

Laporan Praktek Topik Khusus III

Message Queue



SEMESTER VI

DISUSUN OLEH :

MUHAMMAD ABEL AL-FAHREZI (2211083034)

**PROGRAM STUDI TEKNOLOGI REKAYASA PERANGKAT LUNAK
JURUSAN TEKNOLOGI INFORMASI
POLITEKNIK NEGERI PADANG
2025**

A. Landasan Teori

1. Golang (Go Language)

Golang, yang dikembangkan oleh Google, merupakan bahasa pemrograman sumber terbuka yang dirancang untuk efisiensi, kesederhanaan, dan keandalan. Dengan dukungan bawaan untuk konkurensi melalui goroutine dan channel, Golang sangat cocok untuk pengembangan aplikasi web, sistem terdistribusi, dan aplikasi jaringan. Fitur garbage collection otomatis mengurangi beban pengelolaan memori, sementara sifat statis diketiknya membantu mendeteksi kesalahan pada waktu kompilasi. Kinerja tinggi Golang menjadikannya pilihan populer untuk membangun API dan layanan web, dengan framework seperti Gin dan Echo yang mempermudah pengembangan aplikasi web.

2. API

API (Application Programming Interface) merupakan fondasi utama dalam pengembangan aplikasi web modern. API memungkinkan aplikasi yang berbeda untuk berkomunikasi dan bertukar data. Dalam konteks ini, RESTful API menjadi pilihan populer karena arsitekturnya yang berbasis pada prinsip-prinsip HTTP, memungkinkan operasi CRUD (Create, Read, Update, Delete) dilakukan dengan mudah dan efisien. Operasi CRUD sendiri merupakan inti dari manipulasi data, di mana 'Create' digunakan untuk membuat data baru, 'Read' untuk membaca data, 'Update' untuk memperbaiki data, dan 'Delete' untuk menghapus data.

3. Erlang

Erlang adalah bahasa pemrograman fungsional yang dirancang khusus untuk membangun sistem yang sangat konkuren, toleran terhadap kesalahan, dan dapat diskalakan. Dengan model konkurensi berbasis proses ringan dan pertukaran pesan, Erlang memungkinkan penanganan banyak tugas secara paralel dengan efisien. Filosofi "biarkan gagal" (let it crash) dan kemampuan untuk isolasi proses membuat Erlang sangat cocok untuk aplikasi yang memerlukan keandalan tinggi, seperti sistem telekomunikasi dan perpesanan. Erlang juga dikenal dengan OTP (Open Telecom Platform), sebuah kerangka kerja yang menyediakan pustaka dan prinsip desain untuk membangun sistem yang tangguh.

4. RabbitMQ

RabbitMQ adalah sebuah message broker yang bersifat open-source yang mengimplementasikan protokol Advanced Message Queuing Protocol (AMQP). Fungsinya adalah untuk menerima dan meneruskan pesan. RabbitMQ memungkinkan aplikasi-aplikasi untuk berkomunikasi dan berbagi data secara efisien, bahkan dalam lingkungan yang terdistribusi. RabbitMQ mendukung

berbagai pola pertukaran pesan, seperti point-to-point dan publish/subscribe, sehingga sangat fleksibel untuk berbagai kasus penggunaan.

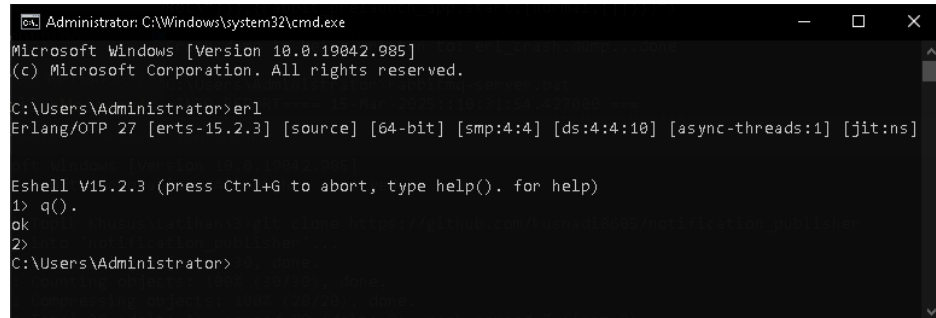
B. Tools

- Erlang
- Golang
- Visual Studio Code
- RabbitMQ

C. Langkah Kerja

1. Install Erlang dan RabbitMQ

- Download dan Install Erlang (RabbitMQ memerlukan erlang)
- Pastikan erlang terinstall

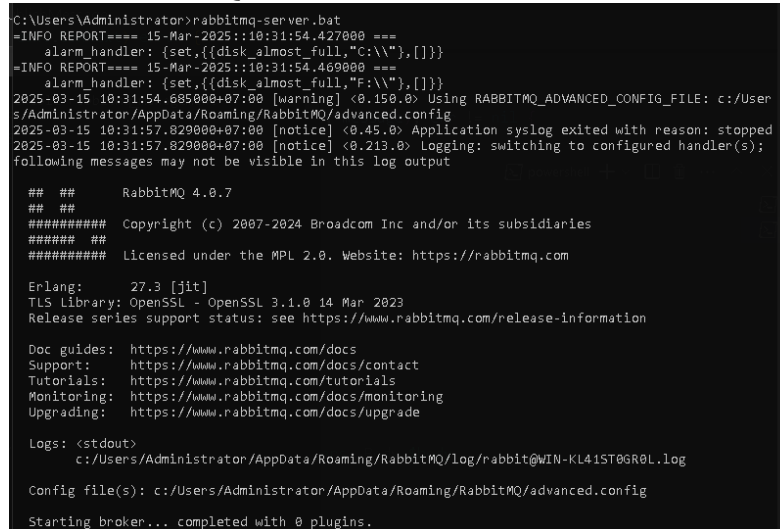


```
Administrator: C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.19042.985]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Administrator>erl
Erlang/OTP 27 [erts-15.2.3] [source] [64-bit] [smp:4:4] [ds:4:4:10] [async-threads:1] [jit:ns]

Eshell V15.2.3 (press Ctrl+G to abort, type help(). for help)
1> q().
ok
2>
C:\Users\Administrator>
```

- Download dan Install RabbitMQ
- Menambahkan RabbitMQ ke PATH
- Menjalankan RabbitMQ



```
C:\Users\Administrator>rabbitmq-server.bat
=INFO REPORT==== 15-Mar-2025:10:31:54.427000 ===
alarm_handler: {set,{[disk_almost_full,"C:\\"],[]}}
=INFO REPORT==== 15-Mar-2025:10:31:54.469000 ===
alarm_handler: {set,{[disk_almost_full,"F:\\"],[]}}
2025-03-15 10:31:54.685000+07:00 [warning] <0.150.0> Using RABBITMQ_ADVANCED_CONFIG_FILE: c:/User
s/Administrator/AppData/Roaming/RabbitMQ/advanced.config
2025-03-15 10:31:57.829000+07:00 [notice] <0.45.0> Application syslog exited with reason: stopped
2025-03-15 10:31:57.829000+07:00 [notice] <0.213.0> Logging: switching to configured handler(s);
following messages may not be visible in this log output

## ##
## ##
#####
##### Copyright (c) 2007-2024 Broadcom Inc and/or its subsidiaries
#####
##### Licensed under the MPL 2.0. Website: https://rabbitmq.com

Erlang: 27.3 [jit]
TLS library: OpenSSL - OpenSSL 3.1.0 14 Mar 2023
Release series support status: see https://www.rabbitmq.com/release-information

Doc guides: https://www.rabbitmq.com/docs
Support: https://www.rabbitmq.com/docs/contact
Tutorials: https://www.rabbitmq.com/docs/tutorials
Monitoring: https://www.rabbitmq.com/docs/monitoring
Upgrading: https://www.rabbitmq.com/docs/upgrade

Logs: <stdout>
c:/Users/Administrator/AppData/Roaming/RabbitMQ/log/rabbit@WIN-K141ST0GR0L.log
Config file(s): c:/Users/Administrator/AppData/Roaming/RabbitMQ/advanced.config
Starting broker... completed with 0 plugins.
```

2. Clone Repository Notification Publisher

- lakukan clone repository yang akan dijadikan bahan praktek
- membuat file go.mod di direktori Anda, yang akan melacak dependensi proyek Anda.
- menambahkan dependensi yang diperlukan yang ditemukan dalam kode Anda ke file go.mod.
- menyalin semua dependensi proyek ke direktori vendor di dalam direktori proyek Anda.

```
Administrator: C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19042.985]
(c) Microsoft Corporation. All rights reserved.

F:\Abel\Topik Khusus\Latihan\3>git clone https://github.com/kusnadi8605/notification_publisher
Cloning into 'notification_publisher'...
remote: Enumerating objects: 30, done.
remote: Counting objects: 100% (30/30), done.
remote: Compressing objects: 100% (20/20), done.
remote: Total 30 (delta 4), reused 28 (delta 2), pack-reused 0 (from 0)
Receiving objects: 100% (30/30), 5.61 KiB | 630.00 KiB/s, done.
Resolving deltas: 100% (4/4), done.

F:\Abel\Topik Khusus\Latihan\3>cd notification_publisher

F:\Abel\Topik Khusus\Latihan\3\notification_publisher>go mod init notification_publisher
go: creating new go.mod: module notification_publisher

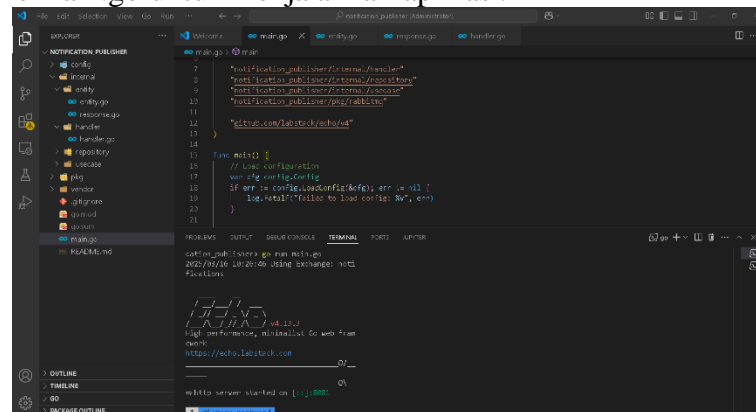
F:\Abel\Topik Khusus\Latihan\3\notification_publisher>go mod tidy
go: to add module requirements and sums:
    go mod tidy

F:\Abel\Topik Khusus\Latihan\3\notification_publisher>go mod tidy
go: finding module for package github.com/labstack/echo/v4
go: finding module for package github.com/rabbitmq/amqp091-go
go: finding module for package github.com/kelseyhightower/envconfig
go: downloading github.com/kelseyhightower/envconfig v1.4.0
go: downloading github.com/rabbitmq/amqp091-go v1.10.0
go: found github.com/labstack/echo/v4 in github.com/labstack/echo/v4 v4.13.3
go: found github.com/kelseyhightower/envconfig in github.com/kelseyhightower/envconfig v1.4.0
go: found github.com/rabbitmq/amqp091-go in github.com/rabbitmq/amqp091-go v1.10.0
go: downloading go.uber.org/goleak v1.3.0

F:\Abel\Topik Khusus\Latihan\3\notification_publisher>go mod vendor
```

3. Jalankan Aplikasi

- Run file main.go untuk menjalankan aplikasi.



4. Menguji Aplikasi

- Mengirim pesan ke server

- Dan berikut adalah output nya.

5. Clone Aplikasi Consumer

- lakukan clone repository yang akan dijadikan bahan praktek
- membuat file go.mod di direktori Anda, yang akan melacak dependensi proyek Anda.
- menambahkan dependensi yang diperlukan yang ditemukan dalam kode Anda ke file go.mod.
- menyalin semua dependensi proyek ke direktori vendor di dalam direktori proyek Anda.

```
Administrator: C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19042.985]
(c) Microsoft Corporation. All rights reserved.

F:\Abel\Topik Khusus\Latihan\3\notification_consumer>git clone https://github.com/kusnadi8605/notification_consumer
Cloning into 'notification_consumer'...
remote: Enumerating objects: 32, done.
remote: Counting objects: 100% (32/32), done.
remote: Compressing objects: 100% (16/16), done.
remote: Total 32 (delta 6), reused 32 (delta 6), pack-reused 0 (from 0)
Receiving objects: 100% (32/32), 4.22 KiB | 617.00 KiB/s, done.
Resolving deltas: 100% (6/6), done.

F:\Abel\Topik Khusus\Latihan\3\notification_consumer>cd notification_consumer

F:\Abel\Topik Khusus\Latihan\3\notification_consumer\notification_consumer>go mod init notification_consumer
go: creating new go.mod: module notification_consumer
go: to add module requirements and sums:
  go mod tidy

F:\Abel\Topik Khusus\Latihan\3\notification_consumer\notification_consumer>go mod tidy
go: finding module for package github.com/rabbitmq/amqp091-go
go: finding module for package github.com/kelseyhightower/envconfig
go: found github.com/kelseyhightower/envconfig in github.com/kelseyhightower/envconfig v1.4.0
go: found github.com/rabbitmq/amqp091-go in github.com/rabbitmq/amqp091-go v1.10.0

F:\Abel\Topik Khusus\Latihan\3\notification_consumer\notification_consumer>go mod vendor

F:\Abel\Topik Khusus\Latihan\3\notification_consumer\notification_consumer>code .

F:\Abel\Topik Khusus\Latihan\3\notification_consumer\notification_consumer>
```

6. Menjalankan Aplikasi Consumer

- Run file main.go pada folder email

```
PS F:\Abel\Topik Khusus\Latihan\3\notification_customer\notification_cons
umer> go run cmd/email/main.go
2025/03/16 10:21:44 Using Exchange: notifications
2025/03/16 10:21:44 [EMAIL] Listening for messages...
█
```

➤ Run file main.go pada folder SMS

```
PS F:\Abel\Topik Khusus\Latihan\3\notification_customer\notification_cons
umer> go run cmd/sms/main.go
2025/03/16 10:22:54 Using Exchange: notifications
2025/03/16 10:22:54 [SMS] Listening for messages...
█
```

➤ Run file main.go pada folder fcm

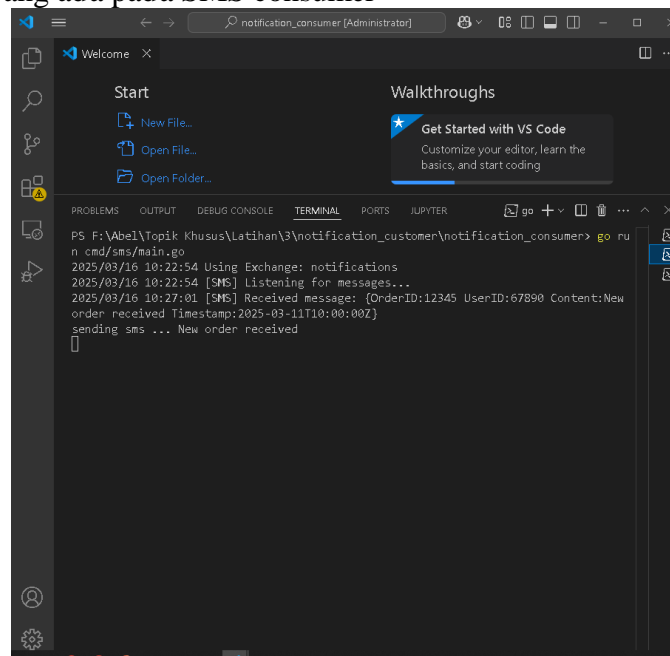
```
PS F:\Abel\Topik Khusus\Latihan\3\notification_customer\notification_cons
umer> go run cmd/fcm/main.go
2025/03/16 10:23:09 Using Exchange: notifications
2025/03/16 10:23:09 [FCM] Listening for messages...
█
```

7. Respon atau outputnya

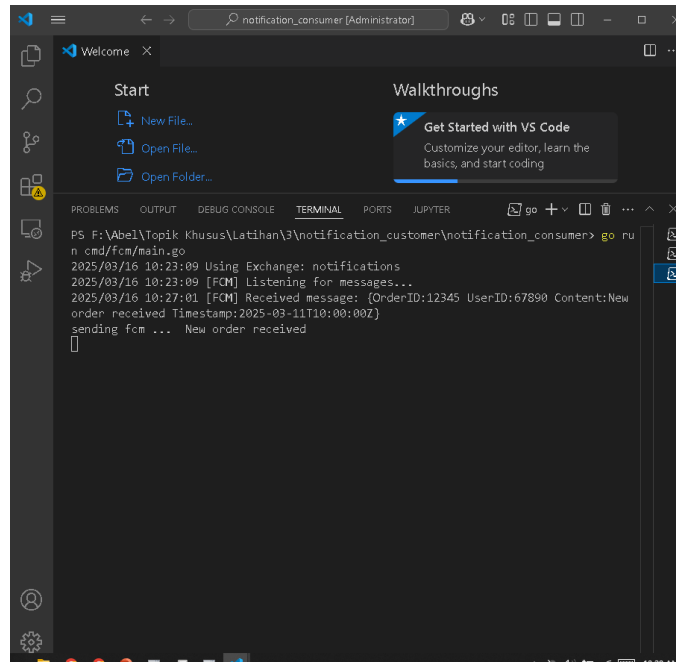
➤ Respon yang ada pada email consumer

```
2025/03/16 10:27:01 [EMAIL] Received message: {OrderID:12345 UserID:67890 Content:Ne
w order received Timestamp:2025-03-11T10:00:00Z}
sending email ... New order received
█
```

➤ Respon yang ada pada SMS consumer



➤ Respon yang ada pada FCM consumer



D. Kesimpulan

Penggunaan RabbitMQ sebagai *message queue* dengan Go menghadirkan solusi yang kuat untuk membangun aplikasi yang membutuhkan komunikasi asinkron yang andal dan skalabel. Dengan RabbitMQ, kita dapat memisahkan komponen aplikasi menjadi produsen dan konsumen, memungkinkan komunikasi yang efisien dan fleksibel. Go, dengan pustaka RabbitMQ-nya, memungkinkan implementasi berbagai pola pesan dan penanganan kesalahan yang efektif, memastikan aplikasi dapat menangani kegagalan dan masalah lainnya dengan baik. Keandalan dan skalabilitas RabbitMQ, ditambah dengan efisiensi Go, menjadikan kombinasi ini sangat cocok untuk aplikasi yang memerlukan kinerja tinggi dan ketersediaan yang tinggi..