexpectedly"). Log entries can be output come from the application itself ("The number of logged in users at a given time has reached a hard-coded limit") or from the application's environment or system running the application ("The server has run out of memory and cannot allow any more users to log in"). As well, most logging systems provide a time stamp for each log entry, often to the millisecond and each log entry follows some standard format. This can provide much insight into the question of "Who's using this application?"

## An Example App with Logging

As an example, here's a simple application for generating random numbers called [Corsica](https://github.com/joshmgrant/corsica). This application generates random numbers based on an HTTP request and provides these values in a response. Let's say this application gets two requests, one valid and invalid. If we take a look at the logs provided in the application infrastructure, we see the following lines:



Here are two lines that correspond to each request. Each line has the same format, staring with a time stamp of when the request was sent. The valid request returned an HTTP status of [200](https://http.cat/200) while the invalid request returned an HTTP status of [404](https://http.cat/404). You can also see additional information such as the fact that each request originated from a Chrome version 57 browser and what the end of the HTTP request URL looked like. This log contains other lines as other requests are made, each of which has information similar to these two highlighted lines.

In the case of our sample application, logs are generated automatically as part of application hosting service (Google App Engine) and can be accessed by members of the project. This is true for many hosted web applications but often logs are generated and kept internally on a team.

Logs are a great place to find information to help answer the question "Who uses this application?" What are some approaches we can take to answer this question?

## Answering the Question Qualitatively

One approach to answering the question of "Who uses this application?" is to look at log data overall and find possible patterns. One of easiest pieces of information is looking at how large a log is. If an application's log can fit on one or two pages in a word processor, chances are the application is being lightly used. If an application's log cannot be easily stored on a single hard drive, chances are that the application is being heavily used. This may provide a first step into how *much* an application is being used. This can be an insightful, if surprising, way to find out how active users are on a given application.

An application's log size can also be an indication of an application's age. It's probably unlikely an application with one million log entries was written an hour ago, just as an application with two log entries is unlikely to have been written twenty years ago.

Another easy way to get a hold of how an application is being used is to check how fast log entries are being added. This can give some indication of how many users are *currently* using an application. A log may be large but if no entries have been added in the past week, the application may not be too active now. There may also be patterns in log entry speed. Some applications may have more entries added during an eight-hour period that corresponds to working hours than outside of this window. This may provide some insight to when users are actively using an application.

After looking at these relatively easy approaches to a log, you could take an exploratory approach to looking at logs. A good idea to dive in and see if you can find any patterns upon inspection. You could take a set number of entries that are most recent or from a given period of time and see if anything jumps out. Are most log entries errors, warnings or something else? Do most log entries look the same? Are there any particular messages that appear more or less often? Are log messages easily understandable or not? This may be a good opportunity to apply [blink testing](http://www.satisfice.com/blog/archives/33). This could provide a basis for further investigation and tie back to the question of "Who's using this application?" and how users may use it.

## Answering the Question Quantitatively

Looking at logs from a more quantitative view can also provide great insights. Here, we're looking for hard numbers instead of comparisons. A tester may want to know precisely \_how many\_ errors were logged in a particular time frame. Finding three errors logged in the past week provides a different state of affairs than having six thousand logged in the past week. For sufficiently small logs or logs that grow slowly, you can do some quantitative analysis by hand but most of the time this is impractical. You'll have to look at the entire log to get good quantitative data.

For many testers, this approach may seem daunting. How would you get data like that? Remember that log entries are (typically) standardized. This means logs are a great candidate for parsing using scripts. You could use an off-the-shelf parser or write your own. Some modern logging systems have built-in tools that developers can make use of, so it's worth talking to the individuals responsible for logging to see what data you can pull out.

Log parsers can generate data sets that can give you specific insights to how users are using an application. In the past, I've seen data presented like this from production logs. The most common API call made was the *login()* call, which makes some sense given that the application was a proprietary web application used in large companies. All users would have to login to the application before doing anything so this is a helpful check on our testing assumptions. This data was presented in graph form, with various charts representing the number of calls being made to some server-side APIs of interest. This was really helpful in figuring out what APIs were being used the most and the least, which in turn gives some indication of which parts of the application users are using the most and the least.

## Conclusion

If you want to answer the question "Who uses this application?" there's many approaches. Analyzing logs can help provide answers to this question. The main advantage to analyzing logs is that testers get real user data. Instead of guessing or hoping they know how a user uses an application, logs show which parts of the application are getting used and what's going on with the application in production. Testers can use this data to inform other testing practices too, such as generating better test data or creating new exploratory test themes.

Most of all, it may give testers some assurance that someone is actually using the application or feature they're testing.