# Test Driven Development (TDD) Works

The current Engineering interviewing process, when exploring white-board algorithm implementations, often reveals aspects of engineering not considered by either the interviewer or candidate prior to the actual interaction. The two most common ignored issues in my experience are: a lack of testing approach (e.g., *TDD*) when white boarding, and sometimes limited discussion of performance considerations.

I would like to use a recent interview experience to highlight both topics, and consider TDD in more detail. We will first review the design session interactions, briefly consider performance implications, and finally consider *TDD* aspects. You can jump ahead to the TDD discussion by jumping to title “XXXXX” below.

## The Initial Problem

We must count unique client IP references to our web-site over some time period (a day let’s say). Note that we are only considering 32 bit IP address. We are not required to persist the IP access counts outside of the session. We will need to support various query types (e.g., counts for a specific subnet of the client URL.)

## Initial Proposed Solution

Use an In-Memory data grid to record client references ([Client-URL, Access Count] pairs.) A Data Grid platform offers a mechanism to store counts as necessary, and handles both distributing counting across monitored servers, and an aggregated view of the counts. The platforms support complex queries as well. Please review reference #1 for a background on Data Grid platforms.

## The Modified Problem

Imagine we have a single server and we must provide our own solution. What would be an appropriate data structure to store client URL and access counts? How might we query for subnet usage accounts? A subnet is a group of 32 bit client IPs beginning with the same bit pattern. Please see reference #2 for an explanation of subnets.

## White Board Solution

We would record the client URL references in a ***HashMap*** that used the 32 bit URL as a key and kept the access counts as the value associated with the client URL. We would obtain the subnet counts with this process:

1. Xxxxxx
2. Yyyyyy
3. Zzzzzzz
4. wwwww

Please see reference #3 for the detailed code implementing the flow description above.

## Resources

1. An in-memory data grid overview and examples: <https://www.predictiveanalyticstoday.com/top-memory-data-grid-applications/>.
2. Subnet of IP explanation: <https://www.pcwdld.com/subnet-mask-cheat-sheet-guide>.
3. GitHub code repository parent for this article (and others) is: <https://github.com/DonaldET/DemoDev>, and the topic of this article is based at <https://github.com/DonaldET/DemoDev/tree/master/dev-topics-codingexams/dev-topics-liveramp-bitsearch>.
4. sssss