

Pengembangan Jasa Pengiriman Barang Menggunakan Program Golang Map Delivery Order Di Pt Ruang Raya Indonesia

Muhammad Prada Hiresando¹, Anggiet Bracmatya², Febrian Murti Dewanto³, Bambang Agus Herlambang⁴

^{1,3,4}Informatika, Universitas PGRI Semarang, Kota Semarang

²PT. Ruang Raya Indonesia, Kota Jakarta Selatan

*Email korespondensi: pradasando@gmail.com

Abstract

The system backend is the part that is responsible for managing applications and databases so that they can communicate with each other properly and smoothly to support the system interface to work according to its function. This information system aims to facilitate companies in shipping goods in several cities. The research method applied in this study is the Waterfall method. The Waterfall method is one of the SDLC models that is often used in the development of information systems or software. By using the golang map delivery order, the company stores order information in a format and creates a function that can filter by day and calculate the total cost of delivery. There are also parameters that must be input which contain order data and the day of delivery of goods. This Information System can also calculate additional costs, namely the shipping admin in the form of what percentage based on the day and will get results by returning values in the form of folders containing filtered order data and calculating the total shipping costs. Using Map Delivery Order with features that can assist services in shipping goods and simplify calculations when ordering transactions to destination cities.

Keywords; Backend, Information System, Golang, Map Delivery Order, Filtering, Delivery.

Abstrak

Backend sistem merupakan bagian yang bertanggung jawab dalam mengelola aplikasi dan database agar dapat saling berkomunikasi dengan baik dan lancar guna mendukung antarmuka sistem bekerja sesuai dengan fungsinya. Sistem Informasi ini bertujuan untuk mempermudah perusahaan dalam pengiriman barang di beberapa kota. Metode penelitian yang diterapkan pada penelitian ini adalah metode Waterfall. Metode Waterfall merupakan salah satu model SDLC yang sering digunakan dalam pengembangan sistem informasi atau perangkat lunak. Dengan menggunakan golang map delivery order perusahaan tersebut menyimpan informasi pesanan dalam format dan membuat sebuah fungsi yang dapat melakukan filtering berdasarkan hari dan menghitung total biaya pengiriman. Terdapat juga parameter yang harus diinputkan yang berisi data-data pesanan dan hari pengiriman barang. Pada Sistem Informasi ini juga dapat menghitung biaya tambahan yaitu admin pengiriman yang berupa berapa persen berdasarkan hari dan akan mendapatkan hasil dengan mengembalikan nilai berupa map yang berisi data-data pesanan yang telah di filter dan perhitungan total biaya pengiriman. Oleh sebab itu, menggunakan Map Delivery Order dengan fitur yang dapat membantu jasa dalam pengiriman barang dan mempermudah perhitungan saat transaksi pemesanan ke kota tujuan.

Kata Kunci; Backend, Sistem Informasi, Golang, Map Delivery Order, Filtering, Pengiriman.

1. Pendahuluan

Teknologi Informasi adalah suatu teknologi yang digunakan untuk mengolah data termasuk memproses, mendapatkan, menyusun, menyimpan, memanipulasi data dalam berbagai cara untuk menghasilkan informasi yang berkualitas, yaitu informasi yang relevan, akurat dan tepat waktu, yang digunakan untuk keperluan pribadi, bisnis, dan pemerintahan dan merupakan informasi yang strategis untuk pengambilan keputusan.

Perkembangan Teknologi Informasi memacu suatu cara baru dalam kehidupan, dari kehidupan dimulai sampai dengan berakhir, kehidupan seperti ini dikenal dengan e-life, artinya kehidupan ini sudah dipengaruhi oleh berbagai kebutuhan secara elektronik. Dan sekarang ini sedang semarak dengan berbagai huruf yang dimulai dengan awalan e seperti e-commerce, e-government, e-education, e-library, e-journal, e-medicine, e-laboratory, e-biodiversi, dan yang lainnya lagi yang berbasis elektronika[1].

Ruangguru merupakan perusahaan rintisan dari PT. Ruang Raya Indonesia yang resmi berdiri pada tahun 2014, merupakan perusahaan teknologi terbesar dan terlengkap di Indonesia yang berfokus pada layanan berbasis pendidikan non-formal. Perusahaan ini dipimpin oleh seorang CEO ditemani oleh seorang CPO. Dan Ruangguru menjadi perusahaan rintisan terbaik di Indonesia dalam aspek teknologi pendidikan.

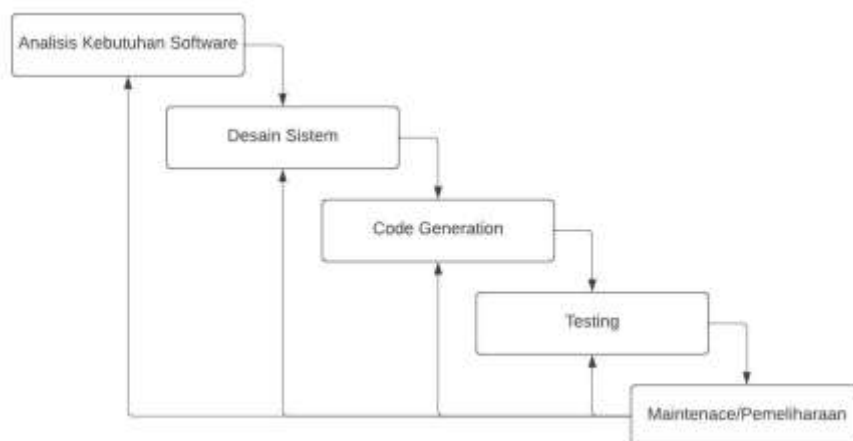
Ruangguru adalah aplikasi belajar terlengkap yang dapat membantu segala kesulitan belajar. Aplikasi Ruangguru menyediakan sistem tata kelola pembelajaran yang dapat digunakan anak untuk mendapatkan ilmu pengetahuan baik dalam bentuk audio, visual maupun audio visual[2].

Kegiatan ekonomi pada dasarnya merupakan suatu kegiatan yang berhubungan dengan benda dan harta yang dapat dikelola dan dimanfaatkan[3]. Dengan menggunakan Map Delivery Order ini, perusahaan yang menyediakan jasa pengiriman barang yang awalnya kesulitan dalam menentukan harga barang saat mengirimkan barang ke kota tujuan yang berbeda-beda, jam operasional yang kurang teratur dalam menerima dan mengirimkan barang, dan juga menentukan kota-kota yang akan menjadi tujuan pengiriman. Akan bisa menjadi lebih baik dan teratur.

Sistem Informasi ini nantinya bertujuan untuk menyediakan jasa pengiriman barang dengan harga berbeda-beda berdasarkan kota tujuan dan hari pengiriman. Perusahaan tersebut menyimpan informasi pesanan dalam format dan kode kota tujuan. Setelah itu, dengan fungsi akan mengembalikan nilai berupa map yang berisi data-data pesanan yang telah di filter dan perhitungan total biaya pengiriman.

2. Metode

Metode penelitian yang diterapkan pada penelitian ini adalah metode Waterfall. Metode Waterfall merupakan salah satu model SDLC yang sering digunakan dalam pengembangan sistem informasi atau perangkat lunak. Model ini menggunakan pendekatan sistematis dan berurutan. Tahapan dalam model ini dimulai dari tahap perencanaan hingga tahap pengelolaan (maintenance) dan dilakukan secara bertahap. Pengembang perlu mengetahui lebih lanjut tentang bagaimana proses pengembangan sistem jika menggunakan model waterfall dan juga karakteristik dari model waterfall tersebut. Adapun model waterfall yang digunakan dapat dilihat pada Gambar 1.



Gambar 1. Model Waterfall

Adapun penjelasan dari metode pengembangan sistem yang terdapat pada Gambar 1 adalah sebagai berikut:

1. Analisis Kebutuhan Software

Proses pengumpulan kebutuhan dilakukan secara intensif untuk mespesifikasikan kebutuhan perangkat lunak agar dapat dipahami perangkat lunak seperti apa yang dibutuhkan oleh user. Penulis melakukan analisis kebutuhan fungsional dalam perancangan sistem jasa pengiriman barang yang meliputi data customer, data penerima, data tujuan, data tarif, data transaksi, laporan transaksi[4].

2. Desain Sistem

Pada tahap ini penulis melakukan desain sistem untuk memecahkan permasalahan yang terjadi dimana penulis merancang sistem berorientasi objek menggunakan usecase diagram, perancangan input, perancangan output, dan perancangan struktur data sesuai dengan kebutuhan yang diperlukan. Dan desain sitem yang akan dirancang terdapat 1 orang aktor yaitu admin. Dengan aktor admin dapat melakukan mengelola data admin, mengelola data kategori, mengelola data informasi pesanan, melihat detail pengiriman, memproses data pembayaran, melihat dan mencetak laporan[5].

3. Code Generation

Rancangan sistem dijadikan sebagai untuk pengimplementasian atau pembuatan kode program. Penulisan kode program merupakan tahap penerjemahan desain yang dibuat kedalam bentuk perintah-perintah dengan menggunakan bahasa pemrograman. Tahapan ini penulis menggunakan Visual Studio Code sebagai text editor dan dirancang sesuai dengan kebutuhannya dan melakukan pengujian untuk setiap unit atau modul yang telah dibuat yang terdiri dari mengelola data admin, mengelola data kategori, mengelola data informasi pesanan, melihat detail pengiriman, memproses data pembayaran, melihat dan mencetak laporan yang diperlukan, Golang sebagai bahasa pemrograman.

4. Testing

Untuk meminimalisir kesalahan pengujian fokus pada perangkat lunak secara dari segi logika dan fungsional dan memastikan bahwa semua bagian sudah diuji. Penulis menggunakan whitebox testing sebagai metode pengujian yang dilakukan dengan cara menganalisa dan meneliti struktur internal dan kode dari perangkat lunak lalu memasukan input dan mengecek apakah output sesuai dengan yang diharapkan atau tidak.

5. Maintenance / Pemeliharaan

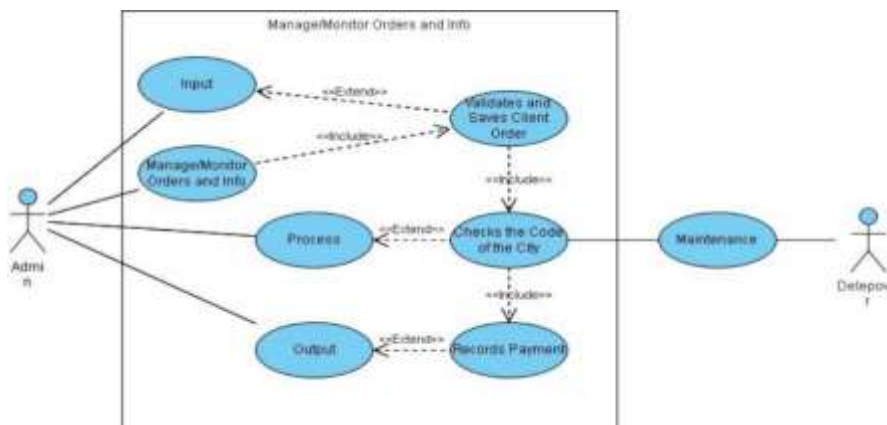
Pada tahap ini dilakukan pemeliharaan software seperti melakukan perbaikan-perbaikan terhadap sistem yang berjalan atau menambahkan fungsi tambahan sesuai dengan keinginan pihak PT. Ruang Raya Indonesia. Pada penelitian ini, peneliti hanya

melakukan proses pengembangan sistem sampai pada tahap keempat yaitu tahap testing.

3. Hasil dan Pembahasan.

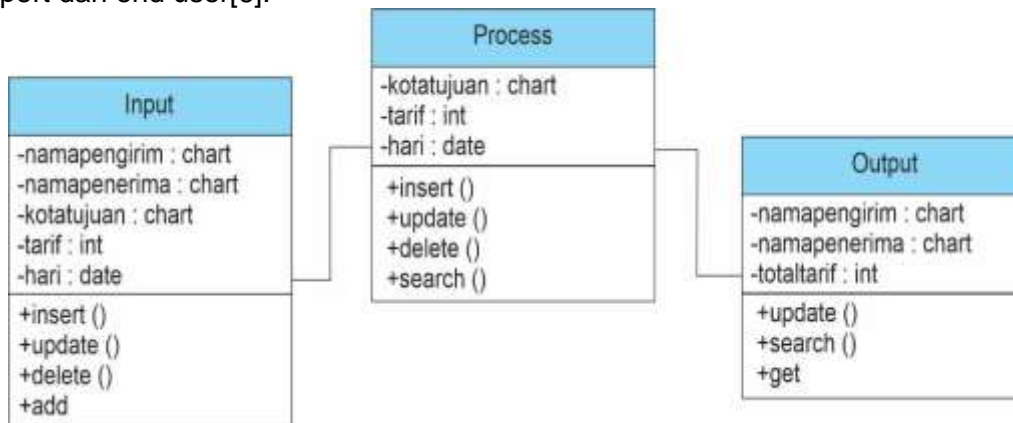
3.1. Penyajian Hasil

a. Hasil Desain Sistem



Gambar 2. Use Case Diagram

Diagram yang menggambarkan hubungan antara admin dengan system. Juga sebuah diagram yang digunakan untuk mengetahui fungsi yang ada didalam sebuah system dan mempresentasikan sebuah interaksi admin dengan system. Manfaat dari use case diagram ini untuk memudahkan komunikasi dengan menggunakan domain expert dan end user[6].

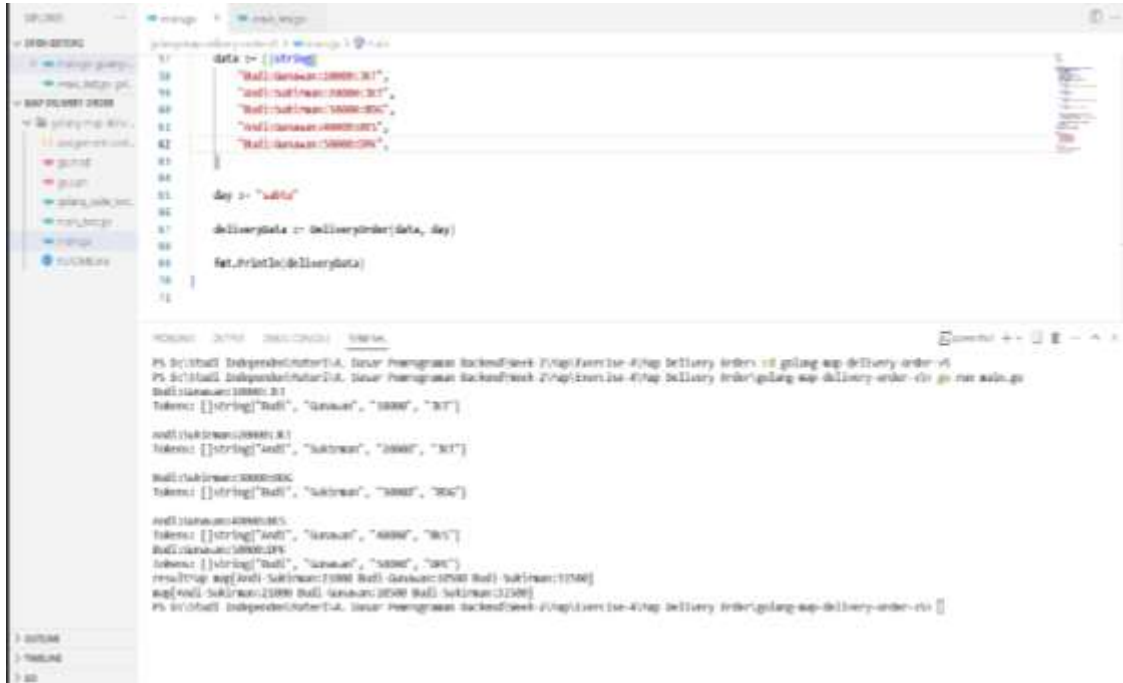


Gambar 3. Class Diagram

Diagram Class atau kelas diagram menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Di bawah ini merupakan gambar class diagram untuk menggambarkan struktur sistem yang akan dibuat untuk membangun sistem informasi manajemen ekspedisi pengiriman barang.

b. Hasil Output Map Delivery Order

Berikut adalah Hasil dari Program Golang Map Delivery Order:

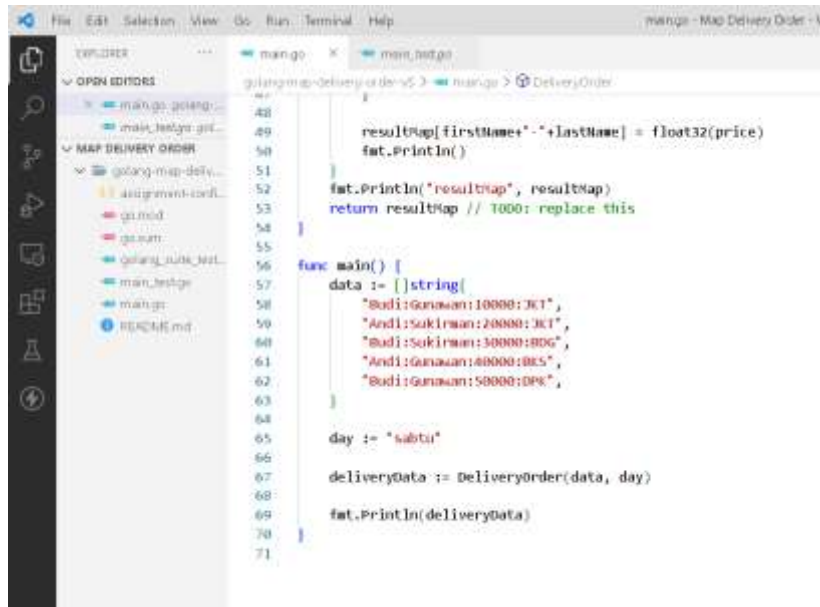


Gambar 4. Hasil Map Delivery Order

Hasil Map Delivery Order ini adalah dimana seorang admin memasukkan data pesanan dari client lalu data tersebut diproses dan diseleksi dengan fungsi yang sudah dibuat, dengan data tersebut admin juga memilih kota tujuan dengan kode yang sudah ditentukan dan juga admin menentukan harga sebelum diproses. Setelah data terproses dan diseleksi, admin memunculkan atau mengembalikan hasil dan laporan data pemesanan client yang terdiri dari nama pengirim, nama penerima, dana total harga.

3.2. Pembahasan

1. Analisis Kebutuhan Software



Gambar 3. Data Client

Proses Analisis Kebutuhan Software dilakukan secara intensif untuk mespesifikasikan kebutuhan perangkat lunak agar dapat dipahami perangkat lunak seperti apa yang dibutuhkan oleh admin. Data yang dibutuhkan saat menganalisis yaitu data pengirim, data penerima, data kota tujuan, dan hari pengiriman. Menurut data yang dikumpulkan di lapangan, disesuaikan dengan rumusan masalah dan tujuan penulisan, serta sesuai dengan langkah-langkah dan metode pengumpulan data pada bab sebelumnya, penulis akan membahas prosedur untuk jasa pengiriman di PT Ruang Raya Indonesia[7].

Setelah menganalisis langkah selanjutnya yaitu proses seleksi dan menentukan hasil laporan data transaksi yang akan dilaporkan ke client. Hasil dari proses seleksi tersebut berbentuk map yang berisi data-data pesanan yang telah di filter dan perhitungan total biaya pengiriman. Map adalah tipe data asosiatif yang ada di Go, berbentuk key-value pair. Untuk setiap data (atau value) yang disimpan, disiapkan juga key-nya. Key harus unik, karena digunakan sebagai penanda (atau identifier) untuk pengaksesan value yang bersangkutan.

2. Desain Sistem

Use case Diagram adalah gambaran efek fungsionalitas yang diharapkan oleh sistem. Diagram use case merupakan pemodelan untuk menggambarkan kelakuan (behavior) sistem yang akan dibuat. Diagram use case mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem yang akan dibuat.

Diagram yang menggambarkan hubungan antara actor dengan system. Juga sebuah diagram yang digunakan untuk mengetahui fungsi yang ada didalam sebuah system dan mempresentasikan sebuah interaksi actor dengan system. Manfaat dari use case diagram ini untuk memudahkan komunikasi dengan menggunakan domain expert dan end user.

3. Code Generation

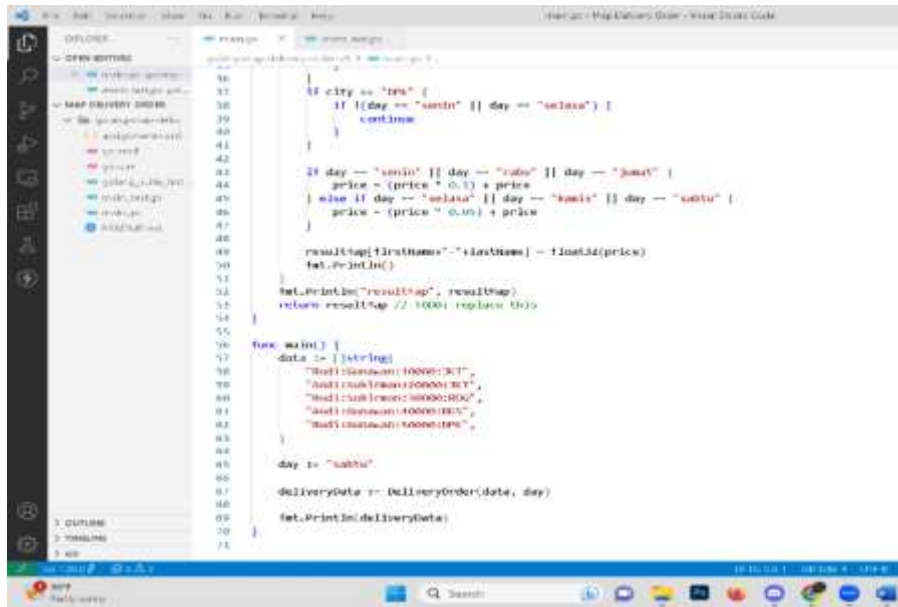
Software yang digunakan yaitu *Visual Studio Code, grader-cli, go*.

```

1 package main
2
3 import (
4     "fmt"
5     "strconv"
6     "strings"
7 )
8
9 // TODO: answer here
10
11 func DeliveryOrder(data []string, day string) map[string]float32 {
12     resultMap := make(map[string]float32)
13     for _, d := range data {
14         fmt.Println(d)
15         tokens := strings.Split(d, ",")
16         fmt.Printf("tokens: %v\n", tokens)
17         firstName := tokens[0]
18         lastName := tokens[1]
19         price, _ := strconv.ParseFloat(tokens[2], 32)
20         city := tokens[3]
21
22         if city == "DDG" {
23             if !(day == "rabu" || day == "kamis" || day == "sabtu") {
24                 continue
25             }
26         }
27         if city == "BCT" {
28             if day == "minggu" {
29                 continue
30             }
31         }
32         if city == "MCS" {
33             if !(day == "selasa" || day == "kamis" || day == "jumat") {
34                 continue
35             }
36         }
37         if city == "DKK" {

```

Gambar 4. Code Generation a

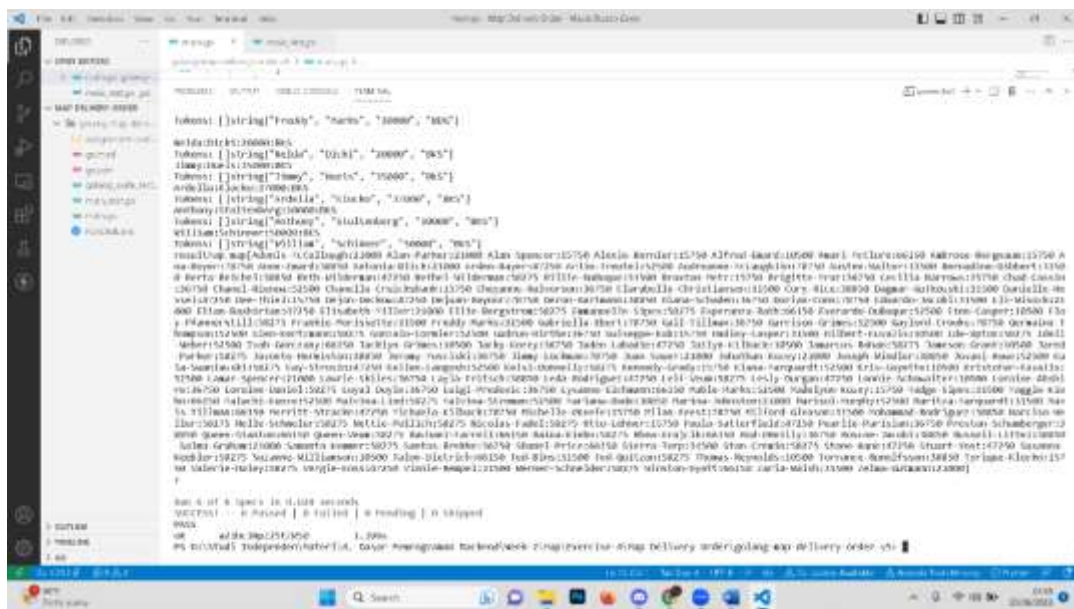


```
if city == "iris" {  
    if (day == "senin" || day == "selasa") {  
        continue  
    }  
  
    if day == "senin" || day == "jumat" || day == "sabtu" {  
        price = (price * 0.5) + price  
    } else if day == "selasa" || day == "kamis" || day == "cabtu" {  
        price = (price * 0.1) + price  
    }  
  
    resultMap["firstName"] = firstName + firstName(price)  
    set.println()  
  
    set.println("resultMap", resultMap)  
    return resultMap // 10000 replasi 000  
}  
  
func main() {  
    data := []string{  
        "Andi@gmail.com:10000:001",  
        "Andi@gmail.com:20000:002",  
        "Andi@gmail.com:30000:003",  
        "Andi@gmail.com:40000:004",  
        "Andi@gmail.com:50000:005",  
    }  
  
    day := "sabtu"  
  
    deliveryData := DeliveryOrder(data, day)  
  
    set.Println(deliveryData)  
}
```

Gambar 5. Code Generation b

Code Generation dirancang sesuai dengan kebutuhannya dan melakukan pengujian untuk setiap unit atau modul yang telah dibuat yang terdiri dari mengelola data admin, mengelola data kategori, mengelola data informasi pesanan, melihat detail pengiriman, memproses data pembayaran, melihat dan mencetak laporan yang diperlukan[8].

4. Tesing



```
tokensi []string{"Freddy", "nana", "10000", "001"}  
set.Println(10000:001)  
tokensi []string{"Melika", "Dhika", "20000", "005"}  
set.Println(10000:005)  
tokensi []string{"Tomy", "nana", "30000", "005"}  
set.Println(10000:005)  
tokensi []string{"Andi", "nana", "40000", "001"}  
set.Println(10000:001)  
tokensi []string{"Sudung", "Sulandari", "50000", "001"}  
set.Println(10000:001)  
tokensi []string{"Siti", "Siti", "60000", "005"}  
set.Println(10000:005)  
tokensi []string{"Alexa", "Alexa", "70000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "80000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "90000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "10000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "11000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "12000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "13000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "14000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "15000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "16000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "17000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "18000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "19000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "20000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "21000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "22000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "23000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "24000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "25000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "26000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "27000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "28000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "29000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "30000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "31000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "32000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "33000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "34000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "35000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "36000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "37000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "38000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "39000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "40000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "41000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "42000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "43000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "44000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "45000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "46000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "47000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "48000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "49000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "50000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "51000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "52000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "53000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "54000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "55000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "56000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "57000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "58000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "59000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "60000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "61000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "62000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "63000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "64000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "65000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "66000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "67000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "68000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "69000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "70000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "71000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "72000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "73000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "74000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "75000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "76000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "77000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "78000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "79000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "80000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "81000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "82000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "83000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "84000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "85000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "86000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "87000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "88000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "89000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "90000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "91000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "92000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "93000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "94000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "95000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "96000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "97000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "98000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "99000", "001"}  
set.Println(10000:001)  
tokensi []string{"Aldo", "Aldo", "100000", "001"}  
set.Println(10000:001)
```

Gambar 6. Testing

Untuk Testing menggunakan whitebox testing sebagai metode pengujian yang dilakukan dengan cara menganalisa dan meneliti struktur internal dan kode dari perangkat lunak lalu memasukkan input dan mengecek apakah output sesuai dengan yang diharapkan atau tidak.

White-Box Testing yaitu menguji perangkat lunak dari segi desain dan kode program apakah mampu menghasilkan fungsi-fungsi, masukan dan keluaran yang sesuai dengan spesifikasi yang dibutuhkan[9].

Dengan menggunakan whitebox testing akan mempermudah juga dalam pengecekan dimana terjadinya error, skipped, dan panicked yang terlihat setelah dijalankan melalui terminal.

Setelah melakukan testing langkah selanjutnya yaitu mencari solusi bagaimana cara mengatasi error tersebut dan jagan sampai terjadi kesalahan atau kekurangan pada coding yang nantinya bisa menyebabkan error.

5. Maintenance / Pemeliharaan

Pemeliharaan software seperti melakukan perbaikan-perbaikan terhadap sistem yang berjalan atau menambahkan fungsi tambahan sesuai dengan keinginan pihak PT. Ruang Raya Indonesia. Semisal terjadi suatu error pada saat testing, akan segera diperbaiki dan mencari solusi untuk mengatasi error tersebut.

Pemeliharaan suatu software diperlukan, termasuk di dalamnya adalah pengembangan, karena software yang dibuat tidak selamanya hanya seperti itu. Ketika dijalankan mungkin saja masih ada error kecil yang tidak ditemukan sebelumnya, atau ada penambahan fitur-fitur yang belum ada pada software tersebut. Pengembangan diperlukan ketika adanya perubahan dari eksternal perusahaan seperti ketika ada pergantian sistem operasi, atau perangkat lainnya[10].

4. Kesimpulan

Setelah melaksanakan penelitian di PT. Ruang Raya Indonesia dapat diberikan kesimpulan bahwa dengan menggunakan metode Waterfall program Map Delivery Order berhasil dirancang dan dibuat. Penelitian program jasa pengiriman barang di PT. Ruang Raya Indonesia dapat memudahkan client untuk mengirimkan barang dan membantu dalam pemesanan data client serta mengirimkan data client kepada jasa pengiriman dalam bentuk map atau hasil laporan transaksi. Pengembangan back end Map Delivery Order yang dilakukan dengan membuat suatu sistem dengan bahasa pemrograman Golang.

5. Referensi

- [1] B. Bondy, U. Klages, F. Müller-Spahn, and C. Hock, "Cytosolic free $[Ca^{2+}]$ in mononuclear blood cells from demented patients and healthy controls," *Eur. Arch. Psychiatry Clin. Neurosci.*, vol. 243, no. 5, pp. 224–228, 1994, doi: 10.1007/BF02191578.
- [2] R. Di and K. Pekanbaru, "Strategi Marketing Public Relations Pt . Ruang Raya Indonesia Dalam Mempromosikan," 2022.
- [3] M. Burhanudin, "Wakalah Bil Ujrah Dalam Investasi Jasa Pengiriman Barang," *ADLIYA J. Huk. dan Kemanus.*, vol. 13, no. 1, pp. 27–38, 2019, [Online]. Available: <https://core.ac.uk/download/pdf/234031864.pdf>
- [4] Aceng Abdul Wahid, "Analisis Metode Waterfall Untuk Pengembangan Sistem Informasi," *J. Ilmu-ilmu Inform. dan Manaj. STMIK*, no. November, pp. 1–5, 2020.
- [5] F. Iskandar, Y. A. Lola, and K. Desi, "Perancangan Aplikasi Penyewaan Alat Berat berbasis Web pada PT . Indotruck Citra Pramata Jambi," *Peranc. Apl. Penyewaan Alat Berat Berbas. Web pada PT . Indotruck Citra Pramata Jambi*, vol. 12, no. 2, pp. 1056–1068, 2017.
- [6] M. Rahmatuloh and M. Rizky Revanda, "Rancang Bangun Sistem Informasi Jasa Pengiriman Barang Pada Pt. Haluan Indah Transporindo Berbasis Web," *J. Tek. Inform.*, vol. 14, no. 1, pp. 54–59, 2022.
- [7] D. P. Dewi, H. Harjoyo, and A. Salam, "Prosedur Administrasi Jasa Pengiriman Barang Di Pt Citra Van Titipan Kilat Tangerang," *J. Sekr. Univ. Pamulang*, vol. 7, no. 1, p. 1, 2020, doi: 10.32493/skr.v7i1.4570.
- [8] I. Ahmad, A. T. Prastowo, E. Suwarni, and R. I. Borman, "Pengembangan Aplikasi

- Online Delivery Sebagai Upaya Untuk Membantu Peningkatan Pendapatan,” *JMM (Jurnal Masy. Mandiri)*, vol. 5, no. 6, pp. 4–12, 2021.
- [9] B. Iskandar and A. U. Hamdani, “Desain Dan Pengujian Sistem Informasi Jasa Pengiriman Barang Studi Kasus : Pt. Xyz,” *STMIK AMICOM Yogyakarta*, pp. 67–72, 2017.
- [10] Y. D. Wijaya and M. W. Astuti, “Sistem Informasi Penjualan Tiket Wisata Berbasis Web Menggunakan Metode Waterfall,” *Semin. Nas. Teknol. Inf. dan Komun.*, p. 274, 2019.