Some possible observations include:

> *Every operation took at least 0.9 milliseconds, and every operation finished in under 4.2 milliseconds.*
>
> *This distribution is bimodal, with one cluster centered around 1 millisecond and another around 1.6 milliseconds. Any operation that had not completed after a millisecond was most likely to take 1.6 milliseconds overall.*
>
> *This distribution is not symmetric: both clusters of the histogram look like bell curves that have been skewed right.*
>
> *Additionally, the distribution also has a tail from 2-4 milliseconds. These requests took much longer than average, but there are not many of them.*

Tech fact: Are you wondering why the distribution is bimodal? Here's why …

Lots of Internet software can make educated guesses about what users are about to do, even if they can't make those predictions with 100% accuracy. For example, the people who programmed YouTube *know* that YouTube will recommend a video to automatically start playing next after your current video finishes; they programmed YouTube to do that, after all! So, compared to the millions of other videos on YouTube, the video YouTube recommended to play for you next is most likely *by far* to be the next video you play. Since YouTube can be reasonably confident that video will be played soon, it can start working on that video ahead of time. If it anticipates correctly (i.e. you do let that video auto-play next), then your request for the video completes faster than normal, because some of the work was already done by the time you requested it.

It's likely something similar is going on in the software program this histogram was describing. From the histogram, I'd estimate it usually takes about 1.6 milliseconds for this program to process a request it did <u>not</u> anticipate; but, any time the program was able to correctly anticipate a future request, it was able to get about 0.6 milliseconds of the work done ahead of time – as such, those correctly anticipated requests only took (1.6 – 0.6 =) roughly 1 millisecond overall.

We end up with a bimodal distribution: one cluster for the "fast" requests that were anticipated correctly, and another for the "slow" requests which the software did not anticipate.