

Code Scalability

Case Reference: Google



Litcoder Parameter

Code Manageability focuses on the ease with which the codebase can be managed, maintained, and extended over time. It advocates for modularization, separation of concerns, and the use of design patterns to break down complex functionalities into smaller and more manageable units.

Code scalability refers to the quality of the code base to easily scale to higher levels of user demands, including greater volume and more user functionalities. Scalability issues surface in production and might be associated with flaws such as a memory leak, or an unclosed DB connection.

About Google

Google is one of the most recognizable and influential companies in the world, renowned for its innovative technologies that deliver a vast array of digital services. Google processes over 3.5 billion search queries daily through an intricate architecture optimized for rapid response speed i.e. low latency, accuracy, and scalability to massive search traffic volumes. Its backend architecture spans over an estimated 1.4 million search servers deployed across 35 regions 106 zones, and 173 network edge locations serving customers in over 200 countries and territories.

Problems Faced by Google

Despite maintaining a 99.9% uptime benchmark, Google Search faced acute issues, including an outage at its London data center due to the simultaneous failure of redundant cooling systems. Additionally, concerns have been raised regarding quality of search results such as the racial, gender and ideological biases exhibited by Google's AI algorithms in search suggestions and image labeling. This has led to heightened public scrutiny, with government regulators and online advertisers filing suits over alleged anticompetitive practices and conflicts of interest, potentially necessitating unbundling in vertical search domains. Left unchecked, architectural technical debt, outdated disaster recovery provisions, and centralized integrity risks pose significant threats to Google's brand reputation.

Some reasons for these include technical debt emanating from expedient hacks and workaround patches shipping without review and tight stack couplings. These had accumulated as patchwork upgrades and architectural shortcuts were taken to meet unrealistic deadlines. Centralized integrity risks grew rapidly as competing product teams duplicated critical functions such as identity, storage, and analytics in their cloud silos instead of sharing common services.

Solution Adopted by Google

Google aimed to power its search infrastructure through architectural decentralization, improved software development practices, and organizational realignment. The company utilized an analytics Lake House as a centralized data repository to enhance compartmentalization and address data silos. They engaged in quarterly engineering satisfaction surveys to effectively manage technical debt and implemented regulated failover redundancy for all critical path components, mitigating single points of failure. Additionally, Google transitioned towards DevSecOps collaboration between security, infrastructure, and product teams when developing new features. Collectively, decentralizing risk while enhancing engineering rigor, resilience testing, and reliability guardrails aims to correct Google's structural oversights. However successful transformation still hinges on sustained culture change.

In response to the search reliability breaches, Google initiated substantial outreach and legal appeasements. They announced a public roadmap outlining plans for search architectural updates and cybersecurity improvements, acknowledging and apologizing for failing user trust while promising sustained enhancements. Furthermore, to address algorithmic bias, Google formed an independent advisory panel comprising diverse external civil rights leaders to provide ongoing transparency and input regarding fairness considerations in search and AI.

References: [Click Here](#)

