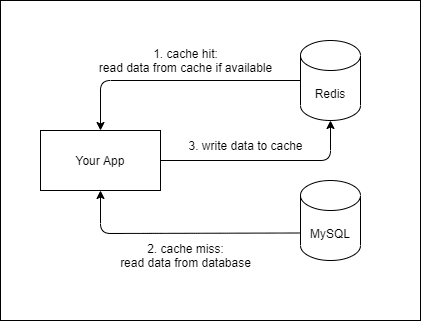
Cache Aside:

When you develop an app that uses database as storage, i.e. MySQL, you probably need to use a cache layer to help your application serve data more quickly, especially for serving data that's not changed too often. Whether you stored it in application memory or external cache server, i.e. Redis. This is called cache-aside pattern.

[](https://res.cloudinary.com/practicaldev/image/fetch/s--B1wqoEh1--/c_limit%2Cf_auto%2Cfl_progressive%2Cq_auto%2Cw_880/https:/dev-to-uploads.s3.amazonaws.com/i/waop2o50ozhvohmrofre.png)

**Read-through/write-through (rt/wt):** This is where the application treats cache as the main data store and reads data from it and writes data to it. the cache is responsible for reading and writing this data to the database, thereby relieving the application of this responsibility.

**Write through Cache:**  
 Write through is a storage method in which data is written into the cache and the corresponding main memory location at the same time. The cached data allows for fast retrieval on demand, while the same data in main memory ensures that nothing will get lost if a crash, power failure, or other system disruption occurs.

**Write- through cache:**  
Write-Through is a caching technique in which when the application writes data to the cache, the cache assumes that the application wants to update the data in the database as well. It then takes the job of writing the data to the database itself, which makes it a write-through cache.

The cache is able to perform this writing task by registering a write-through provider developed by you that contains all the persistence code specific to your database that would otherwise have existed in your application. The write-through provider is implemented as an interface that is provided by a cache such a Ncache. Once implemented, the provider is deployed on all the cache servers and your persistence code now resides on the cache.

Question 2:

Message queue:

A message queue is **a form of asynchronous service-to-service communication used in serverless and microservices architectures**. Messages are stored on the queue until they are processed and deleted. Each message is processed only once, by a single consumer.

Uses:

* Redundancy via Persistence. ...
* Traffic Spikes. ...
* Improve Web Application Page Load Times. ...
* Batching for Efficiency. ...
* Asynchronous Messaging. ...
* Decouple by Using Data Contracts. ...
* Transaction Ordering and Concurrency Challenges. ...
* Improve Scalability.

PuB-SUB QUEUE:

Publish–subscribe is a sibling of the message queue paradigm, and is typically one part of a larger message-oriented middleware system. Most messaging systems support both the pub/sub and message queue models in their API; e.g., Java Message Service (JMS).

This pattern provides greater network scalablity and a more dynamic network topology, with a resulting decreased flexibility to modify the publisher and the structure of the published data.

Uses:

Pub/Sub is used for **streaming analytics and data integration pipelines to ingest and distribute data**. It is equally effective as messaging-oriented middleware for service integration or as a queue to parallelize tasks

Project:

UBER-SYSTEM DESIGN:

2 sec – Websocket



[This Photo](http://pngimg.com/download/80140) by Unknown Author is licensed under [CC BY-NC](https://creativecommons.org/licenses/by-nc/3.0/)

[Rider info] [GPS coordinates]

Driver’s (1) Payment

Database MS Verifier &

Revenue Analyser

Load link to Client’s

Balancer Database

(1)- Encrypted

Payment

Gateway

Sharded DB’s in Server

* **Riders / Customers :**

**Total Users : 500 Million**

**Active Users : 50 Million**

**Single User : 25 Rides per month.**

* **Driver:**

**Total Users : 50 Million**

**Active Users : 5 Million**

**Single Driver shift : 6 Hours**

At Peak : 500K Requests per sec.

5 Million Active User

AVG Shifts – 6 Hours

Total Shifts 4

5M / 4 = 1250K Active Drivers at any instance

1250K / 2 = 625K requests per Sec.

At Peak hours: 1300K Requests per sec.

* Added payment gateway for transactions for Driver and to verify payments from Rider’s side.
* 128- bit SSL Encrypted gateway was used to make payments. If a payment is delayed or something unprecedented situations happens , we can add a Additional CDN database for temporary storage of Ride’s hash values and bank links like “Remember my card” option. ----[OPTIONAL]
* The client’s database was attached to the main database to ensure a easy & express pick up- or drop for the customer.
* One more Subrat sir, this is from a basic learner of System design Essentials, I hope this is correct and this is purely based on my Imaginary Techniques that is mainly rely to :--
* Customer Satisfaction
* Driver’ Quick pay
* Security of Databases. 😊😊😊😁