DIGITAL TALENT SCHOLARSHIP

Thematic Academy
Tema pelatihan:

Pengenalan Microservice
(Pertemuan 14)

KOMINFO
BADAN PENELITIAN
DAN PENGEMBANGAN
SUMBER DAYA MANUSIA
Jl. Medan Merdeka Barat No. 9
Jakarta Pusat, 10110

Bagian 1

Monolitik dan Microservice

Apa Itu Monolitik

- Desain aplikasi yang memiliki arsitektur yang terpusat dan teknologi yang seragam, atau lebih dikenal dengan istilah Monolithic application architecture.
- Arsitektur aplikasi monolitik ini menggunakan kode sumber dan teknologi yang serupa untuk menjalankan semua tugas-tugasnya.

Apa Itu Monolitik

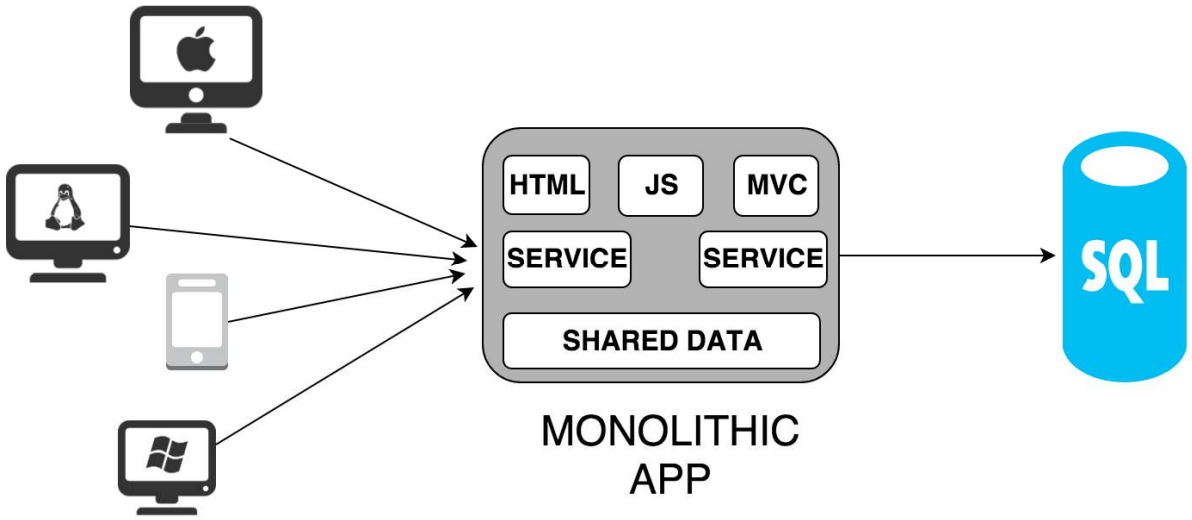
- WordPress merupakan contoh paling mudah untuk menggambarkan arsitektur aplikasi yang bersifat monolitik, dimana dalam satu aplikasi kita dapat memiliki frontend sekaligus backend.
- Semua fitur security, performance, manajemen konten, statistik, semuanya dibangun dengan menggunakan PHP dan database MySQL, dalam kode sumber yang sama.

Perkembangan Aplikasi

- Perkembangan berikutnya adalah dengan memisahkan aplikasi berdasarkan user role-nya.
- Ada frontend yaitu aplikasi yang akan diakses oleh pengakses blog (user) dan Backend yang digunakan oleh kontributor konten dan admin blog.
- Bisa juga memisahkan halaman reporting, member dan lainnya, dengan tujuan agar satu fungsi tidak akan mengganggu fungsi yang lainnya.

Arsitektur Monolitik

- Sebagai contoh yaitu aplikasi enterprise



Arsitektur Monolitik

- Aplikasi Enterprise dibangun dalam tiga bagian: database, client-side, dan server-side.
- Dimana Server-side akan menangani request HTTP kemudian menjalankan beberapa logika sesuai dengan domain, kemudian mengambil dan memperbarui data dari database, dan mengirim data tersebut ke sisi client-side.

Kekurangan Aplikasi Monolitik

- Ketika aplikasi menjadi besar (banyak yang akses) peforma akan menurun (kecuali punya banyak uang buat bayar server yang lebih bagus)
- Ketika akan mengubah teknologi pada aplikasi maka akan mengubah secara keseluruhan aplikasi.
- Jika terjadi error pada salah satu fungsi maka akan mempengaruhi keseluruhan aplikasi.

Kelebihan Aplikasi Monolitik

- Mudah dibangun
- Mudah di uji
- Mudah di deploy ke server atau cloud

Apa itu Microservice?

- Microservices berarti membagi aplikasi menjadi layanan yang lebih kecil dan saling terhubung tidak seperti aplikasi monolitik.
- Setiap microservice merupakan aplikasi kecil yang memiliki arsitektur heksagonal sendiri yang terdiri dari logika beserta berbagai adapternya (bahasa pemrograman, dan lain-lain).

Apa itu Microservice?

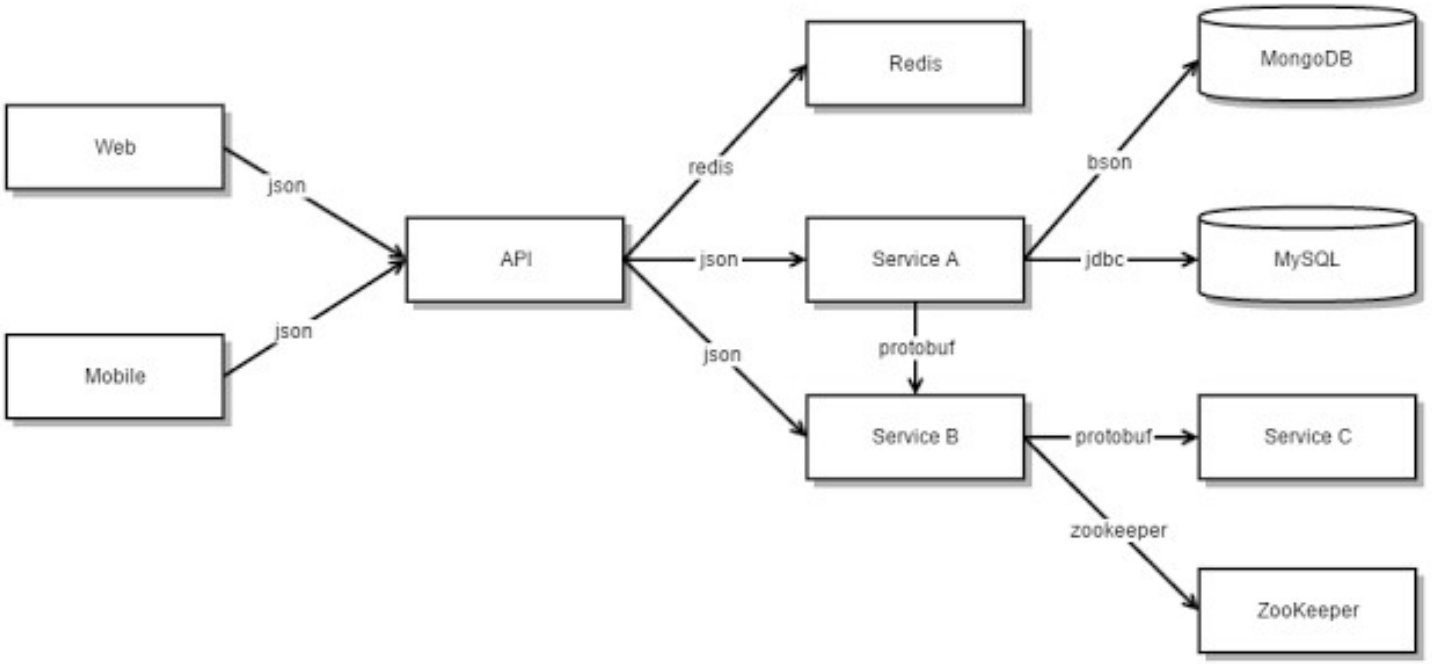
- Pola arsitektur Microservice secara signifikan mempengaruhi hubungan antara aplikasi dan database.
- Di satu sisi, pendekatan ini bertentangan dengan gagasan model data enterprise-wide.
- Namun, memiliki skema database per service sangat penting jika ingin mendapatkan keuntungan dari layanan microservice.

Apa itu Microservice?

- Jadi intinya microservice yaitu membagi service ke bagian yang lebih kecil dimana service-service tersebut saling berhungan satu sama lain.
- Selain itu, dalam setiap services yang dibuat bisa menggunakan teknologi yang berbeda-beda.

Microservice

- Contoh Arsitektur Microservices



Microservice

- Sedangkan untuk implementasi ke web, android, iOS dan lain-lain tidak bisa secara langsung.
- Dimana kita harus membuat terlebih dahulu yang namanya API Gateway.
- API Gateway memiliki tugas seperti load balancing, caching, access controll , API metering, dan monitoring.

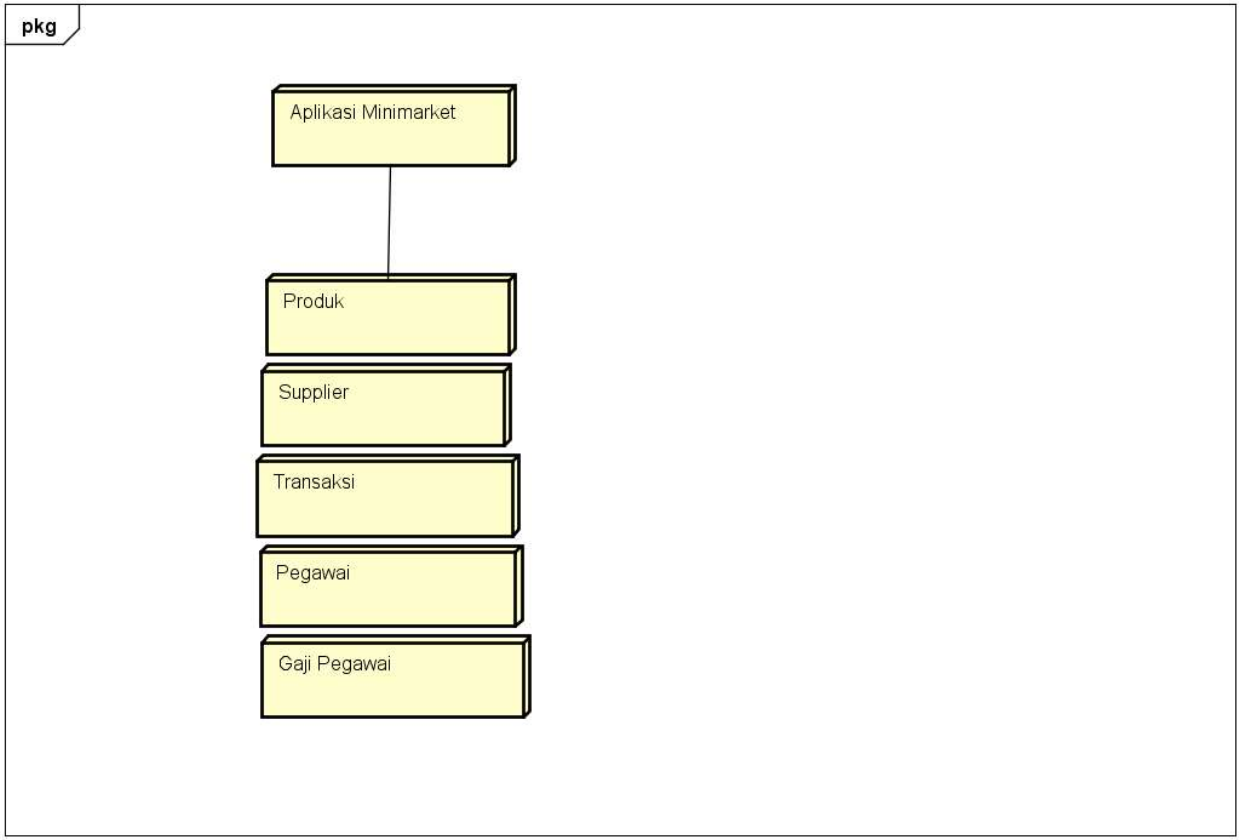
Kelebihan Microservice

- Aplikasi scalabale, secure dan reliable
- Setiap service berdiri sendiri
- Perawatannya lebih mudah
- Tidak ada hambatan dalam menggunakan teknologi baru
- Setiap tim developer dapat mengembangkan setiap services-nya tanpa mengganggu services yang lain

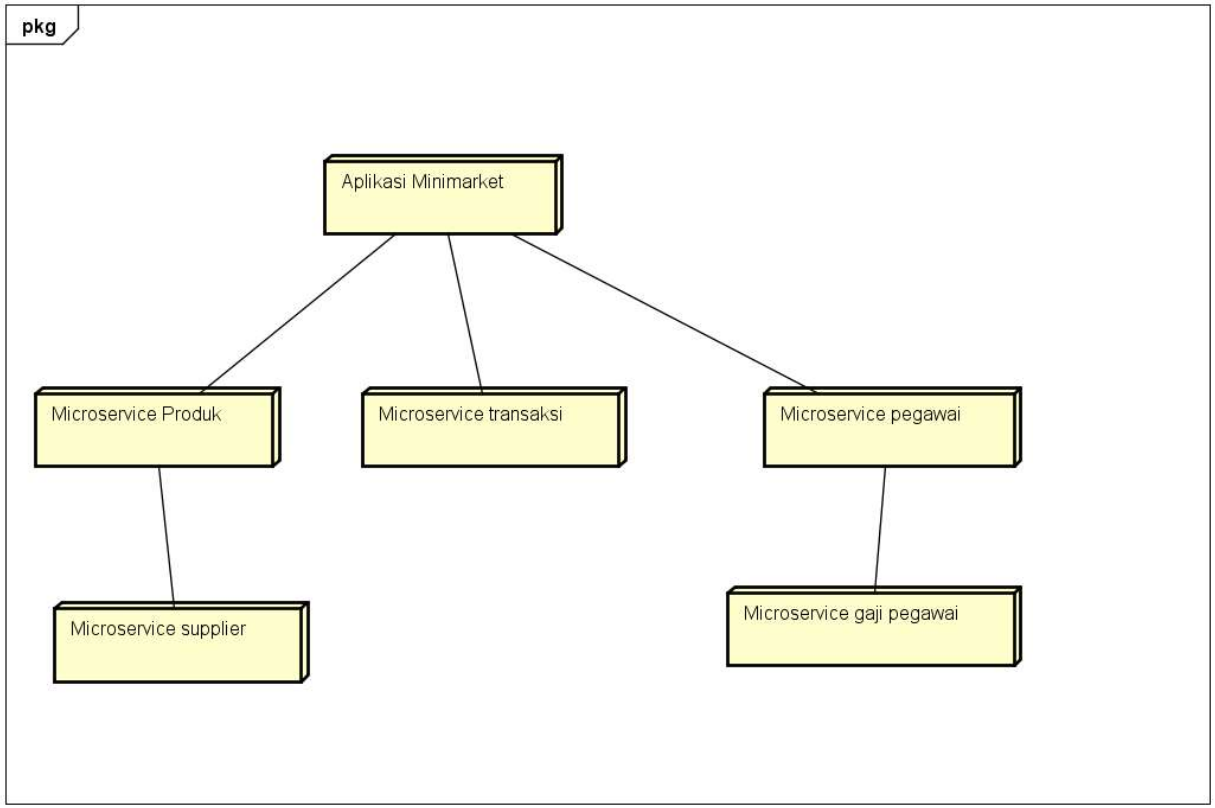
Kekurangan Microservice

- Ketika satu entity pada database berubah maka setiap entity yang sama di setiap database service harus diubah
- Untuk beberapa kasus , sulit untuk menerapkan perubahan services jadi perlu perancangan yang matang.
- Deployment yang kompleks, perlu konfigurasi untuk menjalankan setiap services karena memiliki runtime yang berbeda, tidak seperti aplikasi monolitik tinggal upload , deploy dan beres.
- Perlu automation yang tinggi dalam melakukan deployment.

Contoh Monolitik

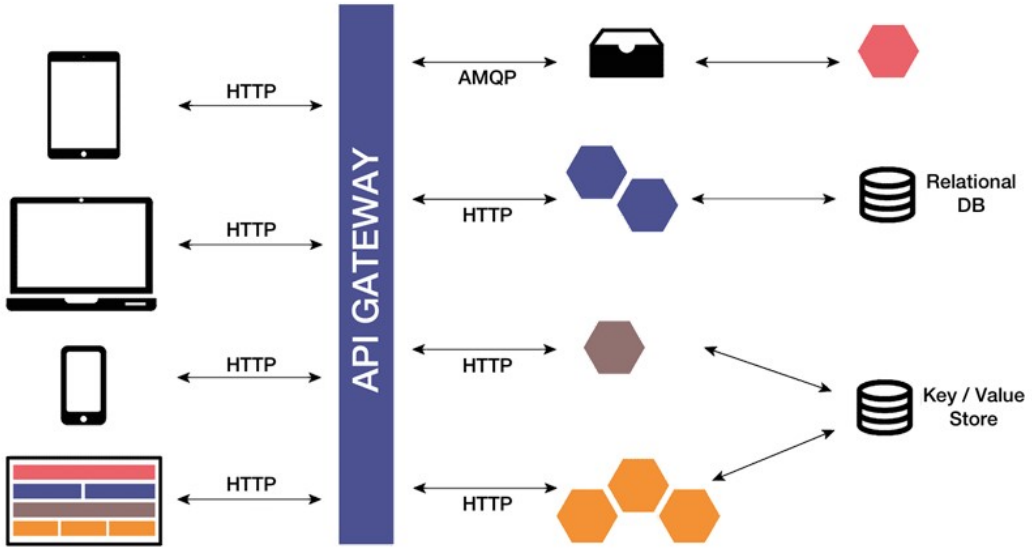


Contoh Microservice



Peran Microservice

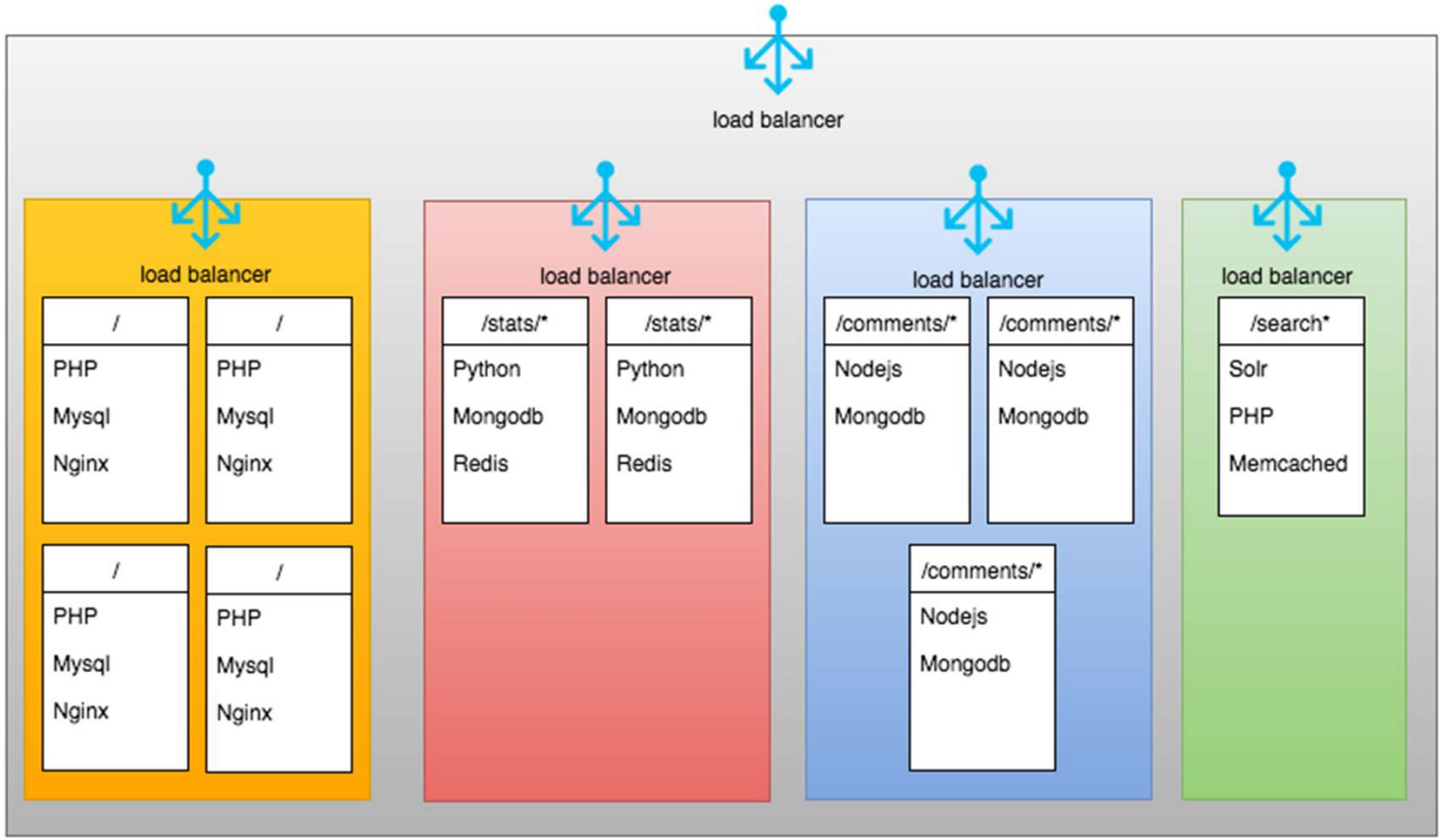
- Faktor-faktor kelebihan microservice adalah fokus kepada tugas-tugas kecil, terpisah dan sangat independen, layaknya blok-blok kecil.



Menggunakan Microservice

- Untuk implementasi microservices yang baik paling tidak kita harus menggunakan dua layer dari load balancer.
- Yang pertama berfungsi membagi traffic berdasarkan URL, dan yang kedua berguna untuk membagi traffic berdasarkan load server.
- Apabila digambarkan maka implementasi microservices akan memiliki desain infrastrukturnya akan seperti gambar selanjutnya.

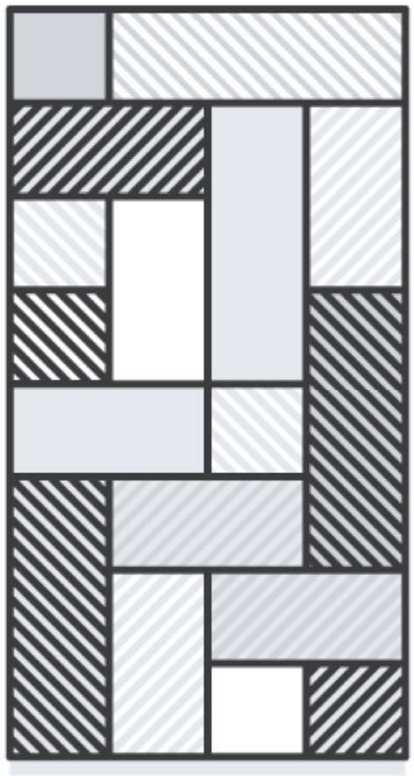
Menggunakan Microservice



Bagian 2

Evolution from Monolithics to Microservices

Monolitik



Challenges with monolithic software

DIGITAL TALENT SCHOLARSHIP

- Difficult to scale
- Architecture is hard to maintain and evolve
- Lack of agility
- Long Build/Test/Release Cycles (who broke the build?)
- New releases take months
- Lack of innovation
- Operations is a nightmare (module X is failing, who's the owner?)
- Long time to add new features
- Frustrated customers

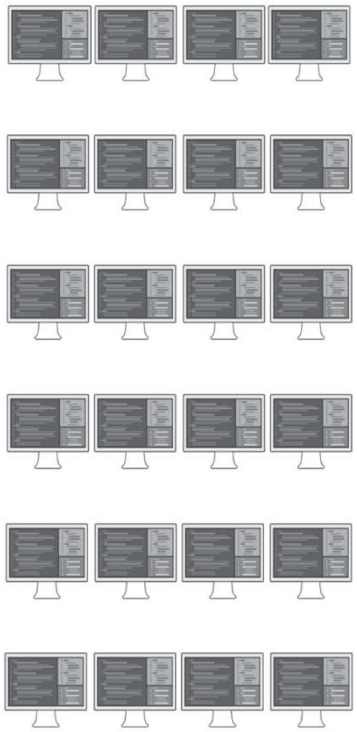
Bagian 3

How could those things happen?

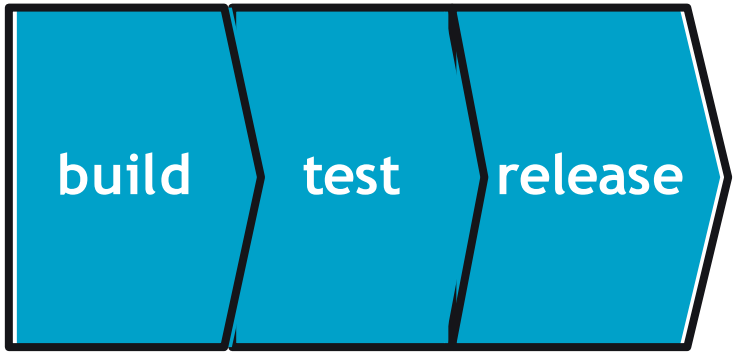
Monolithic Development Lifecycle

DIGITAL TALENT SCHOLARSHIP

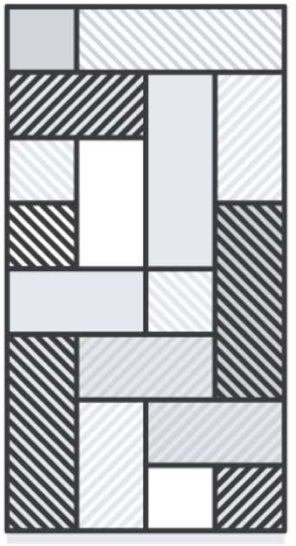
Developers



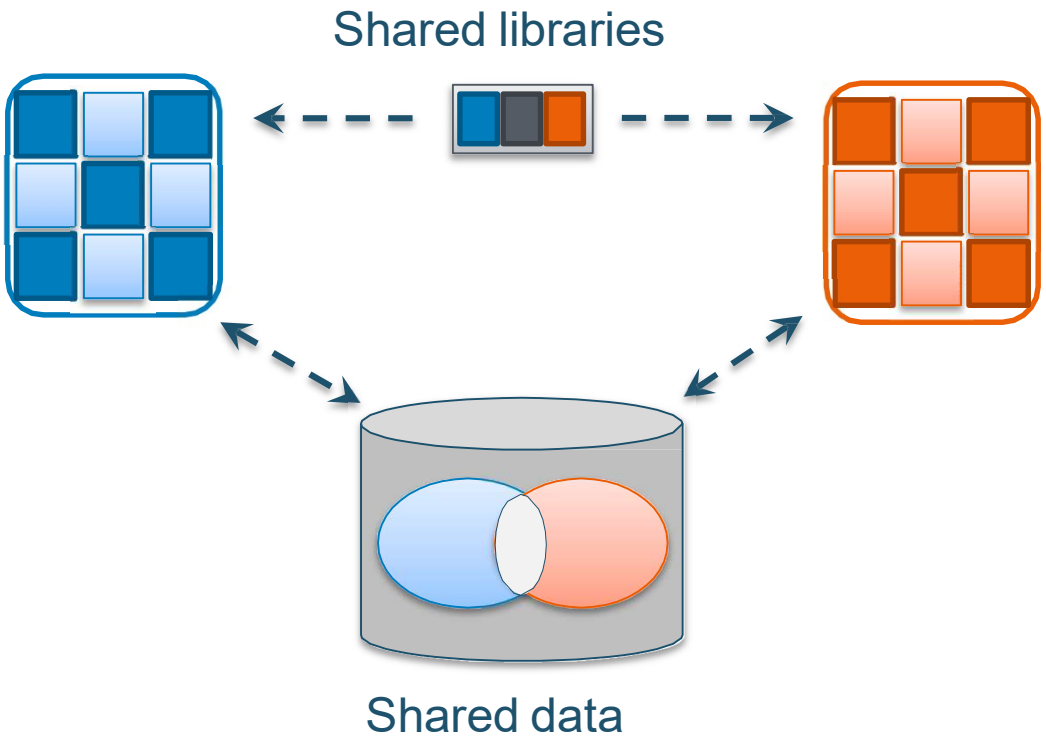
Delivery pipeline



App
(aka the "monolith")



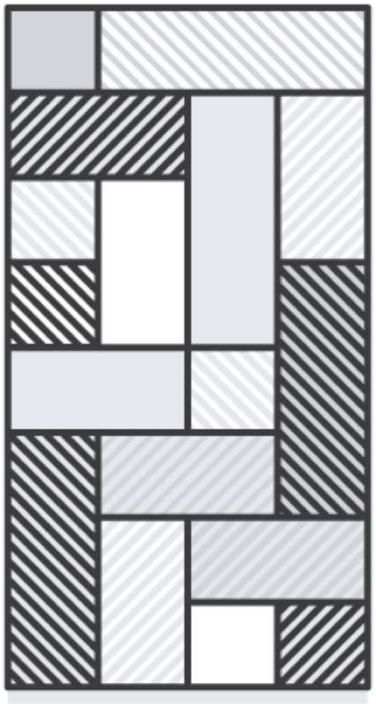
Too much software coupling

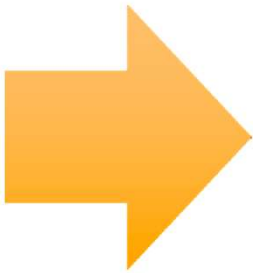


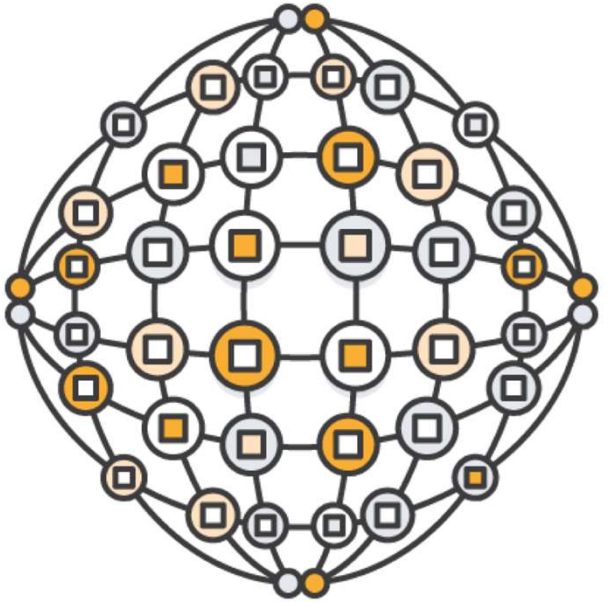
Bagian 4

So, what’s the solution then?

Microservice







Microservice

service-oriented architecture

composed of

loosely coupled elements

that have

bounded contexts

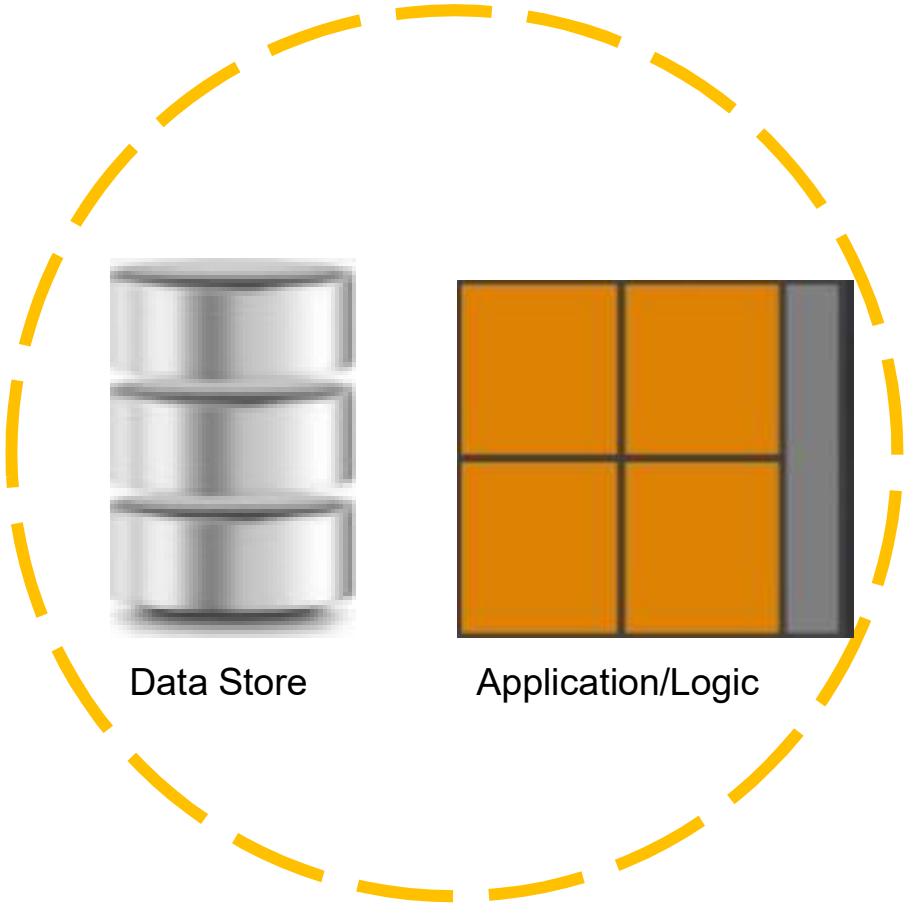
Services communicate with each other over the network

You can update the services independently; updating one service doesn't require changing any other services.

Self-contained; you can update the code without knowing anything about the internals of other microservices.

- Adrian Cockcroft (VP, Cloud Architecture Strategy at AWS)

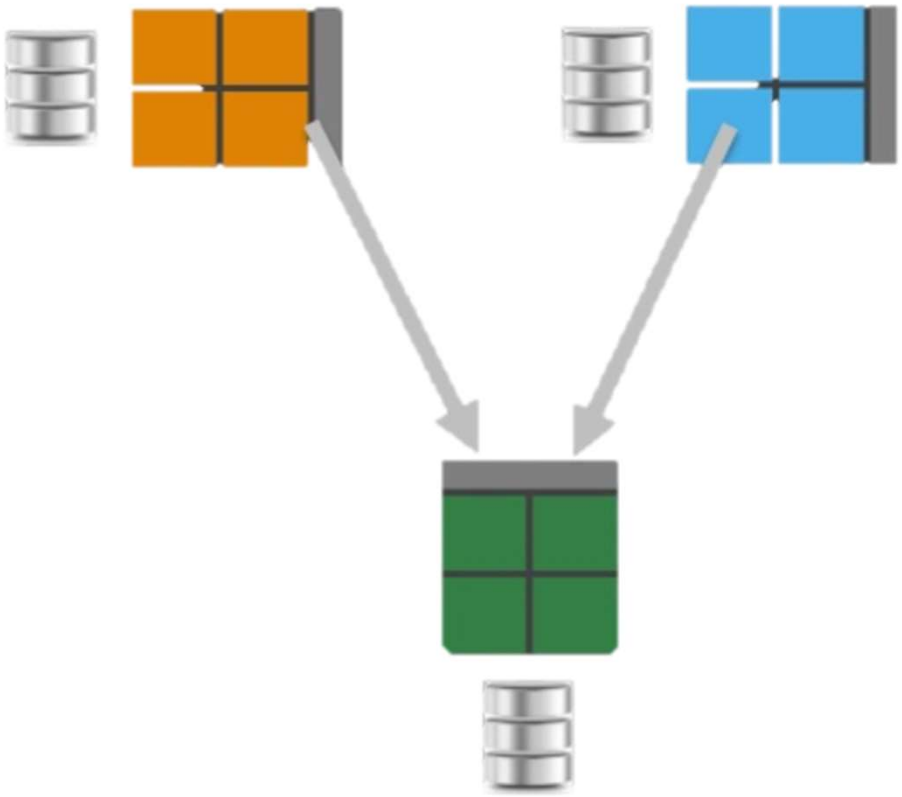
Anatomy of a Microservice



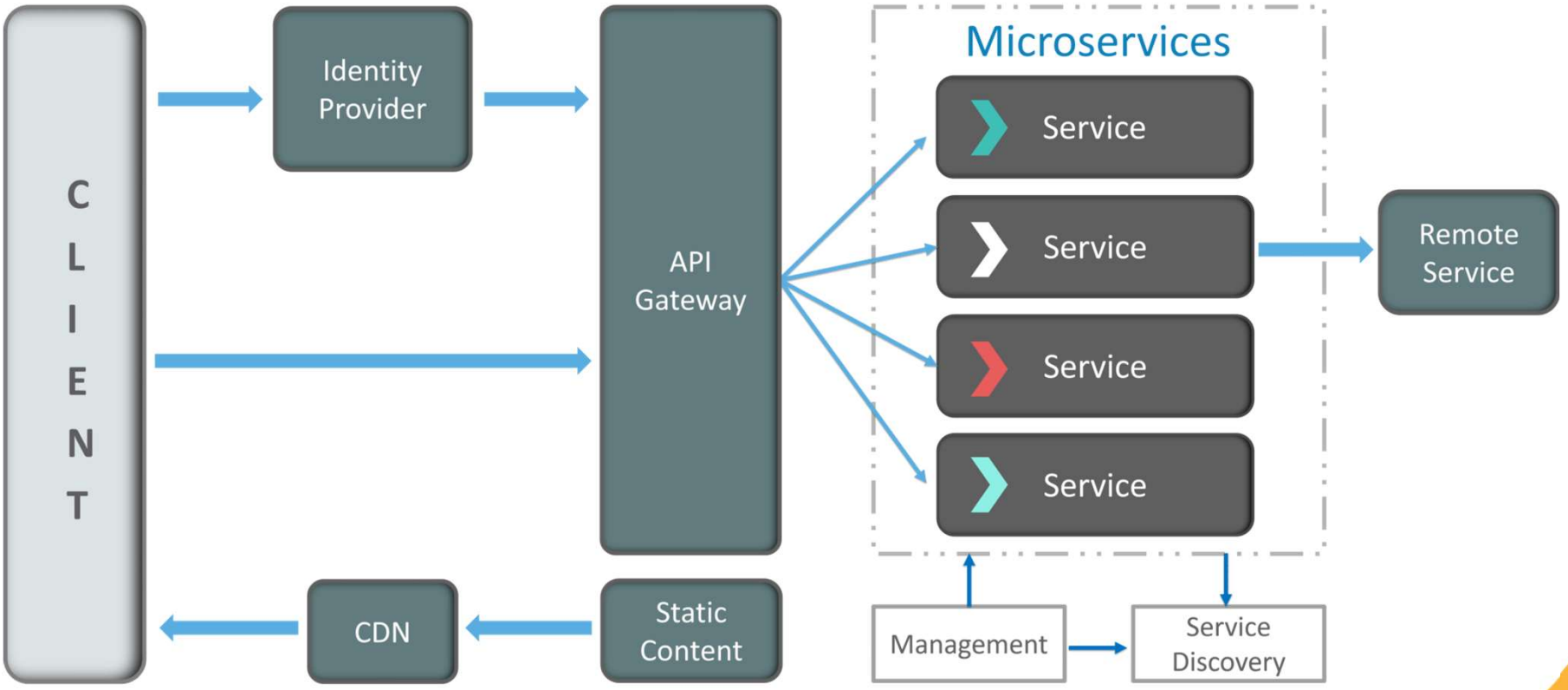
Public API

POST /restaurants
GET /restaurants

Avoid software coupling

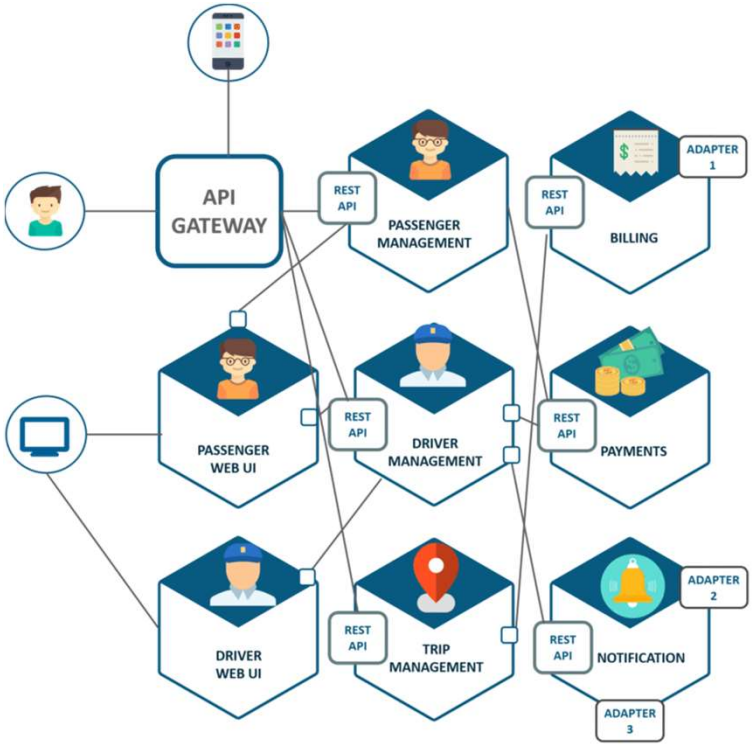


Ecosystem of Microservices



Example : Grab Transformation

Monolithic



Microservice

Bagian 5

Core Principals of Microservice

Principles of Microservices

DIGITAL TALENT SCHOLARSHIP

1. Rely only on the public API
- Hide your data
- Document your APIs
- Define a versioning strategy

2. Use the right tool for the job
- Polygot persistence (data layer)
- Polyglot frameworks (app layer)

3. Secure your services
- Defense-in-depth
- Authentication/authorization

4. Be a good citizen within the ecosystem
- Have SLAs
- Distributed monitoring, logging, tracing

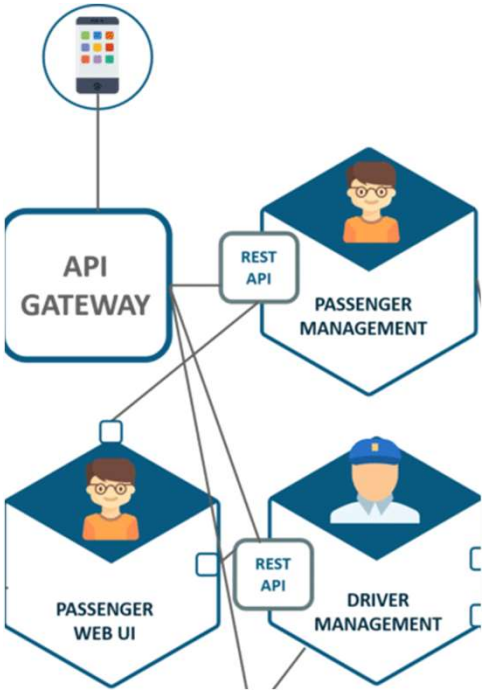
5. More than just technology transformation
- Embrace organizational change
- Favor small focused dev teams

6. Automate everything
- Adopt DevOps

Bagian 6

Case Study

Create your first Microservice using REST API



- **API Gateway** berfungsi sebagai gateway dari aplikasi android, ios dan web
- **Rest API** yang menghubungkan antara API Gateway dengan Passenger Management, Passenger Web UI dan Driver Management melalui REST API
- **Microservice** pada setiap layanan bekerja dan bertukar pesan dengan REST API
- **Database** pada setiap layanan dipisahkan masing - masing

1st step: Connect to Database

```go
package main

import (
    "database/sql"
    "fmt"
    "log"
    _ "github.com/go-sql-driver/mysql"
)

type Products struct {
    Sku          string
    Product_name string
    Stocks       int
}
```

```go
func main(){
    var products Products
    var arr_products []Products
    db, err := sql.Open("mysql","root:@tcp(127.0.0.1:3306)/go_coba")
    defer db.Close()

    if(err != nil) {
        log.Fatal(err)
    }
    rows, err := db.Query("Select sku,product_name,stocks from products ORDER BY sku DESC")
    if err!= nil {
        log.Print(err)
    }
    count:=0
    for rows.Next(){
        if err := rows.Scan(&products.Sku, &products.Product_name, &products.Stocks); err != nil {
            log.Fatal(err.Error())

        }else{
            arr_products = append(arr_products, products)
            fmt.Println(arr_products[count])
        }
        count++
    }
}
```

https://indraoctama.com/golang-for-beginner-part-1.2-print-data-as-json-api-20180609/

2nd step: Setup REST API

DIGITAL TALENT SCHOLARSHIP

```go
package main

import (
    "database/sql"
    "log"
    "encoding/json"
    _ "github.com/go-sql-driver/mysql"
    "net/http"
    "github.com/gorilla/mux"
)

type Products struct {
    Sku          string
    Product_name string
    Stocks       int
}
```

```go
type ResponseProduct struct {
    Status int
    Message string
    Data   []Products

}
```

```go
func main(){
    router := mux.NewRouter()
    router.HandleFunc("/getproducts",returnAllProducts).Methods("GET")
    http.Handle("/",router)
    log.Fatal(http.ListenAndServe(":12345",router))

}
```

```go
func returnAllProducts(w http.ResponseWriter, r *http.Request){
    var products Products // variable untuk memetakan data product yang terbagi menjadi 3 field
    var arr_products []Products // menampung variable products ke dalam bentuk slice
    var responseProd ResponseProduct //variable untuk menampung data arr_products yang nantinya aka
    db, err := sql.Open("mysql","root:@tcp(127.0.0.1:3306)/go_coba")
    defer db.Close()

    if(err != nil) {
        log.Fatal(err)
    }

    rows, err := db.Query("Select sku,product_name,stocks from products ORDER BY sku DESC")
    if err!= nil {
        log.Print(err)
    }
    //bentuk perulangan untuk me render data dari mySQL ke struct dan slice data products
    for rows.Next(){
        if err := rows.Scan(&products.Sku, &products.Product_name, &products.Stocks); err != nil {
            log.Fatal(err.Error())

        }else{
            arr_products = append(arr_products, products)
        }
    }

    responseProd.Status = 1 //mengisi valus status = 1 , dengan asumsi pasti success
    responseProd.Message = "Success"
    responseProd.Data = arr_products // mengisi komponen Data dengan data slice arr_products

    //mengubah data sstruct menjadi JSON
    json.NewEncoder(w).Encode(responseProd)
```

3rd step: Setup REST API


```
{"status":1,"message":"Success","Data":[{"sku":"SSI-D01466064-X3-BLA","product_name":"Salyara Plain Casual Big Blouse (XXXL,Black)","stocks":52},{"sku":"SSI-D01466013-XX-
BLA","product_name":"Salyara Plain Casual Big Blouse (XXL,Black)","stocks":77},{"sku":"SSI-D01401071-LL-RED","product_name":"Zeomila Zipper Casual Blouse
(L,Red)","stocks":76},{"sku":"SSI-D01401064-XL-RED","product_name":"Zeomila Zipper Casual Blouse (XL,Red)","stocks":44},{"sku":"SSI-D01401050-MM-
RED","product_name":"Zeomila Zipper Casual Blouse (M,Red)","stocks":73},{"sku":"SSI-D01326299-LL-NAV","product_name":"Siunfhi Ethnic Trump Blouse (L,Navy)","stocks":127},
{"sku":"SSI-D01326286-LL-KHA","product_name":"Siunfhi Ethnic Trump Blouse (L,Khaki)","stocks":210},{"sku":"SSI-D01326223-MM-KHA","product_name":"Siunfhi Ethnic Trump
Blouse (M,Khaki)","stocks":209},{"sku":"SSI-D01326205-MM-NAV","product_name":"Siunfhi Ethnic Trump Blouse (M,Navy)","stocks":143},{"sku":"SSI-D01326201-XL-
KHA","product_name":"Siunfhi Ethnic Trump Blouse (XL,Khaki)","stocks":186},{"sku":"SSI-D01322275-XL-WHI","product_name":"Thafqya Plain Raglan Blouse
(XL,White)","stocks":116},{"sku":"SSI-D01322234-LL-WHI","product_name":"Thafqya Plain Raglan Blouse (L,White)","stocks":105},{"sku":"SSI-D01220388-MM-
SAL","product_name":"Devibav Plain Trump Blouse (M,Salem)","stocks":216},{"sku":"SSI-D01220357-SS-YEL","product_name":"Devibav Plain Trump Blouse
(S,Yellow)","stocks":74},{"sku":"SSI-D01220355-XX-YEL","product_name":"Devibav Plain Trump Blouse (XXL,Yellow)","stocks":140},{"sku":"SSI-D01220349-LL-
YEL","product_name":"Devibav Plain Trump Blouse (L,Yellow)","stocks":101},{"sku":"SSI-D01220346-LL-SAL","product_name":"Devibav Plain Trump Blouse
(L,Salem)","stocks":151},{"sku":"SSI-D01220338-XX-SAL","product_name":"Devibav Plain Trump Blouse (XXL,Salem)","stocks":65},{"sku":"SSI-D01220334-XL-
YEL","product_name":"Devibav Plain Trump Blouse (XL,Yellow)","stocks":110},{"sku":"SSI-D01220322-MM-YEL","product_name":"Devibav Plain Trump Blouse
(M,Yellow)","stocks":121},{"sku":"SSI-D01220307-XL-SAL","product_name":"Devibav Plain Trump Blouse (XL,Salem)","stocks":182},{"sku":"SSI-D01037822-XX-
BLA","product_name":"Dellaya Plain Loose Big Blouse (XXL,Black)","stocks":8},{"sku":"SSI-D01037812-X3-BLA","product_name":"Dellaya Plain Loose Big Blouse
(XXXL,Black)","stocks":54},{"sku":"SSI-D01037807-X3-BWH","product_name":"Dellaya Plain Loose Big Blouse (XXXL,Broken White)","stocks":74},{"sku":"SSI-D00864661-MM-
NAV","product_name":"Deklia Plain Casual Blouse (M,Navy)","stocks":13},{"sku":"SSI-D00864652-SS-NAV","product_name":"Deklia Plain Casual Blouse (S,Navy)","stocks":2},
{"sku":"SSI-D00864614-XL-NAV","product_name":"Deklia Plain Casual Blouse (XL,Navy)","stocks":97},{"sku":"SSI-D00864612-LL-NAV","product_name":"Deklia Plain Casual Blouse
(L,Navy)","stocks":8},{"sku":"SSI-D00791091-XL-BWH","product_name":"Zalekia Plain Casual Blouse (XL,Broken White)","stocks":137},{"sku":"SSI-D00791077-MM-
BWH","product_name":"Zalekia Plain Casual Blouse (M,Broken White)","stocks":138},{"sku":"SSI-D00791015-LL-BWH","product_name":"Zalekia Plain Casual Blouse (L,Broken
White)","stocks":154},{"sku":"ffffff-ccc-ikik","product_name":"Zalekia Plain Casual Jeans (L,Broken White)","stocks":35}]}
```

3rd step: Get Service

DIGITAL TALENT SCHOLARSHIP

```go
package main

import (
    "encoding/json"
    "fmt"
    "io/ioutil"
    "log"
    "net/http"
    "time"
)

type Products struct {
    Status int
    Message string
    Data   []Products
}
```

```go
func main() {

    url := "http://localhost:12345/getproducts"

    spaceClient := http.Client{
        Timeout: time.Second * 2, // Timeout after 2 seconds
    }

    req, err := http.NewRequest(http.MethodGet, url, nil)
    if err != nil {
        log.Fatal(err)
    }

    req.Header.Set("User-Agent", "spacecount-tutorial")

    res, getErr := spaceClient.Do(req)
    if getErr != nil {
        log.Fatal(getErr)
    }

    if res.Body != nil {
        defer res.Body.Close()
    }
```

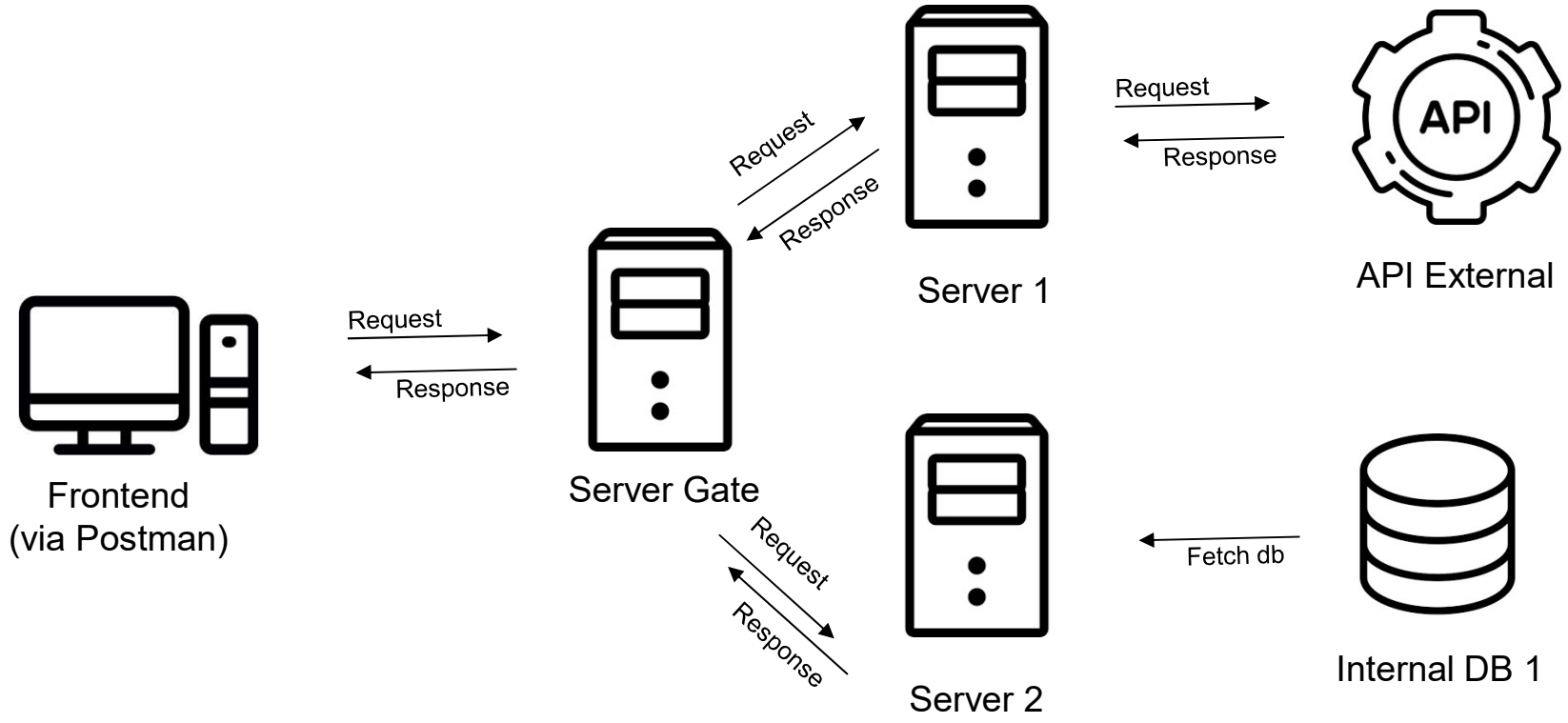
```go
    body, readErr := ioutil.ReadAll(res.Body)
    if readErr != nil {
        log.Fatal(readErr)
    }

    prod1 := Products{}
    jsonErr := json.Unmarshal(body, &prod1)
    if jsonErr != nil {
        log.Fatal(jsonErr)
    }

    fmt.Println(prod1.Status)
    fmt.Println(prod1.Message)
    fmt.Println(prod1.Data)
}
```

3rd step: Get Service

```
Octavianos-MBP:rest octavianopratama$ go run grab_json.go
1
Success
[{0  []} {0  []} {0  []} {0  []} {0  []} {0  []} {0  []} {0  []} {
 {0  []} {0  []} {0  []} {0  []} {0  []} {0  []} {0  []} {0  []} {
 {0  []} {0  []} {0  []} {0  []} {0  []} {0  []} {0  []} {0  []} {
 {0  []} {0  []}]
```

Untuk data masih berisi null, anda dapat menggunakan Decode JSON kembali untuk mendapatkan Data

Case Study (1 Jam 30 Menit)



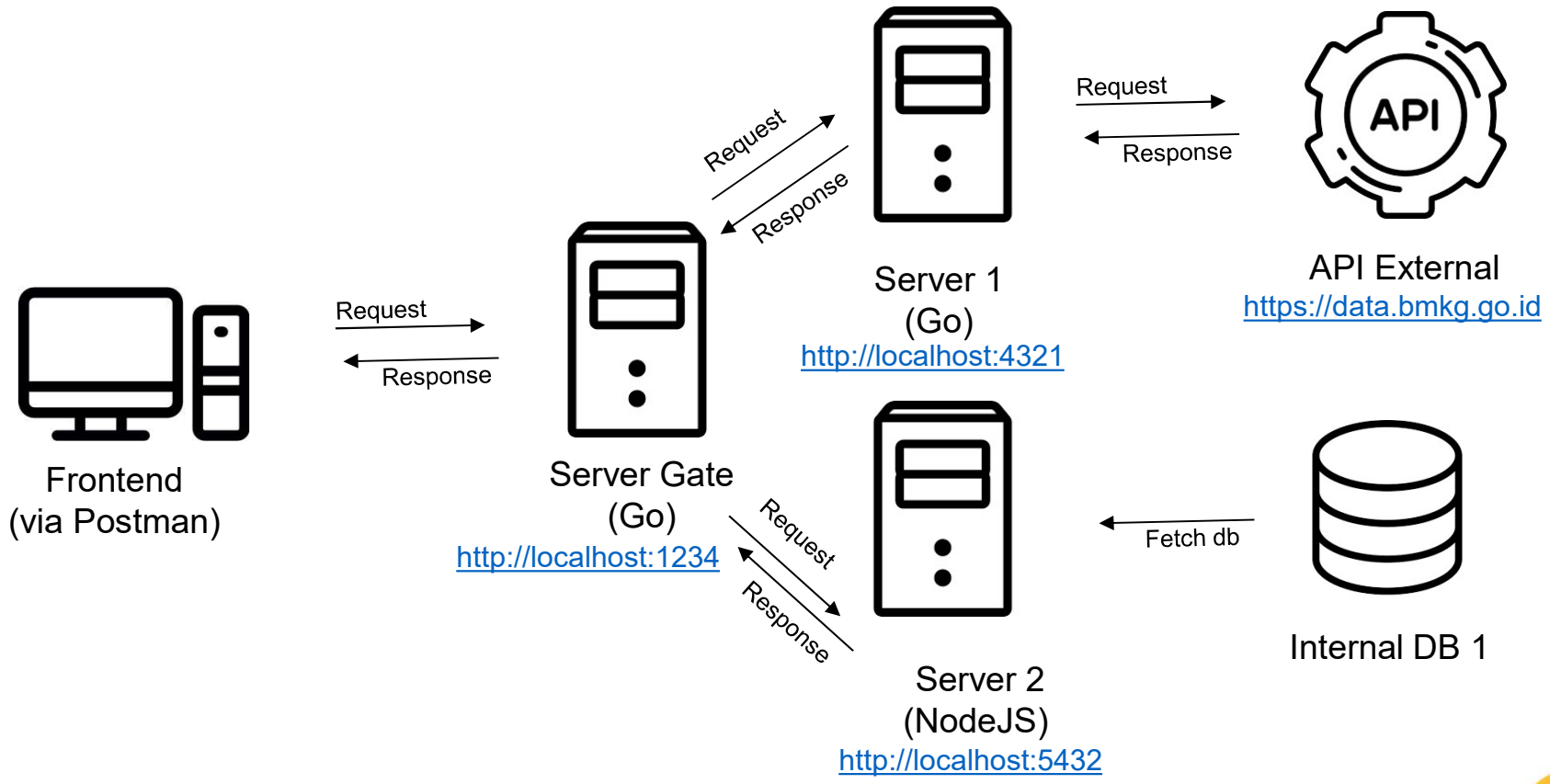
Case Study

- **Server 1** dengan kasus mengambil data dari suatu API contoh anda dapat mengambil data dari data.bmkg.go.id
- **Server 2** dengan kasus mengambil data dari internal database
- **Server Gate** menghubungkan Request dari Frontend, menuju ke Server 1 dan Server 2
- **Frontend** melalui aplikasi Postman mengujicoba API Server Gate, misalkan request service A, maka data akan diambil pada server 1, sementara ketika request service B, maka data akan diambil pada server 2

Bagian 7

Latihan Microservice

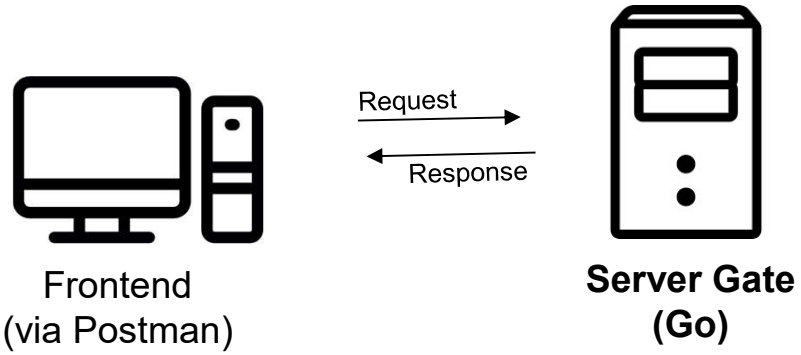
Arsitektur Microservice



Penjelasan Detail

- Terdapat 3 Server: Gate (Golang), Server 1 (Golang) dan Server 2 (NodeJS). Masing – masing memiliki tugas:
  - Gate: Sebagai gerbang server yang menghubungkan antara Frontend dan Backend dan AUTH
  - Server1 : Sebagai server yang menghubungkan dengan API External
  - Server2: Sebagai server yang menghubungkan dengan database internal
- **Gate,** menerima input dari Frontend, untuk kemudian langkah awal adalah menerima username dan password untuk divalidasi. Jika valid maka session akan aktif dan Frontend dapat request data dari server 1 dan server 2
- **Server1,** menerima request dari Gate untuk mengakses external API seperti dari https://api.bmkg.go.id untuk kemudian data nya akan dikirimkan berupa response ke Gate guna diteruskan ke Frontend
- **Server2,** menerima request dari Gate untuk mengakses internal database yang berisi data produk untuk kemudian data nya akan dikirimkan berupa response ke Gate guna diteruskan ke Frontend

Arsitektur Server Gate (30 menit)



Fitur dalam Server:

1. Authentication JWT
2. Diakses oleh Frontend melalui http://localhost:1234
3. Daftar Service sbb:
   - http://localhost:1234/auth
   - http://localhost:1234/getProduk
   - http://localhost:1234/getCuaca
   - dan service lainnya yang dapat anda tambah

Arsitektur Server Gate (30 menit)

1. https://www.nexmo.com/blog/2020/03/13/using-jwt-for-authentication-in-a-golang-application-dr
2. Download code https://github.com/victorsteven/jwt-best-practices
3. Perbaiki error main.go untuk perubahan dari int64 →uint64
4. Test JWT dengan postman
    1. Post localhost:8080/login `{ "username":"username", "password":"password"}`

5. Modifikasi / replikasi code **todo** untuk service gateway

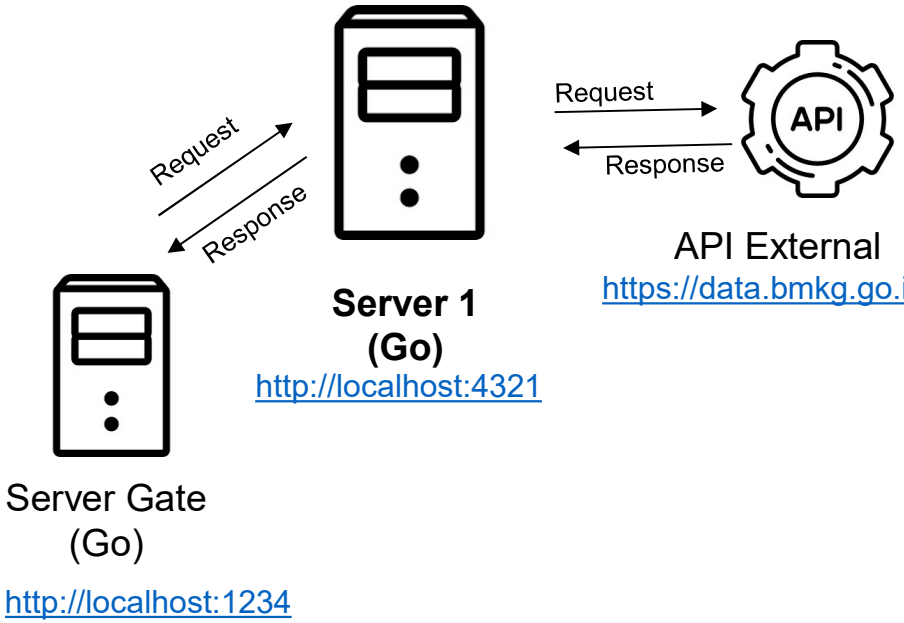
6. Server 1 akses https://data.bmkg.go.id/autogempa.xml lakukan konversi dari xml ke json

Prasyarat

- https://riptutorial.com/redis/example/29962/installing-and-running-redis-server-on-windows
    - Download https://github.com/MSOpenTech/redis/releases/download/win-3.2.100/Redis-x64-3.2.100.zip
    - Double click redis-server.exe

- Run **go run main.go**

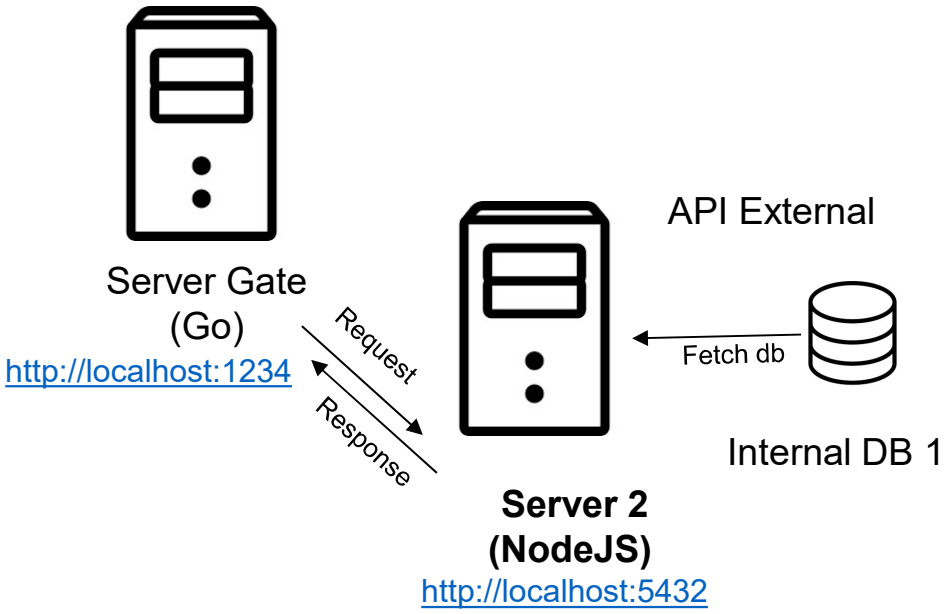
Arsitektur Server 1 (30 menit)



Fitur dalam Server:

1. Menghubungkan ke API External (https://data.bmkg.go.id)
2. Menerima request dari Server Gate
3. Memberikan Response ke server gate

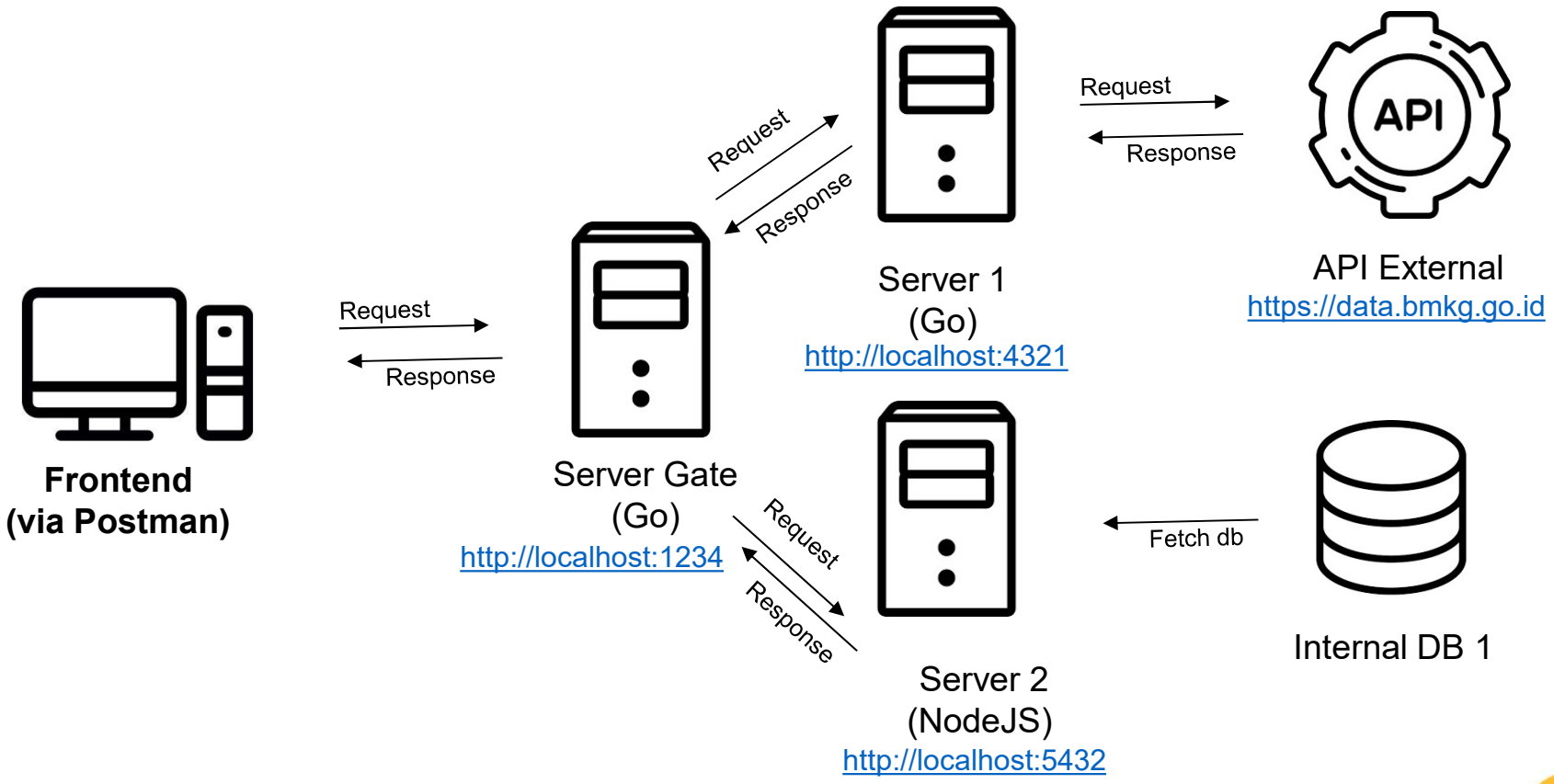
Arsitektur Server 2 (30 menit)



Fitur dalam Server:

1. Menghubungkan ke database internal dengan MySQL / NoSQL
2. Menerima request dari Server Gate
3. Memberikan Response ke server gate

Ujicoba via Postman (30 menit)



DIGITAL TALENT SCHOLARSHIP

#DIGITALINAJA

TERIMA KASIH

KOMINFO
BADAN PENELITIAN DAN PENGEMBANGAN SUMBER DAYA MANUSIA
Jl. Medan Merdeka Barat No. 9
Jakarta Pusat, 10110