## Cheatography

## System Design Template Cheat Sheet by vinu76jsr via cheatography.com/163274/cs/34196/

Compatibility (OS, etc)       reliability         Availability (OS, etc)       Maintainable         Availability       Maintainable         Availability       Scalability         Usage able       Environmental         Storage       Memory Estimates (Cache, what kind of data in Estimates)         Storage       Memory Estimates (Cache, what kind of data in Estimates)         RAM       SSD or disk         Storage       Scenarios to not cover         who will use       How many will use         Usage patterns       How many will use         Usage patterns       How many will use	0. Non-functional F	Requirements (NFR	s)	4. High Level Design	
Availability       Maintainable         Manageable       Scalability         Jsability       Performance         Regulatory       Environmental         2. Estimations       - scaling algo         Throughput       Latency         QPS, read/write)       - Availability, consistency and scale story of each components         QPS, read/write)       - Availability, consistency and scale story of each component         QPS, read/write)       - Components         QPS, read/write)       - DNS - CDN (push vs pull)         RVW ration       Traffic Estimates - write(QPS, volume of data) and read (QPS volumen of data         storage       Memory Estimates (Cache, what kind of data in Estimates it)         RAM       SSD or disk         Features and Expectation       - App layers scaling (microservices, service discovery)         Jsage patterns       - NoSQL (KV< wide column, graph, document	Security		Capacity	APIs for read write scenario	DB schema
Manageable       Scalability         Jaability       Performance         Regulatory       Environmental         2. Estimations       - scaling algo         2. Estimations       - scaling components         Throughput       Latency         QPS, read/write)       - Vailability, consistency and scale story of each component         Provide additional components       - scaling algo         RW ration       Traffic Estimates - write(QPS, volume of data         Storage       Memory Estimates (Cache, what kind of data in itstimates it)         Storage       Memory Estimates (Cache, what kind of data itstimates it)         Features and Expectation       - App layers scaling (microservices, service discovery)         Jase case       Scenarios to not cover         who will use       How many will use         Jsage patterms       - APP - Cassandra, RIAK         Jsage patterms       - CP - HBase, Mongo         A Design Goals       - Cient, CDN, server, DB, applicaiton, query, object	Compatibility (OS, etc)		reliability	Basic algo	HLD for Read/write scenario
Manageable       Scalability         Jsability       Performance         Regulatory       Environmental         2. Estimations       - scaling algo         Throughput       Latency         QPS, read/write)       - Components         Throughput       Latency         RW ration       Traffic Estimates - write(QPS, volume of data) and read (QPS volumen of data         Storage       Memory Estimates (Cache, what kind of data in it)         Storage       Memory Estimates (Cache, what kind of data in it)         Features and Expectation       SSD or disk         Features and Expectation       Scenarios to not cover         Abo will use       How many will use         Jsage patterns       Scenarios to not cover         B. Design Goals       - Apailability         Consistency vs Availability       requests	Availability Maintainable				
Sadinity Eventomatice   Regulatory Environmental   2. Estimations - scaling algo   2. Estimations - Availability, consistency and scale story of each components   Components - Components   - DNS - CDN (push vs pull) - EB (active passive, active active, L4 , L7)   - RAM read (QPS volume of data   Storage Memory Estimates (Cache, what kind of data in Estimates   Estimates it   RAM SSD or disk   Fedures and Expectation Sage Sage opatients Sage patterns Sage Sage Satery vs Avaluations Storage Sage Satery vs Avaluations (Cache, what kind of data in test)     Storage Memory Estimates (Cache, what kind of data in test)   - Storage Memory Estimates (Cache, what kind of data in test)   - Storage Memory Estimates (Cache, what kind of data in test)   - Storage Sconarios to not cover   - Fedures and Expectation - App layers scaling (microservices, service discovery)   - DB (RDBMS, NoSQL) - RDBMS (master-slave, Master, master, federarion, sharding Denormalization, SQL tuning)   - NosQL (KV< wide column, graph, document	Manageable		Scalability	5. Deep Dive	
Regulatory       Environmental       - scaling components         2. Estimations       - Availability, consistency and scale story of each component         Throughput       Latency       - Components         (QPS, read/write)       - Components       - Components         RW ration       Traffic Estimates - write(QPS, volume of data)       - LB (active passive, active active, L4, L7)         and read (QPS volume of data       - Reverse proxy       - App layers scaling ( microservices, service discovery)         Storage       Memory Estimates (Cache, what kind of data in       - App layers scaling ( microservices, service discovery)         RAM       SSD or disk       - RDBMS ( master-slave, Master, master, federarion, sharding Denormalization, SQL tuning)         Features and Expectation       - NoSQL ( KV< wide column, graph, document	Usability		Performance		
- Scaling components - Availability, consistency and scale story of each component - Availability, consistency and scale story of each component - Availability, consistency and scale story of each component - consistency and availability patterns - Components - DNS - CDN (push vs pull) - LB (active passive, active active, L4 , L7) - Reverse proxy - App layers scaling (microservices, service discovery) - DB (RDBMS, NoSQL) - RDBMS (master-slave, Master, master, federarion, sharding Denormalization, SQL tuning) - NoSQL (KV< wide column, graph, document - fast lookups - RAM (redis, memcache - AP - Cassandra, RIAK - CP - HBase, Mongo - Caches - client, CDN, server, DB, application, query, object	Regulatory		Environmental		
and read (QPS volumen of data and read (QPS volumen of data Storage Memory Estimates (Cache, what kind of data in Estimates it) RAM SSD or disk Features and Expectation Features and Expectation See case Scenarios to not cover who will use Scenarios to not cover who will use How many will use Jsage patterns Abp layers scaling (microservices, service discovery) - DB (RDBMS, NoSQL) - RDBMS (master-slave, Master, master, federarion, sharding Denormalization, SQL tuning) - NoSQL (KV< wide column, graph, document - fast lookups - RAM (redis, memcache - AP - Cassandra, RIAK - CP - HBase, Mongo - Caches - client, CDN, server, DB, applicaiton, query, object	2. Estimations Throughput Latency (QPS, read/write)		<ul> <li>Availability, consistency and scale story of each component</li> <li>consistency and availability patterns</li> <li>Components</li> </ul>		
Subarge Intentity Estimates (Cache, what kind of data in Estimates it)   Estimates it)   RAM SSD or disk    Features and Expectation  Features and Expectation  Subarge patterns  Subarge patterns  A Design Goals  Latency / throughput requests  Consistency vs Availability  Istimates (Cache, what kind of data in Features (Features	R/W ration			<ul> <li>- LB (active passive, active active, L4 , L7)</li> <li>- Reverse proxy</li> <li>- App layers scaling (microservices, service discovery)</li> <li>- DB (RDBMS, NoSQL)</li> </ul>	
RAM SSD or disk     Features and Expectation     Features and Expectation     Jse case   Scenarios to not cover   who will use   How many will use   Jsage patterns      B. Design Goals   Latency / throughput requests Consistency vs Availability	Storage Estimates				
Features and Expectation       - fast lookups         Jse case       Scenarios to not cover         who will use       How many will use         Jsage patterns       - AP - Cassandra, RIAK         Jsage patterns       - CP - HBase, Mongo         - Caches       - client, CDN, server, DB, applicaiton, query, object         Consistency vs Availability       - Client, CDN, server, DB, applicaiton, query, object	RAM SSD or disk				
Use caseScenarios to not cover- RAM (redis, memcachewho will useHow many will use- AP - Cassandra, RIAKUsage patterns- CP - HBase, MongoB. Design Goals- CachesLatency / throughput retro- client, CDN, server, DB, applicaiton, query, object	Features and Expectation				
- CP - HBase, Mongo - Caches - Client, CDN, server, DB, applicaiton, query, object - client, CDN, server, DB, applicaiton, query, object	Use case	Scenarios to not cover			
- Caches - Cient, CDN, server, DB, applicaiton, query, object - client, CDN, server, DB, applicaiton, query, object	who will use	How many will use		- AP - Cassandra, RIAK	
- Caches - Client, CDN, server, DB, applicaiton, query, object - client, CDN, server, DB, applicaiton, query, object - client, CDN, server, DB, applicaiton, query, object	Usage patterns			- CP - HBase, Mongo	
Latency / throughput requests Consistency vs Availability					
Consistency vs Availability	3. Design Goals			- client, CDN, server, DB, applic	caiton, query, object
	Latency / throughp	out requests			
weak / strong or eventual consistency	Consistency vs Av	ailability			
	- weak / strong or e	eventual consistenc	у		

- failover / replication availability

## By vinu76jsr

cheatography.com/vinu76jsr/

Published 15th September, 2022. Last updated 15th September, 2022. Page 1 of 1. Sponsored by CrosswordCheats.com Learn to solve cryptic crosswords! http://crosswordcheats.com