

[Open in app](#)

★ Member-only story

# What is SAGA Pattern in Microservice Architecture? Which Problem does it solve?

SAGA is an essential Micro service Pattern which solves the problem of maintaining data consistency in distributed system



Soma · Follow

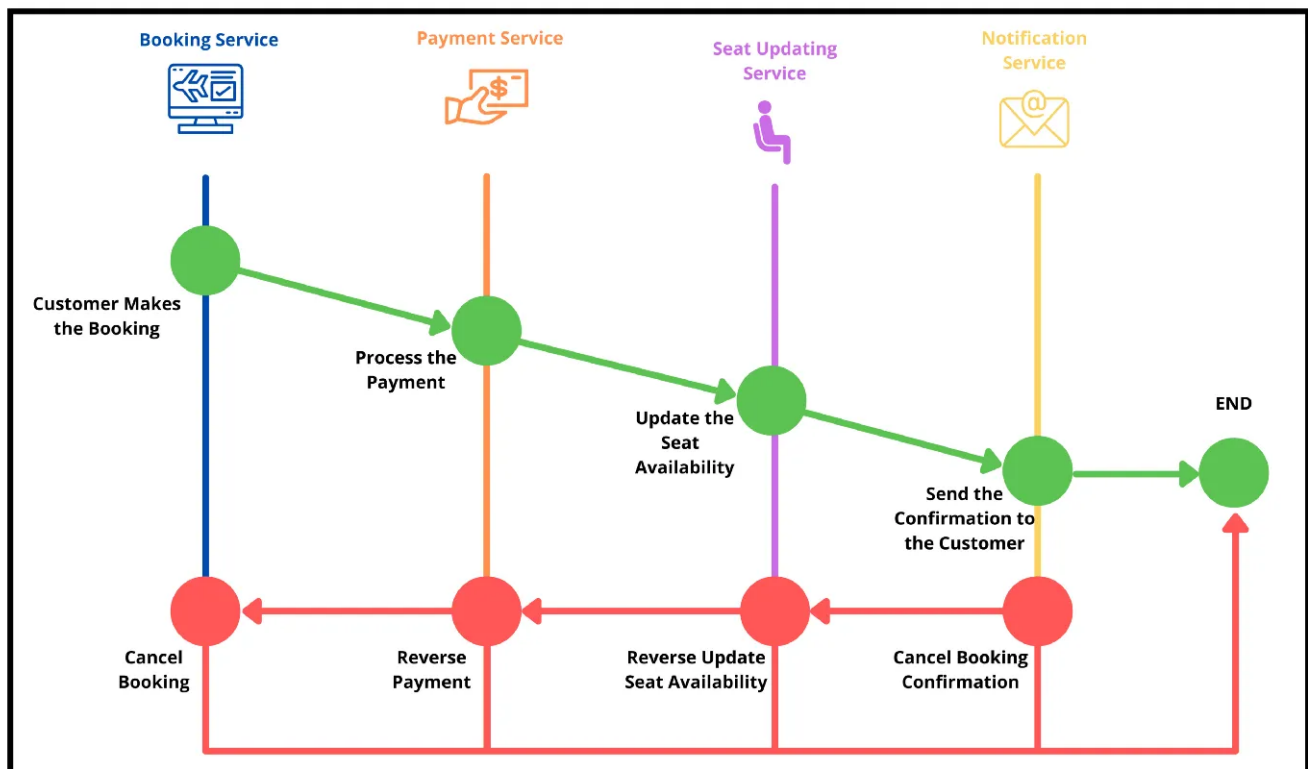
Published in Javarevisited

6 min read · Jan 31

Listen

Share

More



image\_source — Medium

Hello friends, if you are working a Java Microservice or preparing for a Java developer interview where Microservice skills are needed then you must prepare about SAGA Pattern. SAGA is an essential Microservice pattern which aims to solve

the problem of long lived transaction in Microservice architecture. It's also one of the popular [Microservice Interview Question](#) which often asked to experienced developers.

Since Microservice architecture breaks your application into multiple small applications, a single request is also broken down into multiple request and there is a chance that some part of requests are succeed and some parts are failed, in that case, maintaining data consistency is hard.

If are you dealing with real data like placing an order on Amazon then you must handle this scenario gracefully so that if payment is failed then inventory revert back to original state as well as order is not shipped.

In this article, I will explain What is SAGA Pattern? What it does, which problem it solves as well as pros and cons of SAGA Pattern in a Microservice architecture.

By the way, if you are preparing for Java developer interviews then you can also see my earlier articles like [25 Advanced Java questions](#), [25 Spring Framework questions](#), [20 SQL queries from Interviews](#), [50 Microservices questions](#), [60 Tree Data Structure Questions](#), [15 System Design Questions](#), and [35 Core Java Questions](#).

and, if you are not a Medium member then I highly recommend you to join Medium so that you can not only read these articles but also from many others like Google Engineers and Tech experts from FAANG companies. You can **join Medium [here](#)**

#### **Join Medium with my referral link - Soma**

Read this and every story from thousands of great writers on Medium). Read more from great writers on Medium like...

[medium.com](https://medium.com/javarevisited)

## **What is SAGA Design Pattern? What problem does it solve?**

The SAGA (or Saga) pattern is a Microservice design pattern for managing data consistency in distributed systems. It provides a way to handle long-lived

transactions that are composed of multiple steps, where each step is a separate database operation.

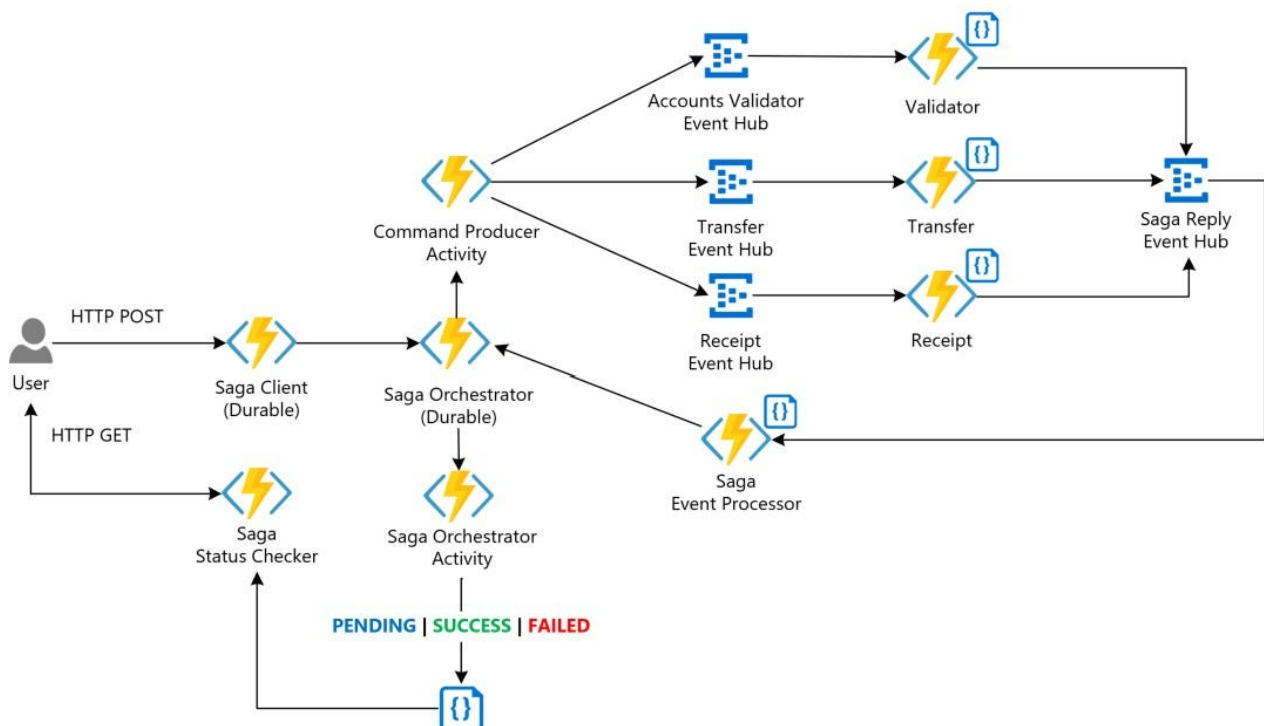
The main idea is to **capture all the steps of the transaction in a database**, so that in case of failure, the system can roll back the transaction to its initial state.

**The SAGA pattern solves the problem of maintaining data consistency in a distributed system**, where it is difficult to guarantee that all operations in a transaction are executed atomically, especially in case of failures.

One of the popular example of the SAGA pattern is an e-commerce transaction like placing an order of Amazon or FlipKart, where an order is placed, payment is deducted from the customer's account, and items are reserved in the inventory.

If any of these steps fail, the previous steps are rolled back to ensure consistency. For instance, if the payment fails, the reservation of items is cancelled. SAGA pattern solves the problem of maintaining consistency in a transaction involving multiple steps that may or may not succeed.

Here is another Microservice architecture diagram to demonstrate how SAGA Pattern works:



## Pros and Cons of SAGA Design Pattern in Microservices Architecture

Whenever we learn a pattern, we should learn its pros and cons as it help you to understand pattern better and also help you to decide when to use them in your application:

Here are some advantages and disadvantages of SAGA pattern in Microservice:

### Advantages:

Here are some advantages of using SAGA Design patter in Microservice architecture:

1. It's easy to implement complex transactions across multiple microservices.
2. Handles failures gracefully and ensures data consistency.
3. Increases system resilience and robustness.
4. Avoids data inconsistencies and lost updates.
5. Provides a clear and well-defined process for compensating transactions.

### Disadvantages

Here are some disadvantage of using SAGA Design patter in Microservice architecture:

1. It's hard to implement and maintain, its also difficult to monitor and debug
2. You will have overhead of storing and managing the state of sagas.
3. It also comes with Performance overhead due to the need to manage transactions across multiple microservices.
4. Your application will also suffer with Increased latency due to the need for multiple round trips between microservices.
5. There is no standardization in implementing sagas across different microservices. Would be better if frameworks like Spring Cloud or Quarks natively support this pattern in future.

## How to Implement SAGA pattern in a Microservice Architecture?

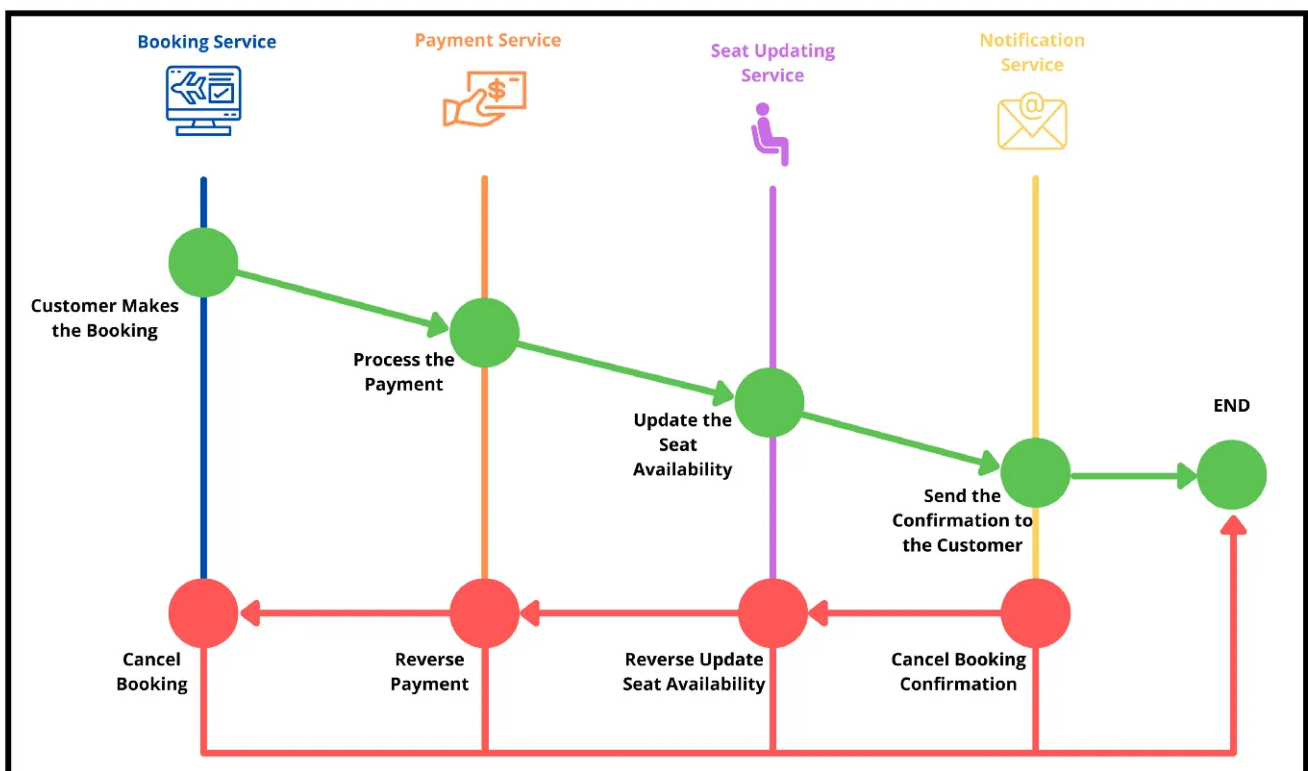
The SAGA pattern can be implemented in a Microservices architecture by breaking down a complex business transaction into multiple, smaller, independent steps or services.

Each step would communicate with its corresponding Microservice to complete a part of the transaction. If any step fails, the system will initiate a compensating transaction to undo the previous steps.

The coordination of these steps can be achieved by using a database, message queue, or a coordination service to store the state of the transaction and to trigger compensating transactions. This way, the system can ensure eventual consistency and handle failures gracefully.

If you are wondering whether any Java Microservice framework which can provide support for SAGA Pattern? Then unfortunately there is no specific Microservice framework that provides direct support for SAGA pattern.

Though, you can implement SAGA Pattern using libraries and frameworks such as Apache Camel or spring integration along with technologies like Apache Kafka, event sourcing, and message-driven architecture.



## Java and Spring Interview Preparation Material

Before any Java and Spring Developer interview, I always use to read the below resources

## Grokking the Java Interview

**Grokking the Java Interview: [click here](#)**

I have personally bought these books to speed up my preparation.

You can get your sample copy [here](#), check the content of it and go for it

**Grokking the Java Interview [Free Sample Copy]: [click here](#)**



*If you want to prepare for the Spring Boot interview you follow this consolidated ebook, it also contains microservice questions from spring boot interviews.*

Grokking the Spring Boot Interview

You can get your copy here — **[Grokking the Spring Boot Interview](#)**



Thank you for reading my article. That's all about **SAGA design Pattern in Microservice Architecture**. It's one of the complex but important pattern to learn and really important from interview point of view.

Even if you have not implement SAGA pattern in your project, its worth noting that because problem of managing distributed transaction and data consistency is a genuine problem and as an experienced developer you should know how to solve it in Microservice architecture.

*Btw, if you are not a Medium member then I highly recommend you to join Medium and read great stories from great authors from real field. You can [join Medium here](#)*

### Join Medium with my referral link - Soma

Read this and every story from thousands of great writers on Medium). Read more from great writers on Medium like...

[medium.com](https://medium.com)

### Other Java Interview Articles you may like to read

#### 21 Lambda and Stream Interview Questions for Experienced Java Programmers

My favorite Java interview questions from Lambda expression and Stream API for experienced Java developers

[medium.com](https://medium.com)

## Difference between @Controller and @RestController in Spring Boot and Spring MVC?

@Controller is used to declare common web controllers which can return HTTP response but @RestController is used to...

medium.com

## 20 SQL Query Interview Questions for Experienced Java Programmers and Data Scientists

SQL queries are commonly asked in Programming Job Interviews, if you are preparing for Tech interviews then practicing...

medium.com

Microservices

Microservice Architecture

Java

Programming

Software Development



Follow

## Written by Soma

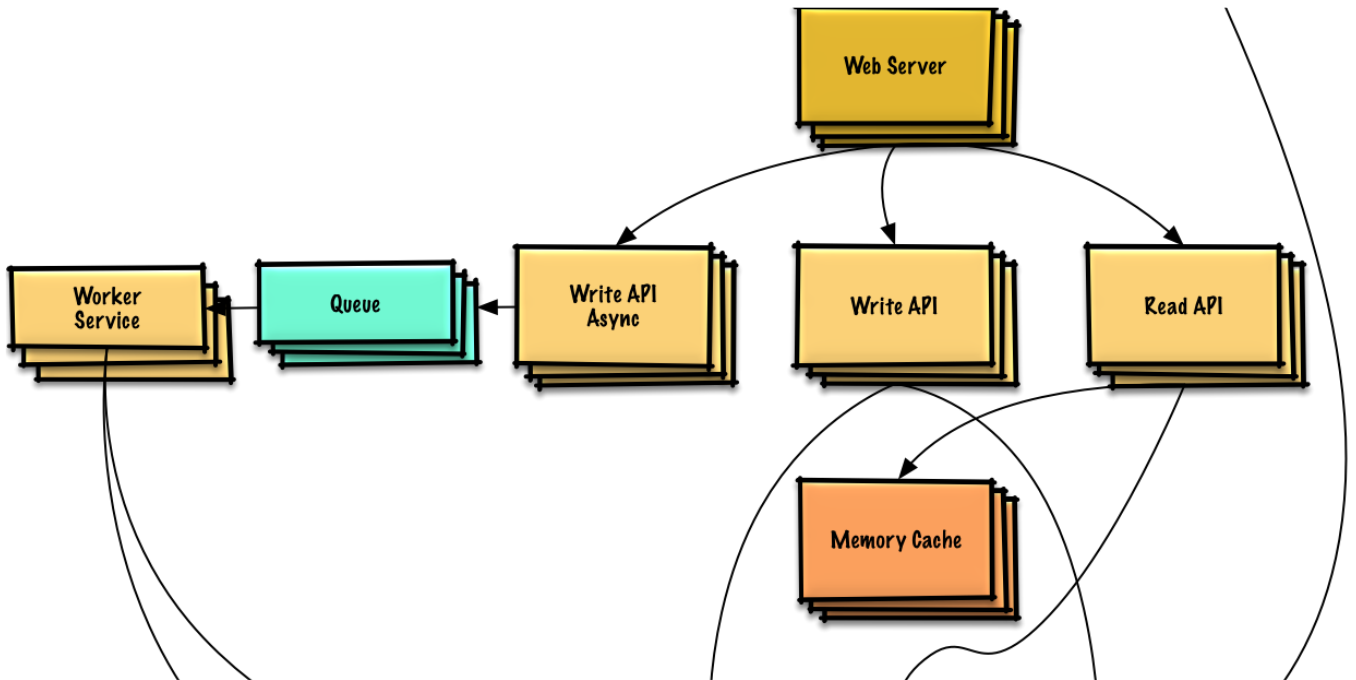
4.1K Followers · Editor for Javarevisited


Java and React developer, Join Medium (my favorite Java subscription) using my link 🙌

[https://medium.com/@somasharma\\_81597/membership](https://medium.com/@somasharma_81597/membership)

## More from Soma and Javarevisited





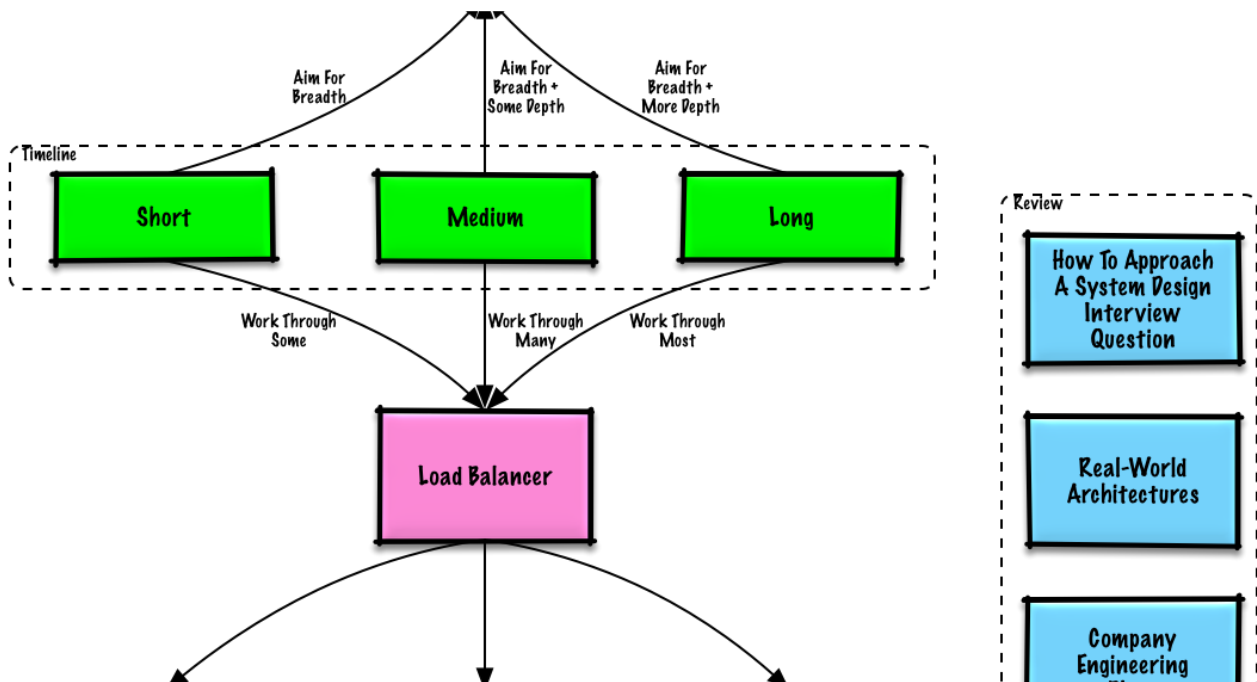
 Soma in Javarevisited


## 10 System Design Topics You Must Prepare for Tech Interviews

These are the 10 topics every developer must prepare for tech interviews to get ahead of 99% of candidates.

★ · 7 min read · Jul 2

 149



 javinpaul in Javarevisited

## 25 Software Design Interview Questions to Crack Any Programming and Technical Interviews

Frequently asked System Design Questions and links to useful resources

12 min read · Apr 27, 2020



1.3K



7



javinpaul in Javarevisited

## 10 Best Backend Frameworks for Web Development in 2023

These are the best backend development frameworks for web development in Java, Ruby, Python, JavaScript, PHP, Scala, and Golang in 2023

9 min read · Mar 9, 2022



398



5





Soma in Javarevisited

# 50+ Spring Boot Interview Questions and Answers for Java Programmers

These are 50+ Spring Boot Interview Questions and answers to Crack your Next Java Developer interview

🌟 · 19 min read · Apr 24



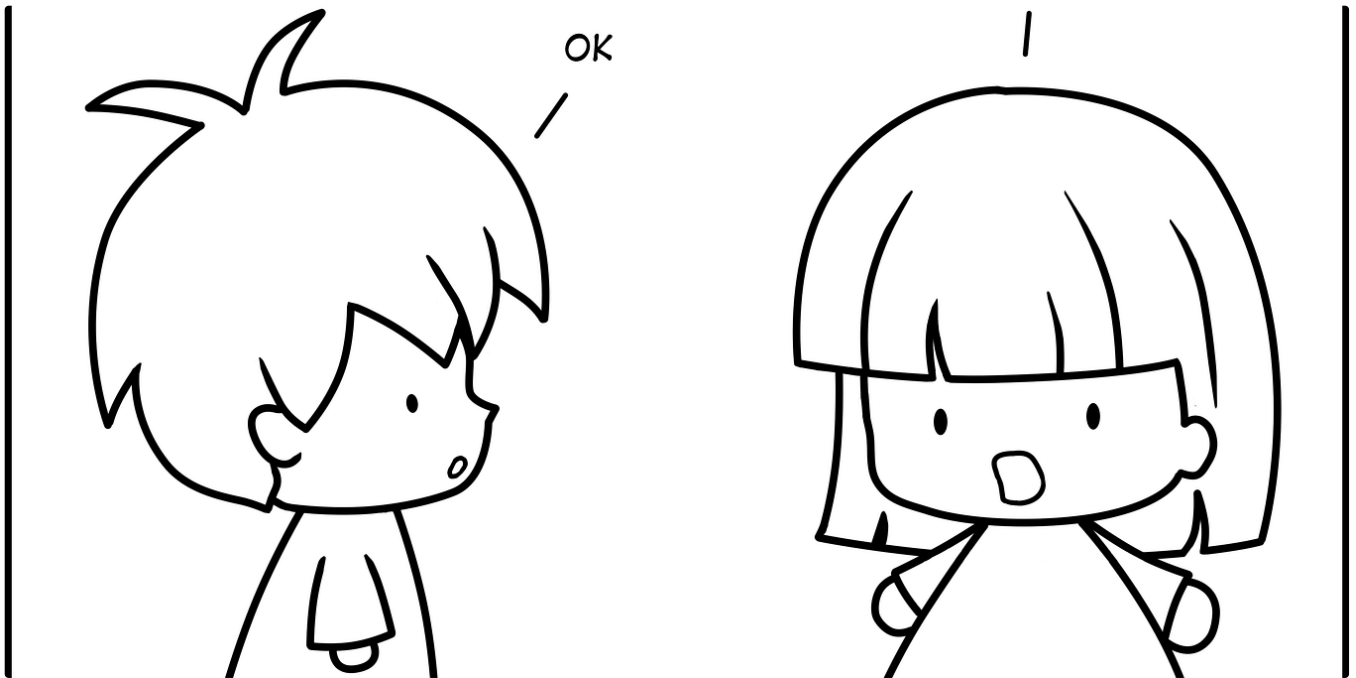
182




See all from Soma

See all from Javarevisited

## Recommended from Medium




 Julie Zhuo in The Year of the Looking Glass

## Average Manager vs. Great Manager

Explained in 10 sketches

2 min read · Aug 12, 2015

 20K    184



 Bubu Tripathy

## Best Practices for designing REST APIs

In this post, we'll discuss some of the best practices for designing RESTful APIs, including real-world examples.

15 min read · Mar 6



338



9



## Lists



### General Coding Knowledge

20 stories · 197 saves



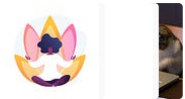
### It's never too late or early to start something

13 stories · 68 saves



### Coding & Development

11 stories · 101 saves



### Stories to Help You Grow as a Software Developer

19 stories · 265 saves



Shubhadeep Chattopadhyay

## Best practices for Clean Code

Clean code is a set of programming practices that emphasize the readability, maintainability, and simplicity of code. Writing clean code is...

7 min read · Mar 3



769



23



Amit Himani

## Distributed Transaction in Spring Boot Microservices

Distributed transactions in microservices refer to transactions that involve multiple microservices, each handling a part of the...



· 5 min read · Feb 17



49



3





 Daryl Goh

## Spring Boot: RequestEntity vs ResponseEntity | RequestBody vs ResponseBody

Spring Boot makes it easy to create stand-alone, production-grade Spring based Applications that you can “just run”.

🌟 · 5 min read · May 21

 40 



 Daniel Zielinski

# Senior Java Software Developer Interview Questions—part 1

## 1. Anemic Model vs Rich Model ?

★ · 3 min read · Feb 18



17



See more recommendations