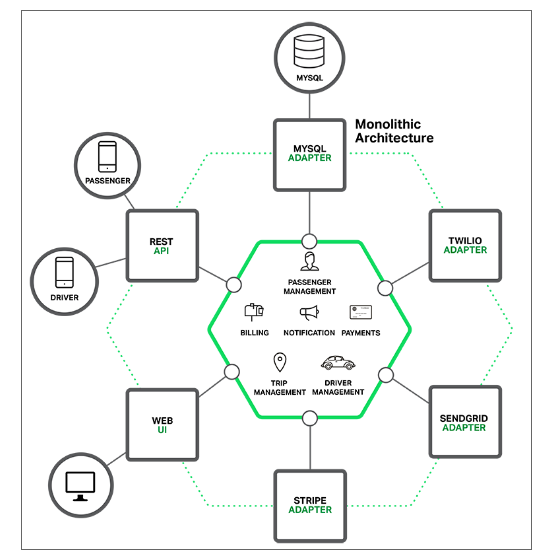
Monolitic Architecture



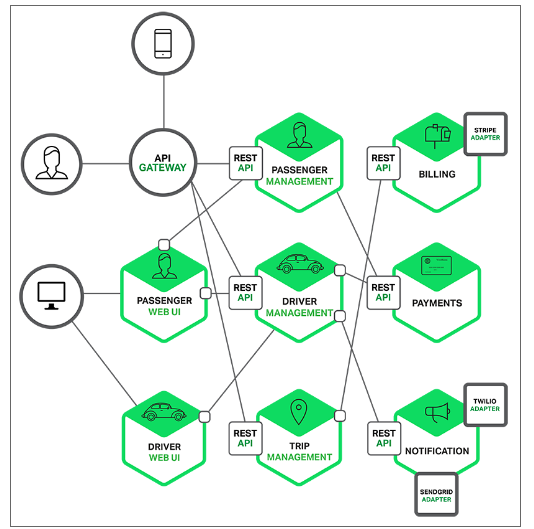
* Kelebihan

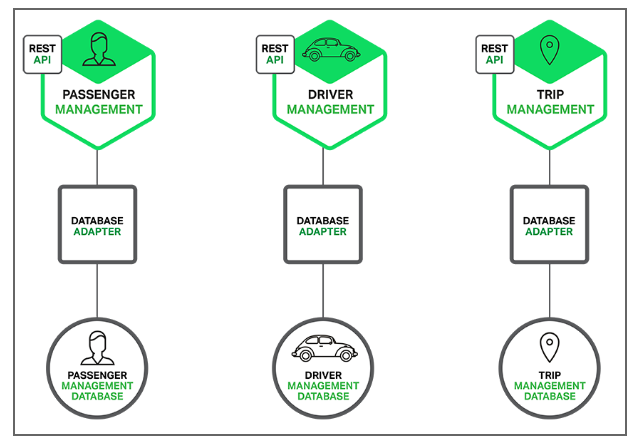
1. Mudah untuk ditest
2. Mudah di deploy

* Kekurangan

1. Relatif sulit dipahami ketika aplikasi sudah berkembang menjadi kompleks
2. Sulit untuk di scale
3. Arsitektur monolit sulit untuk mengadopsi framework baru.

Microservices Architecture





* Kelebihan

1. Mudah untuk dideploy
2. Menangani masalah kompleksitas
3. Pengembangan secara independen
4. Deploy secara independen

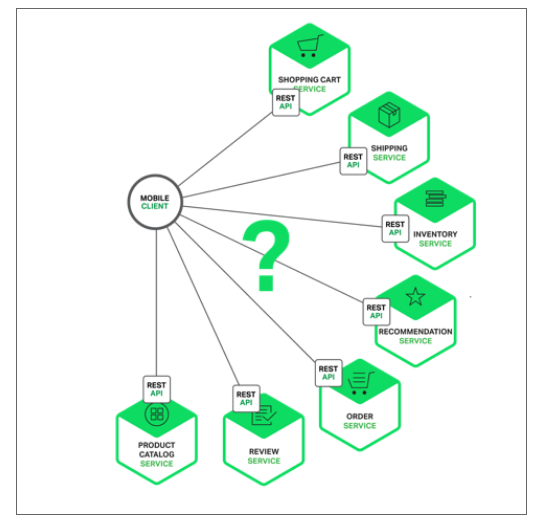
* Kekurangan

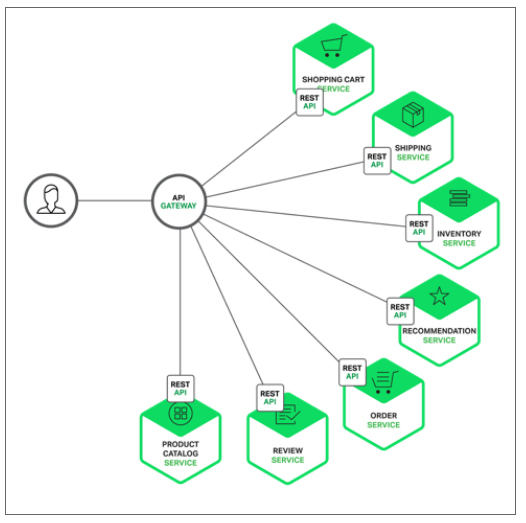
1. Microservices system distributed
2. Database partisi
3. Kompleksitas testing

API Gateway

Server yang merupakan titik masuk tunggal ke dalam system. API Gateway akan sering menangani permintaan dengan memanggil beberapa layanan mikro dan menggabungkan hasilnya.

* Load balancing
* Caching
* Access control
* Api metering
* Monitoring
* Request Routing
* Composition
* Protocol translation





* Kelebihan

1. Merangkum struktur internal aplikasi.

* Kekurangan

1. Dapat terjadi bottleneck
2. Developer harus memperbaharui API gateway untuk setiap endpoints.

Implementasi API Gateway

* Performance and Scalability, dapat menghandle jutaan request.

Supports asynchronous, nonblocking I/O, framework yang mendukung tersebut diantaranya adalah dari keluarga JVM yaitu NIO‑based frameworks such Netty, Vertx, Spring Reactor atau JBoss Undertow dan selain dari keluarga JVM adalah Node.js

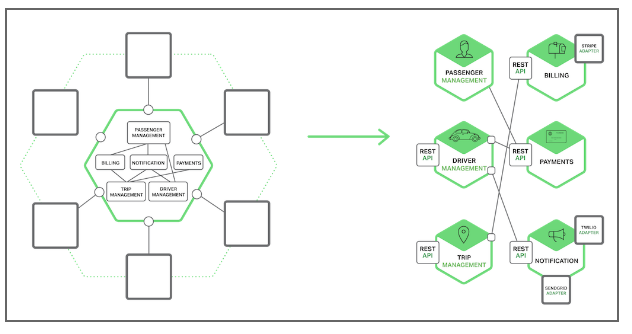
* Using a Reactive Programming Model, memanggil beberapa service (endpoints) dan menggabungkan hasilnya.
* Service Invocation, inter‑process communication
* Service Discovery, API Gateway perlu mengetahui lokasi (alamat IP dan port) dari setiap layanan mikro yang berkomunikasi dengannya

Server‑Side Discovery atau Client‑Side Discovery.

* Handling Partial Failures, Netflix Hystrix

Inter‑process communication

* Messaging
* RPC



Interaction Styles

* One‑to‑one

Setiap permintaan klien diproses oleh satu service instances.

* One‑to‑many

Setiap permintaan diproses oleh beberapa service instances.

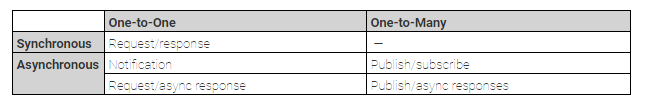
Dimensi kedua adalah apakah interaksi tersebut sinkron atau asinkron.

* Synchronous

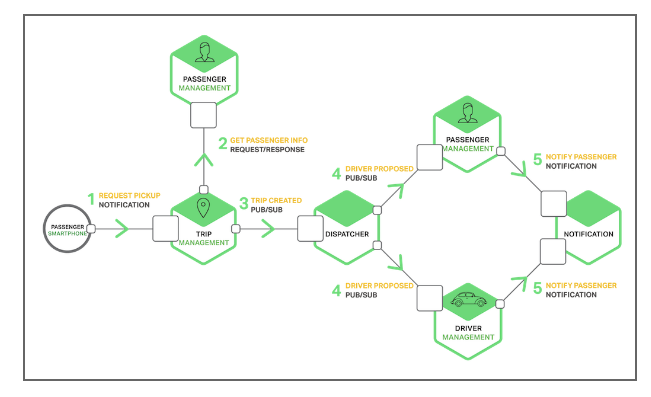
Klien mengharapkan respons yang tepat waktu dari layanan dan bahkan mungkin memblokir saat menunggu.

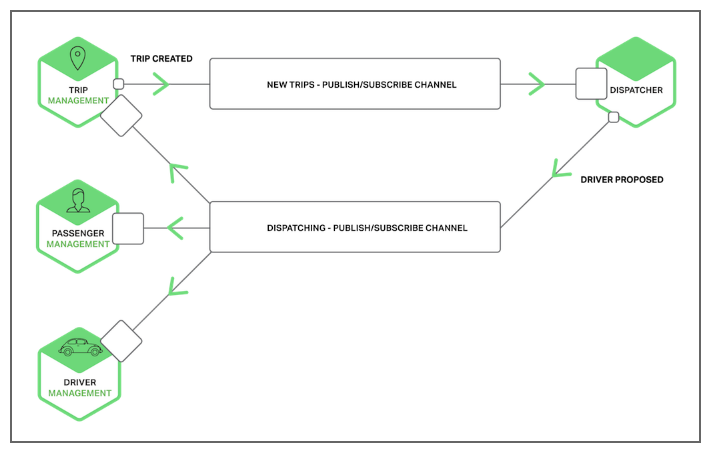
* Asynchronous

Klien tidak memblokir sambil menunggu respons, dan respons, jika ada, belum tentu segera dikirim.



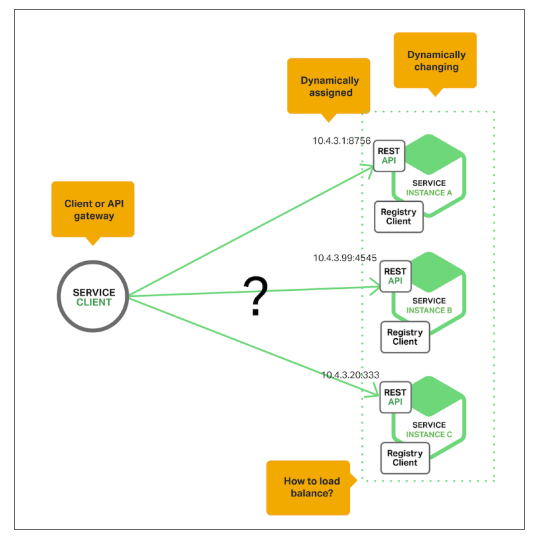
Pesan terdiri dari header (metadata seperti pengirim) dan badan pesan. Pesan dipertukarkan melalui channel. Sejumlah produsen dapat mengirim pesan ke channel

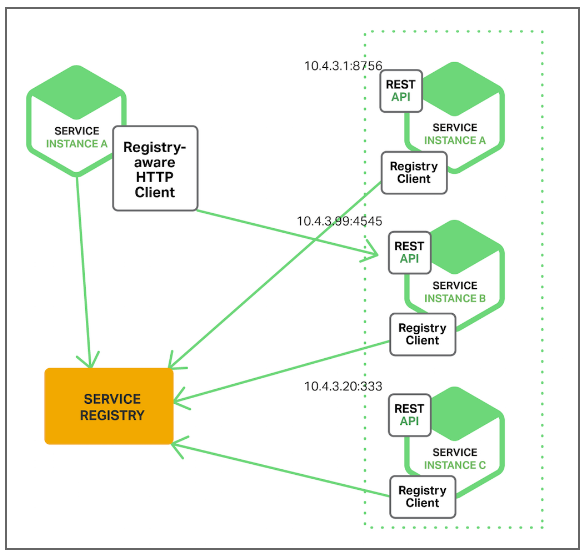




Handling Partial Failure

Service discovery



The Client‑Side Discovery Pattern

The Server‑Side Discovery Pattern

