Cover

**PENGENALAN GOLANG DAN MEMBUAT PACKAGE**

Tutorial Pemrograman Untuk Pemula

**Muhammad Wildan Khalilurrahman**

**Rolly Maulana Awangga**

**Waskitho Cito Adiwiguno**

|  |  |
| --- | --- |
| 2023 | **PT. Penerbit Buku Pedia**  **2023** |

**PENGENALAN GOLANG DAN MEMBUAT PACKAGE**

Tutorial Pemrograman Untuk Pemula

***Penulis:***

Rolly Maulana Awangga

Muhammad Wildan Khalilurrahman

Waskitho Cito Adiwiguno

***ISBN***:

***Editor:***

M. Yusril Helmi Setyawan

***Penyunting:***

***Desain sampul dan Tata letak:***

Waskitho Cito Adiwiguno

***Font:***

Calibri

***Penerbit:***

PT. Penerbit Buku Pedia

***Redaksi:***

Athena Residence Blok. E No. 1, Desa Ciwaruga,

Kec. Parongpong, Kab. Bandung Barat 40559

Tel. 628-775-2000-300

Email : penerbit@bukupedia.co.id

***Distributor:***

Informatics Research Center

Jl. Sariasih No. 54

Bandung 40151

Email : irc@ulbi.ac.id

Cetakan Pertama, 2023

Hak cipta dilindungi undang-undang

Dilarang memperbanyak karya tulis ini dalam bentuk dan

dengan cara apa pun tanpa ijin tertulis dari penerbit

# **PRAKATA**

B

uku ini hadir sebagai sumber informasi yang lengkap dan dapat dipercaya mengenai pemrograman Go, termasuk pembuatan package dan integrasi dengan Google Sheets dan Google Docs API. Data dan statistik yang kami gunakan dalam buku ini telah melalui proses verifikasi dan memberikan gambaran nyata tentang pentingnya kehadiran buku ini dalam dunia pemrograman.

Kami menulis buku ini dengan tujuan memberikan pemahaman yang mendalam kepada pembaca tentang pemrograman Go. Kami ingin buku ini menjadi panduan yang berguna bagi pemula yang baru memasuki dunia pemrograman, serta bagi mereka yang ingin meningkatkan keahlian mereka dalam bahasa Go. Dalam setiap babnya, kami menyajikan metode penyampaian yang jelas dan terstruktur agar pembaca dapat dengan mudah memahami konsep-konsep pemrograman dan mengaplikasikannya secara praktis.

Salah satu keunggulan utama buku ini adalah tutorial yang lengkap, mulai dari langkah-langkah instalasi Go, Git, hingga teks editor yang direkomendasikan. Kami juga memberikan panduan tentang penggunaan ekstensi yang berguna dalam proses pengembangan perangkat lunak. Selain itu, buku ini juga mengupas konsep dasar pemrograman secara umum, sehingga pembaca dapat memperoleh pemahaman yang kuat sebelum memasuki materi yang lebih mendalam. Semua chapter atau bab yang tertulis didalam buku ini akan tertera juga kode program yang dapat diakses melalui link: https://github.com/gibs952/gdocsheets.git

# **DAFTAR ISI**

[**PRAKATA i**](#_Toc139823618)

[**DAFTAR ISI ii**](#_Toc139823619)

[**BAB 1**](#_Toc139823620)[**PENGENALAN 1**](#_Toc139823621)

[A. APA ITU PEMROGRAMAN? 1](#_Toc139823622)

[B. APA ITU GOLANG? 2](#_Toc139823623)

[C. GITHUB DAN GITBASH 3](#_Toc139823624)

[D. API 5](#_Toc139823625)

[1. REST API 6](#_Toc139823626)

[E. HTTP 7](#_Toc139823627)

[F. GOOGLE API 8](#_Toc139823628)

[G. TEXT EDITOR 9](#_Toc139823629)

[**BAB 2**](#_Toc139823630)[**PEMASANGAN APLIKASI 13**](#_Toc139823631)

[A. OVERVIEW 13](#_Toc139823632)

[B. PEMASANGAN GO 13](#_Toc139823633)

[C. PENGGUNAAN GITHUB 16](#_Toc139823634)

[D. PEMASANGAN GITBASH 18](#_Toc139823635)

[E. PEMASANGAN VISUAL STUDIO CODE 20](#_Toc139823636)

[F. PEMASANGAN EKSTENSI GO-LANGUAGE DI VISUAL STUDIO CODE 22](#_Toc139823637)

[**BAB 3**](#_Toc139823638)[**PEMROGRAMAN GO 25**](#_Toc139823639)

[A. KONSEP DASAR PEMROGRAMAN 25](#_Toc139823640)

[1. Algoritma 25](#_Toc139823641)

[2. Variabel, konstanta dan Tipe Data 25](#_Toc139823642)

[3. Struktur Kontrol. 26](#_Toc139823643)

[4. Fungsi dan Metode 27](#_Toc139823644)

[5. Struktur Data 27](#_Toc139823645)

[6. Input dan Output 27](#_Toc139823646)

[7. Paradigma Pemrograman 28](#_Toc139823647)

[B. MEMBUAT PROGRAM “HELLO WORD” DI GO 29](#_Toc139823648)

[1. Inisialisasi Proyek 29](#_Toc139823649)

[2. Membuat file program dan menjalankan hello world 30](#_Toc139823650)

[C. MEMAHAMI STRUKTUR SERTA KETENTUAN PENULISAN GOLANG 31](#_Toc139823651)

[1. Deklarasi Package 33](#_Toc139823652)

[2. Bagian Import 33](#_Toc139823653)

[D. METHOD MAIN 34](#_Toc139823654)

[E. EKSPRESI 35](#_Toc139823655)

[F. BLOK PROGRAM GO-LANGUAGE 35](#_Toc139823656)

[G. PENULISAN BAGIAN KOMENTAR PADA GOLANG 35](#_Toc139823657)

[H. KONSTANTA(CONSTANS) 37](#_Toc139823658)

[I. OUTPUT FUNCTION 38](#_Toc139823659)

[J. FORMATTING WEB 38](#_Toc139823660)

[K. TIPE DATA 39](#_Toc139823661)

[L. PENGENALAN OPERATOR 40](#_Toc139823662)

[1. Aritmatika (Arithmetic) 40](#_Toc139823663)

[2. Assignment 41](#_Toc139823664)

[3. Comparison 41](#_Toc139823665)

[4. Logical 42](#_Toc139823666)

[5. Bitwise 43](#_Toc139823667)

[M. KONDISI(CONDITIONS) 44](#_Toc139823668)

[1. If Statement 44](#_Toc139823669)

[2. if else Statement 45](#_Toc139823670)

[3. if-else-if Statement 46](#_Toc139823671)

[4. Else-if Statement 47](#_Toc139823672)

[5. Nested-if 48](#_Toc139823673)

[N. SWITCH 49](#_Toc139823674)

[1. Switch single-case 49](#_Toc139823675)

[2. Switch multi-case 50](#_Toc139823676)

[O. STRUCT 51](#_Toc139823677)

[P. MAPS 52](#_Toc139823678)

[Q. STRUKTUR KENDALI 53](#_Toc139823679)

[R. PERULANGAN 54](#_Toc139823680)

[S. FUNGSI 55](#_Toc139823681)

[T. ARRAY AND SLICES 56](#_Toc139823682)

[**BAB 4**](#_Toc139823683)[**MEMBUAT PACKAGE 58**](#_Toc139823684)

[A. INISIALISASI PROYEK 58](#_Toc139823685)

[B. OTENTIKASI PROYEK 58](#_Toc139823686)

[C. MENULIS KODE 59](#_Toc139823687)

[1. Google Sheet`` 59](#_Toc139823688)

[2. Google Docs 66](#_Toc139823689)

[D. MENJALANKAN *PACKAGE* GOOGLE DOCS DAN GOOGLE SHEETS 67](#_Toc139823690)

[**LATIHAN-GO 71**](#_Toc139823691)

[**DAFTAR PUSTAKA 79**](#_Toc139823692)

[**TENTANG PENULIS 81**](#_Toc139823693)

# **BAB 1**

# **PENGENALAN**



## **APA ITU PEMROGRAMAN?**

Sebelum mempelajari pemrograman, penting untuk memahami konsep program. Secara umum, program dapat diartikan sebagai suatu rancangan atau rencana yang mencakup prinsip-prinsip dan upaya yang akan dilakukan, baik dalam konteks negara, ekonomi, dan sebagainya. Dalam hal ini, kita dapat menekankan kata "rancangan". Oleh karena itu, program dalam konteks komputer dapat diartikan sebagai serangkaian perintah yang dirancang untuk mengatur dan menjalankan fungsi atau tugas tertentu pada komputer. Komputer di sini merujuk pada perangkat komputasi modern yang menggunakan prosesor untuk memproses data, seperti smartphone, laptop, Playstation, dan berbagai perangkat lainnya. Contoh program yang sering digunakan adalah aplikasi e-commerce, media sosial, dan permainan (game).

Jadi apa itu pemrograman? Pemrograman adalah proses pembuatan, pengembangan, dan penulisan serangkaian instruksi atau kode yang digunakan untuk mengendalikan perilaku komputer dan melaksanakan tugas-tugas tertentu. Secara umum, pemrograman melibatkan pemikiran logis dan algoritma untuk merancang solusi untuk masalah atau tugas tertentu, kemudian menerjemahkan solusi tersebut ke dalam bahasa yang bisa dapat dimengerti oleh komputer (Wali, et al., 2023).

Dalam pemrograman, kode atau instruksi yang ditulis oleh seorang programmer mengikuti sintaks dan aturan tertentu dalam bahasa pemrograman yang digunakan. Bahasa pemrograman adalah sistem yang digunakan untuk mengkomunikasikan instruksi kepada komputer. Setiap bahasa pemrograman memiliki aturan dan struktur yang berbeda, tetapi prinsip dasar pemrograman tetap sama.

Pemrograman memungkinkan pengembang untuk membuat berbagai jenis perangkat lunak dan aplikasi, termasuk aplikasi desktop, aplikasi web, aplikasi seluler, permainan komputer, dan banyak lagi. Dalam pemrograman, kita menggunakan algoritma untuk merancang langkah-langkah yang sistematis dan logis dalam menyelesaikan masalah. Algoritma adalah serangkaian instruksi langkah demi langkah yang diperlukan untuk mencapai tujuan tertentu.

Pemrograman juga melibatkan konsep pengelolaan data, yaitu cara mengorganisasi dan mengelola informasi yang digunakan dalam program. Data dapat disimpan dalam berbagai bentuk, seperti angka, teks, gambar, dan lainnya. Pemrograman memungkinkan kita untuk memanipulasi dan memproses data ini sesuai dengan kebutuhan.

Selain itu, pemrograman juga melibatkan penggunaan alat dan lingkungan pengembangan yang memungkinkan programmer menulis, menguji, dan menjalankan kode mereka. Alat ini termasuk editor kode, kompiler, interpreter, debugger, dan banyak lagi, tergantung pada bahasa pemrograman yang digunakan.

Pemrograman adalah keterampilan yang sangat berharga di era digital saat ini. Dengan pemrograman, kita dapat menciptakan solusi teknologi yang inovatif, memecahkan masalah yang kompleks, dan mengotomatiskan tugas-tugas rutin. Pemrograman juga melatih kita dalam berpikir logis, analitis, dan sistematis, serta mengembangkan kemampuan pemecahan masalah yang penting dalam berbagai bidang.

## **APA ITU GOLANG?**

Go-Language, atau Golang, adalah sebuah bahasa pemrograman modern yang sangat populer di kalangan pengembang perangkat lunak (Go Developer, 2023). Golang dirancang oleh tim Google dengan tujuan untuk menyediakan bahasa yang efisien, mudah dipelajari, dan dapat digunakan dalam pengembangan aplikasi berkecepatan tinggi (Go Developer, 2023).

Salah satu kelebihan utama Golang adalah kinerjanya yang tinggi. Bahasa ini dirancang dengan fokus pada kinerja eksekusi yang cepat dan efisien, sehingga sangat bagus digunakan dalam pengembangan aplikasi yang memerlukan waktu respons yang singkat dan pengolahan data yang intensif.

Golang juga menawarkan fitur-fitur yang memudahkan pengembang dalam menulis kode. Misalnya, Golang memiliki sistem pengumpulan sampah (garbage collection) yang otomatis, sehingga pengembang tidak perlu repot-repot mengelola memori secara manual. Selain itu, Golang memiliki mekanisme konkurensi yang kuat, yang disebut dengan goroutine, yang memungkinkan pengembang untuk menulis kode paralel dengan mudah, meningkatkan kinerja aplikasi.

Sintaksis Golang juga cukup sederhana dan mudah dipahami. Golang menggunakan gaya penulisan camel case untuk penamaan variabel dan fungsi. Bahasa ini mendukung tipe data dasar seperti angka, string, dan boolean, serta menyediakan tipe data terstruktur seperti array, slice, dan map.

Selain itu, Golang memiliki dukungan yang kuat untuk pengembangan aplikasi web dan layanan berbasis mikro. Golang menyediakan paket standar yang lengkap untuk mengakses HTTP, memungkinkan pengembang untuk membuat server web dengan mudah. Selain itu, terdapat pustaka-pustaka pihak ketiga yang luas yang mendukung pengembangan aplikasi berbasis web, basis data, kecerdasan buatan, dan masih banyak lagi.

Dengan kombinasi kecepatan eksekusi yang tinggi, kesederhanaan sintaksis, dukungan konkurensi, dan paket standar yang kuat, Golang menjadi pilihan populer bagi pengembang untuk membangun berbagai jenis aplikasi, mulai dari aplikasi skala kecil hingga sistem berkecepatan tinggi yang kompleks.

## **GITHUB DAN GITBASH**

GitHub adalah platform yang sangat berguna dalam pengelolaan kode sumber proyek (Riko, 2022). Dengan GitHub, kalian dapat menyimpan proyek kalian secara terstruktur dan mengelolanya dengan mudah. Kalian bahkan bisa berkolaborasi dengan teman-teman kalian dalam tim pengembangan. Ini merupakan kesempatan luar biasa untuk belajar dari orang lain, berbagi kode, dan membangun sesuatu bersama-sama. Untuk memulai, kalian perlu membuat akun GitHub. Jangan khawatir, prosesnya cukup sederhana dan kalian bisa langsung mengakses semua fitur hebat yang ditawarkan GitHub.

Berikut adalah beberapa fitur utama github:

1. Repositori: GitHub memungkinkan Anda membuat repositori untuk menyimpan dan mengelola kode sumber proyek. Repositori dapat berisi berbagai jenis file, seperti file kode, gambar, dan dokumen.
2. Kolaborasi: Dalam GitHub, Anda dapat mengundang kolaborator lain untuk berkontribusi pada repositori Anda. Kolaborator dapat melakukan perubahan, mengusulkan perubahan melalui pull request, dan memberikan umpan balik.
3. Version Control: GitHub menggunakan sistem kontrol versi Git, yang memungkinkan Anda melacak perubahan yang dilakukan pada file-file dalam repositori. Anda dapat melihat riwayat perubahan, membandingkan versi, dan mengembalikan ke versi sebelumnya jika diperlukan.
4. Issue Tracking: GitHub menyediakan fitur pelacakan masalah (issue tracking) yang memungkinkan Anda membuat, mengelola, dan melacak masalah atau permintaan fitur dalam proyek. Hal ini memudahkan komunikasi antara pengembang dan pengguna, serta memungkinkan pemantauan progres penyelesaian masalah.
5. GitHub Pages: Fitur ini memungkinkan Anda untuk mempublikasikan situs web statis secara langsung dari repositori GitHub. Anda dapat membuat halaman dokumentasi, blog, atau proyek showcase dengan mudah.

Selanjutnya, kita akan membahas GitBash. GitBash adalah antarmuka baris perintah yang memungkinkan pengguna Windows untuk menggunakan Git dengan mudah. Git adalah sistem pengendalian versi yang digunakan oleh GitHub. Dengan GitBash, kalian dapat menjalankan perintah-perintah Git dengan cepat dan efisien. Misalnya, kalian bisa membuat repositori, mengelola perubahan kode, dan berkolaborasi dengan tim pengembang lainnya. GitBash akan membantu kalian memahami dasar-dasar Git dan mempermudah proses pengelolaan kode kalian.

Dengan GitHub dan GitBash, kalian akan memiliki alat yang sangat bermanfaat dalam perjalanan belajar kalian. GitHub akan membantu kalian mengelola proyek kalian secara efektif dan berkolaborasi dengan orang lain, sementara GitBash akan mempermudah penggunaan Git melalui antarmuka baris perintah yang mudah dipahami.

Berikut adalah beberapa poin penting tentang Git Bash:

1. Git Command Line: Git Bash memberikan Anda akses ke perintah-perintah Git melalui baris perintah. Anda dapat melakukan operasi seperti inisialisasi repositori, membuat commit, membuat dan menggabungkan cabang, mengelola remote repository, dan banyak lagi.
2. Shell Environment: Git Bash menggunakan shell (lingkungan) yang didasarkan pada Bash, yang merupakan shell standar di banyak sistem operasi Unix. Ini memberikan Anda antarmuka yang mirip dengan terminal di Linux atau macOS.
3. Unix Tools: Git Bash juga menyertakan sejumlah alat-alat Unix yang berguna, seperti utilitas ls, cp, mv, rm, dan sebagainya. Hal ini memungkinkan Anda untuk menggunakan perintah-perintah tersebut di Windows tanpa memerlukan instalasi utilitas tambahan.
4. Scripting: Dengan Git Bash, Anda dapat menulis dan menjalankan script shell menggunakan sintaks Bash. Ini memungkinkan otomatisasi tugas dan penggunaan skrip dalam alur kerja Git.
5. Integrasi dengan Windows: Git Bash dapat berintegrasi dengan lingkungan Windows.

## **API**

Gambar 1. Ilustrasi berjalannya API

Coba perhatikan aplikasi sehari-hari yang kita gunakan di ponsel kita. Kita ambil contoh aplikasi whatsapp. Selain bisa mengirim pesan teks dan pesan suara, whatsapp juga bisa mengirimkan lokasi ponsel kita. Jika kita teliti, ketika kita membagikan lokasi kita kepada teman kita atau saudara kita, kita akan melihat bahwa ada tulisan Google pada peta lokasi yang kita bagikan. Artinya whatsapp menggunakan google maps untuk mendapatkan lokasi kita secara akurat. Namun bagaimana whatsapp bisa memdapatkan data bahkan fungsi dari google maps, padahal whatsapp dan google maps adalah aplikasi yang berbeda? Ini semua dapat dilakukan dengan adanya API.

API adalah singkatan dari Application Programming Interface, yang merupakan antarmuka pemrograman aplikasi. API terdiri dari serangkaian aturan dan protokol yang memungkinkan berbagai aplikasi perangkat lunak berinteraksi dan saling berkomunikasi (Reddy, 2011). Seperti halnya manusia

API *memungkinkan* pengembang perangkat lunak untuk menggunakan fungsionalitas yang telah dibuat oleh pengembang lain tanpa harus mengetahui detail implementasi internalnya. Dalam konteks web, API sering digunakan untuk menghubungkan dan berkomunikasi antara server dan aplikasi klien.

API biasanya menggunakan protokol komunikasi standar seperti HTTP (Hypertext Transfer Protocol) untuk mengirimkan permintaan dan menerima respons. API dapat digunakan untuk mengakses dan memanipulasi data, menjalankan operasi tertentu, atau mengintegrasikan fungsionalitas dari sistem yang berbeda.

Contoh penggunaan API termasuk mengakses data dari media sosial seperti Facebook dan Twitter, mengintegrasikan pembayaran online menggunakan API pembayaran seperti PayPal, atau mengambil data lokas dari layanan lokasi melalui API google maps. Berikut adalah contoh kasus mengakses data dari API Media Sosial:

API REST dapat digunakan untuk mengakses dan memanipulasi data dari platform media sosial seperti Facebook atau Twitter. Misalnya, dengan menggunakan API Twitter, seorang pengembang dapat membuat aplikasi yang dapat mengambil data tweet terbaru dari pengguna tertentu atau memposting tweet melalui aplikasi mereka.

Ada beberapa macam API tapi yang akan kita gunakan dalam pemrograman di buku ini adalah REST API.

### REST API

REST API adalah salah satu jenis API yang didasarkan pada prinsip-prinsip Representational State Transfer (REST). REST adalah arsitektur yang dirancang untuk memfasilitasi komunikasi antara sistem yang terdistribusi melalui protokol HTTP (Fielding & Taylor, 2002).

API REST menggunakan metode HTTP seperti GET, POST, PUT, DELETE, dll. untuk mengakses dan berinteraksi dengan sumber daya (resources) yang ada pada server. Setiap sumber daya diberikan URI (Uniform Resource Identifier) yang unik untuk diakses.

Beberapa karakteristik REST API yang penting antara lain:

1. Stateless: Setiap permintaan ke server harus mencakup semua informasi yang diperlukan. Server tidak menyimpan status (state) klien antara permintaan, sehingga setiap permintaan dianggap mandiri.
2. Representational: Data yang dikirimkan oleh server dalam respons dapat berbentuk berbagai representasi seperti JSON, XML, HTML, atau format lainnya, sesuai dengan preferensi klien yang meminta data.
3. Resources: API REST berfokus pada sumber daya (resources) sebagai entitas utama yang dapat diakses dan dimanipulasi. Sumber daya diidentifikasi dengan URI yang unik.
4. CRUD Operations: API REST memungkinkan operasi dasar Create, Read, Update, dan Delete (CRUD) terhadap sumber daya melalui metode HTTP yang sesuai (POST, GET, PUT, DELETE).

API REST dapat digunakan untuk mengakses dan memanipulasi data dari platform media sosial seperti Facebook atau Twitter. Misalnya, dengan menggunakan API Twitter, seorang pengembang dapat membuat aplikasi yang dapat mengambil data tweet terbaru dari pengguna tertentu atau memposting tweet melalui aplikasi mereka.

Contoh protokol penggunaan API REST untuk mengambil tweet terbaru dari pengguna tertentu:

1. Membuat permintaan HTTP GET ke endpoint API Twitter yang sesuai dengan pengguna dan sumber daya yang diinginkan.
2. Server Twitter akan merespons dengan data tweet terbaru dari pengguna tersebut dalam format JSON.
3. Pengembang dapat memanipulasi data JSON yang diterima dan menggunakannya dalam aplikasi mereka.

## **HTTP**

HTTP adalah singkatan dari Hypertext Transfer Protocol (Protokol Transfer Hiperteks). HTTP adalah protokol komunikasi yang digunakan untuk mentransfer data melalui jaringan, terutama di World Wide Web (WWW) (Fielding & Taylor, 2002).

HTTP bekerja berdasarkan model klien-server, di mana klien (misalnya web browser) membuat permintaan kepada server (misalnya web server) untuk mengambil atau mengirimkan data. Permintaan HTTP terdiri dari metode (GET, POST, PUT, DELETE, dll.), URI (Uniform Resource Identifier) yang mengidentifikasi sumber daya yang diminta, header yang berisi informasi tambahan, dan kadang-kadang tubuh (body) yang berisi data yang dikirimkan.

Server kemudian memproses permintaan dan mengirimkan respons HTTP kembali kepada klien. Respons HTTP berisi kode status yang mengindikasikan apakah permintaan berhasil atau gagal, header respons yang berisi informasi tambahan, dan tubuh (body) yang berisi data yang diminta atau hasil operasi yang dilakukan oleh server.

HTTP biasanya menggunakan port 80 untuk komunikasi defaultnya, tetapi juga dapat menggunakan port lain seperti 8080 atau 443 (untuk HTTPS, versi aman dari HTTP).

HTTP mendefinisikan metode operasi seperti:

1. GET: Mengambil data dari server.
2. POST: Mengirimkan data ke server untuk diproses (misalnya mengirimkan formulir).
3. PUT: Mengirimkan data ke server untuk diperbarui.
4. DELETE: Menghapus data dari server.
5. OPTIONS: Mendapatkan informasi tentang kemampuan server.
6. HEAD: Mengambil informasi header dari respons tanpa mengambil tubuh respons.

HTTP telah menjadi dasar bagi komunikasi web dan merupakan protokol yang digunakan oleh web browser, web server, API REST, dan banyak aplikasi yang berinteraksi melalui jaringan.

## **GOOGLE API**

Google Cloud Platform (GCP) adalah kumpulan layanan komputasi awan (*cloud computing)* yang disediakan oleh Google. Layanan yang disediakan GCP ini diantaranya adalah Google Cloud Console dan Google API (Google LLC, t.thn.).

Apa itu Google API? Untuk menjelaskannya dengan bahasa yang mudah dipahami, Google API atau nama aslinya Google Cloud API adalah kumpulan alat dan layanan yang disediakan oleh Google untuk para pengembang (Google LLC, t.thn.). Dengan menggunakan Google API, kita dapat mengakses berbagai fitur dan data yang disediakan oleh layanan Google, seperti Google Maps, Gmail, Google Drive, dan masih banyak lagi.

Sekarang, mari kita lihat beberapa contoh penggunaan Google API. Misalnya, kalian ingin mengintegrasikan peta interaktif dari Google Maps ke dalam aplikasi kalian. Dengan menggunakan Google Maps API, kalian dapat menampilkan peta yang interaktif, menambahkan penanda, dan bahkan mendapatkan rute tercepat antara dua titik. Ini sangat berguna jika kalian ingin membuat aplikasi untuk mencari lokasi atau menunjukkan rute perjalanan kepada pengguna.

Selain itu, Google API juga dapat digunakan untuk mengakses data dari Gmail. Misalnya, kalian ingin mengembangkan aplikasi yang dapat membaca dan memproses email pengguna. Dengan menggunakan Gmail API, kalian dapat mengambil email, mengirim pesan, dan bahkan mengatur label pada email. Ini membuka peluang yang menarik untuk mengembangkan alat-alat yang berguna dalam mengelola email.

Untuk menggunakan Google API, kalian perlu mendaftar dan mendapatkan kunci API. Kunci API ini digunakan untuk mengidentifikasi aplikasi kalian saat berkomunikasi dengan layanan Google. Jangan khawatir, proses pendaftaran dan mendapatkan kunci API cukup sederhana dan tersedia dokumentasi resmi yang dapat membantu kalian.

Namun, sebelum mulai menggunakan Google API, penting bagi kalian untuk memahami dokumentasi yang disediakan oleh Google. Dokumentasi ini akan memberikan petunjuk tentang cara menggunakan API dengan benar, termasuk permintaan dan respon yang diharapkan. Jangan ragu untuk membaca dan mempelajari dokumentasi tersebut secara seksama agar kalian dapat memanfaatkan fitur-fitur Google API dengan baik.

Teman-teman, Google API adalah alat yang amat berguna dalam pengembangan aplikasi. Dengan menggunakan Google API, kalian dapat mengintegrasikan fitur-fitur canggih yang disediakan oleh Google ke dalam aplikasi kalian sendiri. Jangan takut untuk bereksperimen dan menjelajahi berbagai layanan yang disediakan oleh Google API

## **TEXT EDITOR**

Text Editor merupakan sebuah perangkat lunak yang dipakai untuk membuat, mengedit, dan memformat teks. Ini adalah alat yang begitu penting dalam pengembangan perangkat lunak dan pemrograman. Text Editor membantu kita dalam menulis kode program dengan cara yang efisien dan efektif.

Dalam dunia pemrograman, kita menggunakan Text Editor untuk membuat berbagai jenis file seperti file sumber kode (source code) dalam berbagai bahasa pemrograman, file konfigurasi, atau file teks biasa. Text Editor biasanya dilengkapi dengan fitur-fitur yang membantu pemrogram dalam menulis kode, seperti penyorotan sintaks, penyelesaian otomatis, pengelolaan proyek, dan banyak lagi. Ini memungkinkan kita untuk mengembangkan dan memelihara kode dengan lebih mudah.

Sekarang, mari kita perkenalkan dua Text Editor yang umum digunakan dalam pengembangan Golang, yaitu Go Playground dan Visual Studio Code.

Go Playground adalah alat online yang memungkinkan kita untuk menulis, menjalankan, dan berbagi kode Go secara langsung melalui web browser. Dengan Go Playground, kalian dapat mempraktikkan dan menguji kode Go tanpa perlu menginstal perangkat lunak tambahan. Ini adalah tempat yang bagus untuk pemula untuk mulai belajar dan eksperimen dengan Golang.

Selain itu, Visual Studio Code (VS Code) adalah Text Editor populer yang mendukung berbagai bahasa pemrograman, termasuk Golang. VS Code menawarkan banyak fitur yang berguna seperti penyorotan sintaks, penyelesaian otomatis, debugging, pengaturan ekstensi, dan integrasi dengan Git. Dengan antarmuka pengguna yang intuitif dan fleksibilitasnya yang tinggi, VS Code menjadi pilihan yang populer di kalangan pengembang.

Kedua Text Editor ini menawarkan kemudahan dalam menulis dan mengelola kode Golang. Masing-masing mempunyai kelebihan serta fitur yang berbeda. Kalian dapat memilih Text Editor yang paling sesuai dengan preferensi dan kebutuhan kalian.

Berikut adalah beberapa contoh text editor yang populer dan sering digunakan dalam pemrograman:

1. Visual Studio Code (VS Code): VS Code adalah text editor open source yang dikembangkan oleh Microsoft. Ia mendukung berbagai bahasa pemrograman dan menyediakan fitur-fitur seperti penyorotan sintaks, pemformatan kode, integrasi dengan Git, ekstensi yang dapat disesuaikan, dan banyak lagi. VS Code memiliki antarmuka yang intuitif dan ramah pengguna, membuatnya populer di kalangan programmer.
2. Sublime Text: Sublime Text adalah text editor dengan antarmuka yang elegan dan cepat. Ia menyediakan fitur-fitur seperti penyorotan sintaks, navigasi cepat, pemformatan kode, snippet, dan banyak lagi. Sublime Text juga dapat disesuaikan melalui paket ekstensi, yang memungkinkan pengguna untuk menyesuaikan pengalaman pemrograman sesuai kebutuhan.
3. Atom: Atom adalah text editor open source yang dikembangkan oleh GitHub. Ia didesain untuk kecepatan, keindahan, dan kustomisasi. Atom mendukung penyorotan sintaks, integrasi dengan Git, sistem manajemen paket, dan banyak lagi. Atom juga memiliki antarmuka yang ramah pengguna dan fleksibilitas yang tinggi dalam menyesuaikan pengaturan dan tema.
4. IntelliJ IDEA: IntelliJ IDEA adalah text editor yang kuat yang dikembangkan oleh JetBrains. Ia dirancang khusus untuk pengembangan perangkat lunak berbasis Java, tetapi juga mendukung bahasa pemrograman lainnya. IntelliJ IDEA menyediakan fitur-fitur lengkap seperti refactoring kode, analisis statis, debugging, pengembangan web, dan banyak lagi. Ia juga mendukung integrasi dengan berbagai alat dan kerangka kerja populer.
5. Notepad++: Notepad++ adalah text editor yang sederhana, ringan, dan mudah digunakan. Ia menyediakan fitur-fitur dasar seperti penyorotan sintaks, pencarian dan penggantian teks, pemformatan kode, dan lain-lain. Notepad++ juga dapat disesuaikan melalui pengaturan dan plugin yang tersedia.
6. Emacs: Emacs adalah text editor yang sangat fleksibel dan kuat. Ia dapat digunakan untuk pemrograman dalam berbagai bahasa dan menyediakan fitur-fitur seperti penyorotan sintaks, navigasi cepat, pengelolaan proyek, dan banyak lagi. Emacs juga dapat dikustomisasi secara ekstensif melalui konfigurasi dan skrip Lisp.

Itulah beberapa contoh text editor yang populer dan sering digunakan dalam pemrograman. Pilihan text editor tergantung pada preferensi pribadi dan kebutuhan pengembangan masing-masing programmer.

Dalam pembahasan kaliini kita akan menggunakan Teks editor yang bernama Visual Studio Code yang sudah saya sebutkan sebelumnya.Untuk menggunakan Visual Studio Code dan menggunakan bahasa pemrograman Golang atau Go-language,kita butuh untuk menginstal Ekstensi khusus Bahsa “Go” di dalam Visual Studio Code,yang mana Ekstensi (Ekstensions) adalah komponen tambahan yang dapat memperluas fungsionalitas dan kemampuan text editor tersebut. Ekstensi ini dikembangkan oleh komunitas atau pihak ketiga, dan dapat diinstal dan digunakan oleh pengguna Visual Studio Code secara gratis sesuai kebutuhan mereka.

Ekstensi di Visual Studio Code memberikan fitur tambahan, perbaikan, dan alat bantu yang dapat membantu meningkatkan produktivitas pengguna. Fitur yang ditawarkan oleh ekstensi sangat bervariasi, mulai dari penyorotan sintaks dan pemformatan kode, hingga integrasi dengan platform layanan web atau framework khusus.

Contoh beberapa jenis ekstensi yang populer di Visual Studio Code meliputi:

1. Penyorotan Sintaks: Ekstensi ini memberikan penyorotan sintaks yang lebih lengkap untuk bahasa pemrograman tertentu, sehingga memudahkan pengguna dalam membaca dan memahami kode.
2. Pemformatan Kode: Ekstensi ini membantu dalam memformat kode secara otomatis sesuai dengan aturan atau gaya penulisan yang diinginkan.
3. Snippet: Ekstensi ini menyediakan potongan kode yang dapat dengan cepat dimasukkan ke dalam editor, membantu pengguna dalam menulis kode dengan cepat dan mengurangi kesalahan penulisan.
4. Integrasi dengan Alat Eksternal: Ekstensi ini memungkinkan Visual Studio Code berintegrasi dengan alat atau platform lain seperti Git, Docker, atau sistem manajemen basis data, sehingga mempermudah pengguna dalam mengelola proyek dan alur kerja.
5. Tema Visual: Ekstensi ini memberikan pilihan tema visual yang berbeda, sehingga pengguna dapat menyesuaikan tampilan Visual Studio Code sesuai dengan preferensi mereka.
6. Debugger: Ekstensi ini menyediakan dukungan debugging untuk bahasa pemrograman tertentu, memungkinkan pengguna untuk melacak dan memperbaiki kesalahan dalam kode mereka.
7. Ekstensi Khusus Framework atau Platform: Terdapat juga ekstensi yang dikembangkan khusus untuk kerangka kerja atau platform tertentu, seperti ekstensi untuk pengembangan aplikasi web dengan JavaScript, CSS, atau ekstensi untuk kerangka kerja seperti React, Angular, atau Vue.js.

Dengan menggunakan ekstensi di Visual Studio Code, pengguna dapat mengkustomisasi dan meningkatkan pengalaman mereka dalam mengedit kode, meningkatkan produktivitas, dan mendukung pengembangan di berbagai bahasa pemrograman dan lingkungan pengembangan yang berbeda. Pengguna dapat menginstal ekstensi melalui Marketplace yang tersedia di dalam Visual Studio Code, dan mengaktifkannya sesuai kebutuhan mereka.

# **BAB 2**

# **PEMASANGAN APLIKASI**



## **OVERVIEW**

Pada bab ini, kita akan membahas tentang pemasangan aplikasi. Pemasangan aplikasi merupakan langkah penting dalam menghadirkan perangkat lunak ke dalam lingkungan komputasi. Melalui bab ini, kita akan mempelajari proses instalasi aplikasi dengan langkah-langkah yang jelas dan panduan yang mudah dipahami.

Pemasangan aplikasi melibatkan beberapa tahapan, mulai dari pengunduhan (download) aplikasi hingga penyelesaian instalasi. Kita akan mempelajari berbagai metode pemasangan, termasuk penggunaan installer, manajer paket, dan penggunaan repositori perangkat lunak.

Selain itu, bab ini juga akan memberikan penekanan pada pentingnya mengikuti petunjuk instalasi yang disediakan oleh pengembang aplikasi. Ini termasuk persyaratan sistem, kompatibilitas, dan langkah-langkah khusus yang mungkin perlu diikuti.

Dalam pembahasan ini, kita akan melihat contoh pemasangan beberapa aplikasi populer pada platform yang umum digunakan, seperti Windows, macOS, dan Linux. Selain itu, akan diberikan pula petunjuk umum yang dapat digunakan pada berbagai platform.

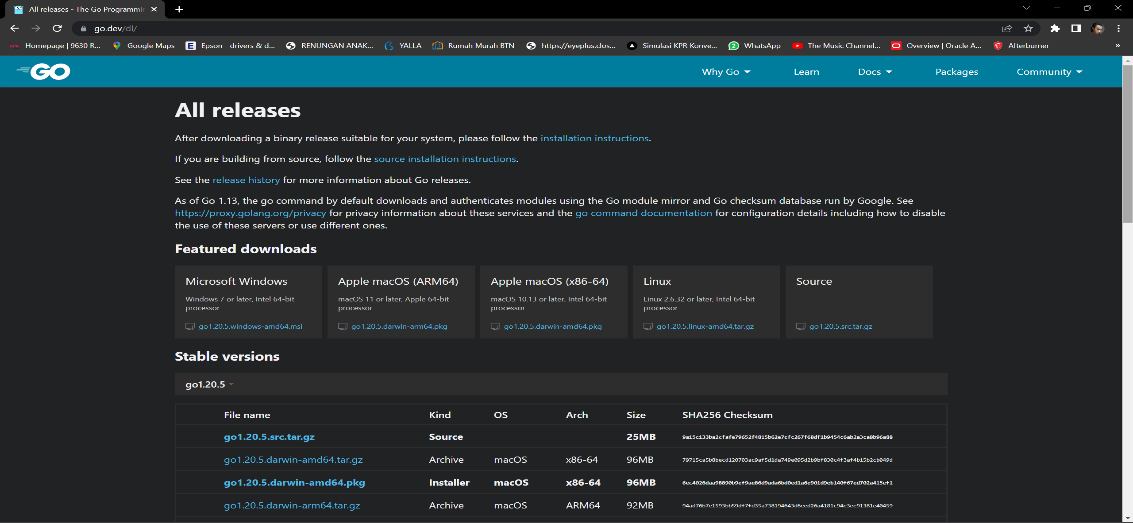
Menginstal aplikasi dengan benar adalah langkah awal yang krusial dalam memastikan aplikasi berfungsi dengan baik dan aman di sistem yang dituju. Oleh karena itu, pemahaman tentang proses pemasangan aplikasi yang efektif dan efisien sangat penting untuk seorang pengembang perangkat lunak.

Dalam bab ini, kita akan memberikan panduan yang komprehensif dan mudah dipahami, serta memberikan contoh nyata untuk membantu memahami langkah-langkah yang terlibat dalam pemasangan aplikasi. Dengan pemahaman yang baik tentang proses ini.

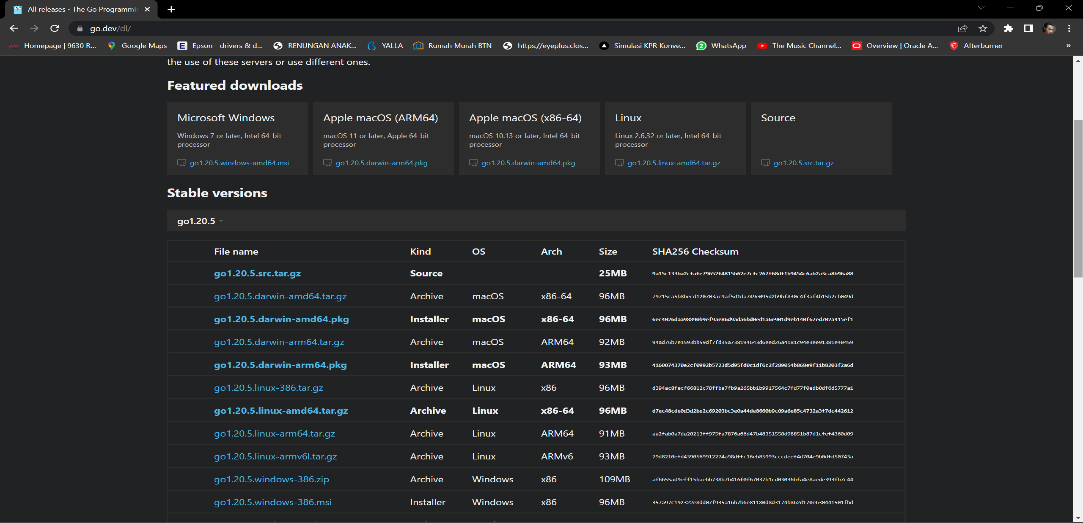
## **PEMASANGAN GO**

Berikut adalah langkah-langkah pemasangan GO:

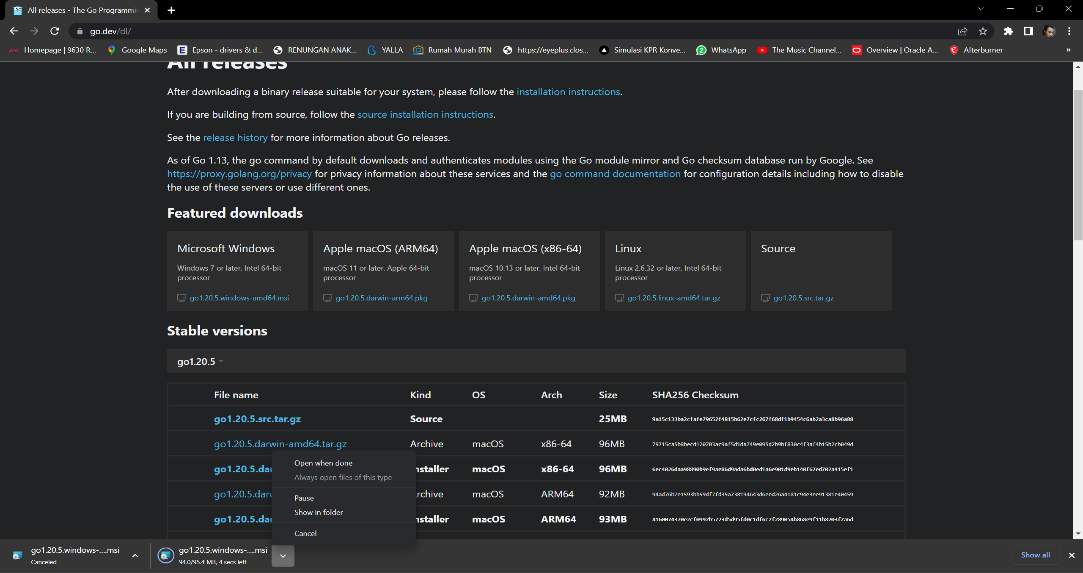
1. Kunjungi situs resmi Golang di https://go.dev/dl/



1. Di halaman unduhan, kalian akan ada opsi untuk memilih versi Golang yang ingin diunduh. Pilih versi yang sesuai dengan sistem operasi yang kalian gunakan. Golang tersedia untuk Windows, macOS, dan Linux.



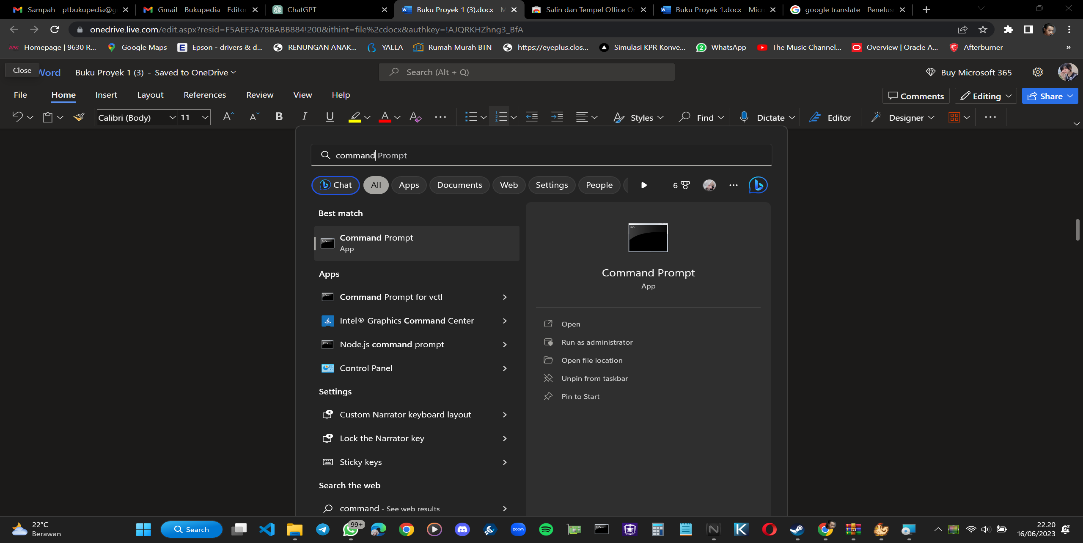
1. Pastikan versi go yang akan di install adalah versi terbaru.
2. Setelah memilih versi yang sesuai, klik tautan unduhan yang sesuai dengan sistem operasi kalian,dalam kasus ini saya menggunakan windows,
3. File instalasi Golang akan mulai diunduh. Tunggu hingga proses unduhan selesai.



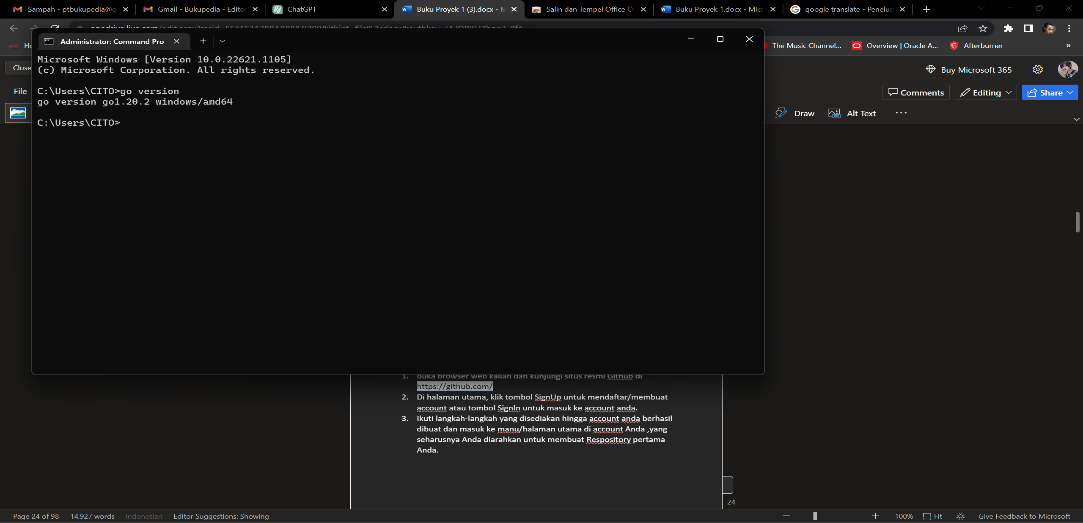
1. Setelah proses unduhan selesai, buka file instalasi yang telah diunduh.



1. Ikuti petunjuk instalasi yang diberikan pada layar. Pastikan untuk memilih opsi yang sesuai dengan preferensi kalian. Sebagian besar pengaturan default akan bekerja dengan baik.
2. Setelah proses instalasi selesai, buka terminal atau command prompt pada sistem operasi kalian.



1. Ketik perintah "go version" untuk memastikan bahwa Golang telah terinstal dengan benar. Jika versi Golang muncul di layar, maka instalasi telah berhasil.

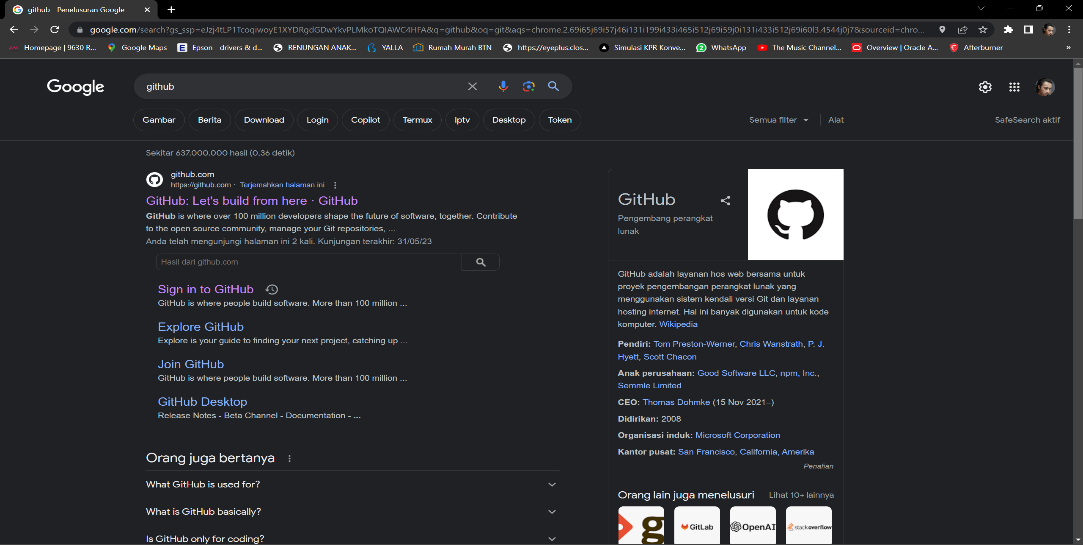


1. GO sudah berhasil terinstall.

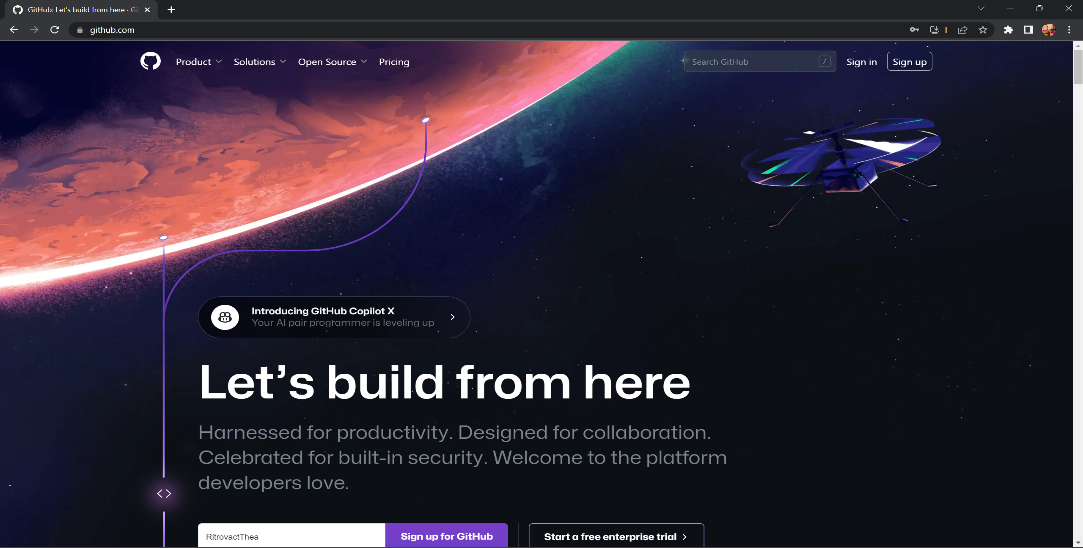
## **PENGGUNAAN GITHUB**

Berikut adalah Langkah-langkah penggunaan Github”:

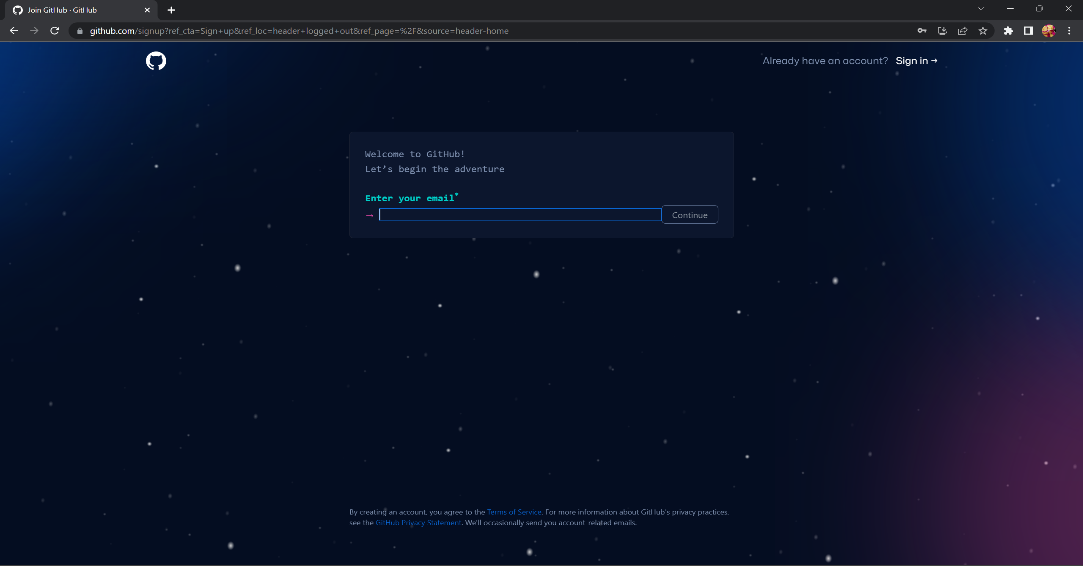
1. Buka browser web kalian dan kunjungi situs resmi Github di <https://github.com/>



1. Di halaman utama, klik tombol SignUp untuk mendaftar/membuat account atau tombol SignIn untuk masuk ke account anda.



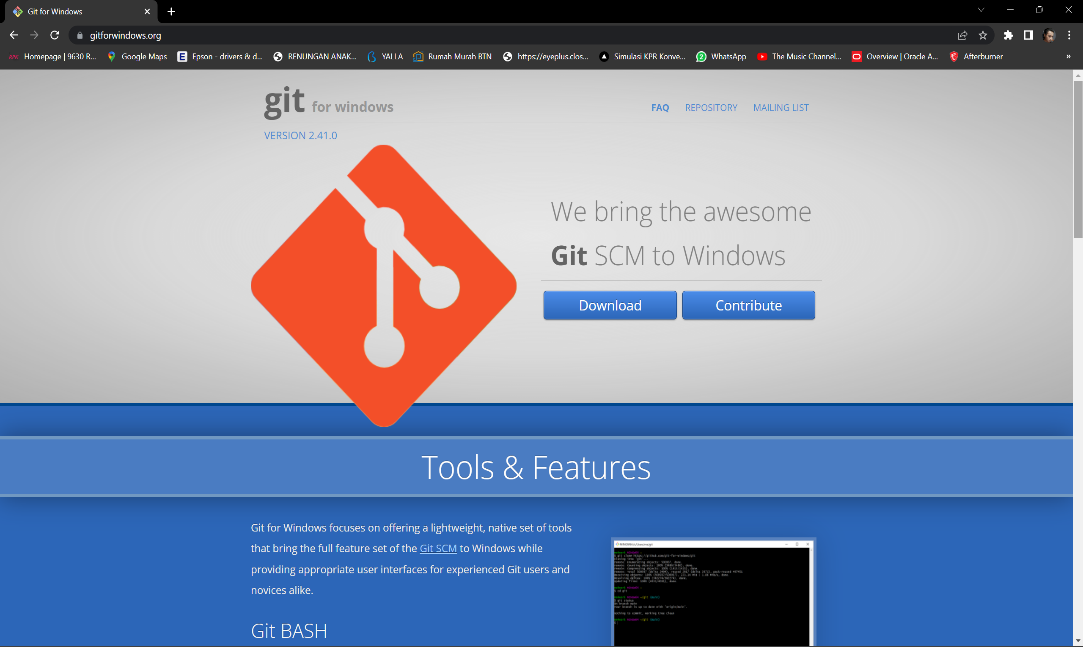
1. Ikuti langkah-langkah yang disediakan hingga account anda berhasil dibuat dan masuk ke manu/halaman utama di account Anda ,yang seharusnya Anda diarahkan untuk membuat Respository pertama Anda.



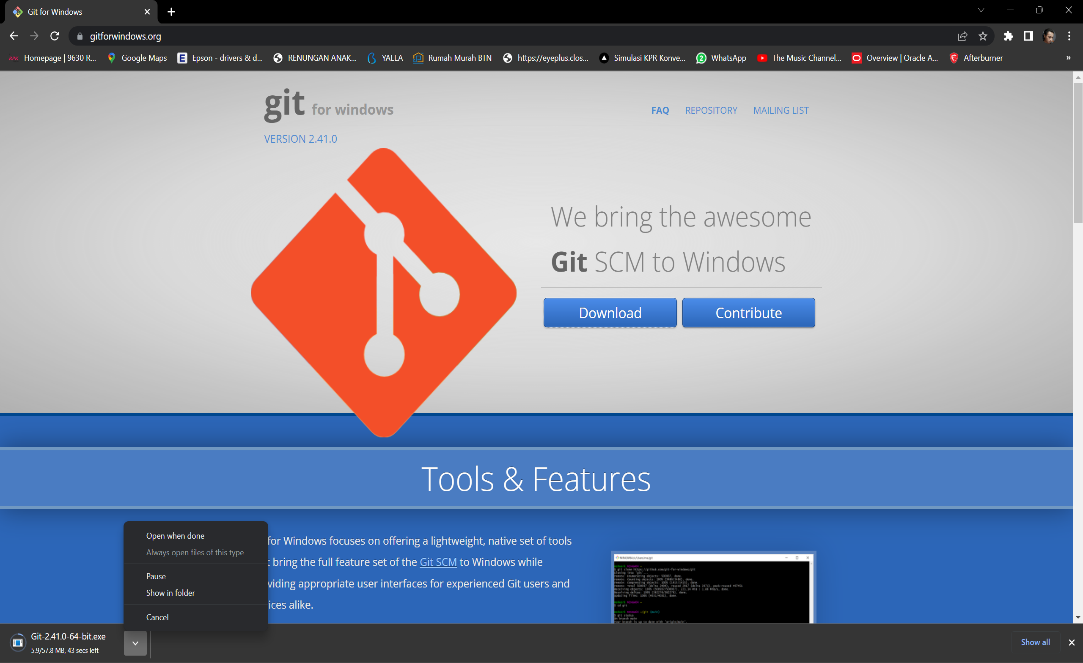
## **PEMASANGAN GITBASH**

Berikut adalah Langkah-langkah penginstalan GitBash di Windows 10:

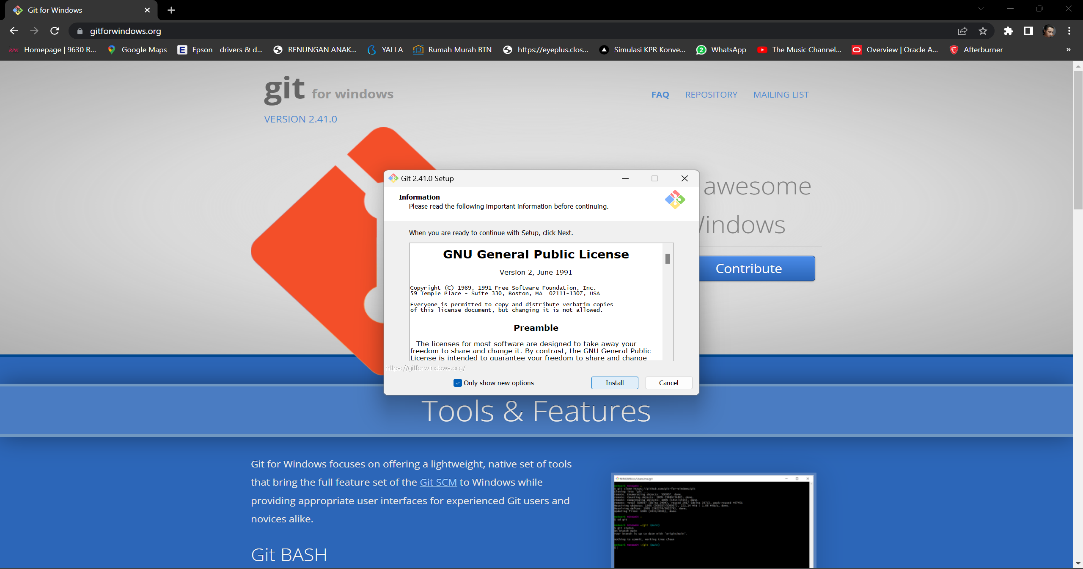
1. Buka browser web kalian dan kunjungi situs resmi Git untuk Windows di <https://gitforwindows.org>.



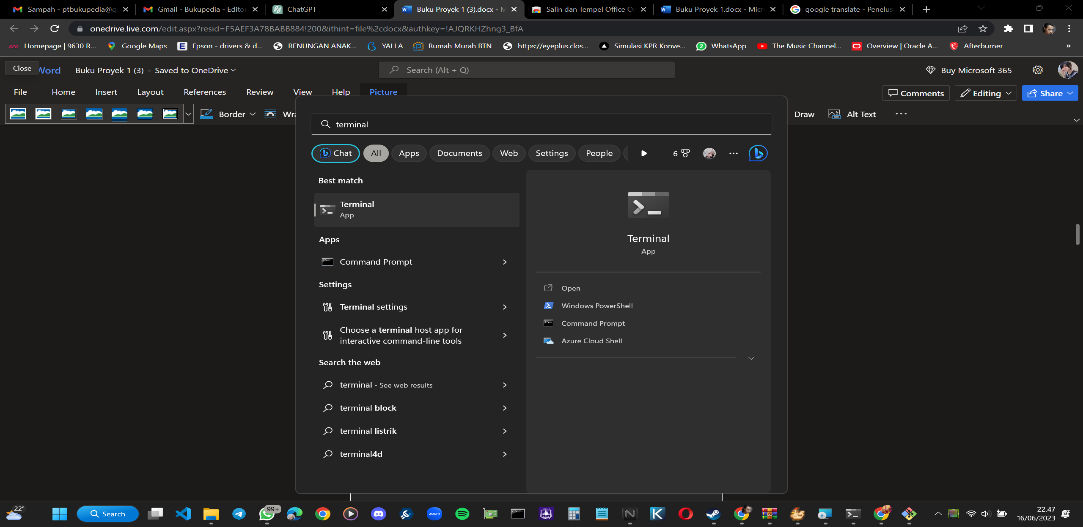
1. Di halaman utama, klik tombol "Download" untuk mengunduh file instalasi Git.



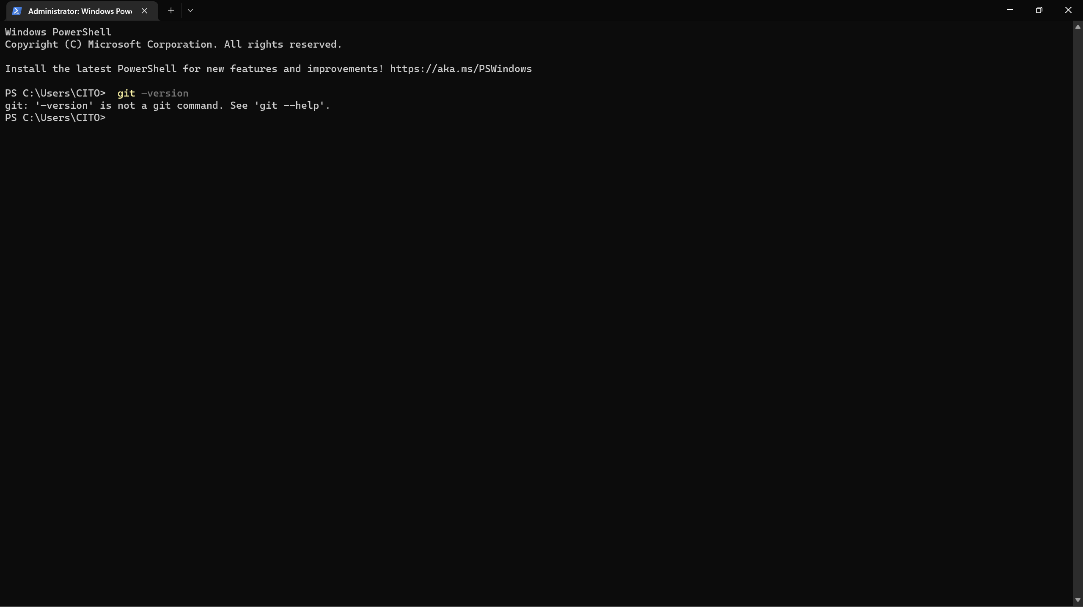
1. Setelah file instalasi terunduh, buka file tersebut dan ikuti petunjuk instalasi yang diberikan pada layar. Perhatikan pengaturan yang disarankan, kecuali jika kalian memiliki kebutuhan khusus.



1. Selama proses instalasi, kalian akan disajikan beberapa pilihan konfigurasi. Jika kalian tidak yakin, biarkan pengaturan default dan klik "Next" atau "Continue" untuk melanjutkan instalasi.
2. Setelah instalasi selesai, kalian dapat mencari aplikasi "Git Bash" di komputer kalian dan membukanya. Ini akan membuka jendela terminal yang menggunakan antarmuka baris perintah Git.



1. Untuk memastikan Git telah terinstal dengan benar, kalian dapat membuka terminal dan mengetikkan perintah berikut untuk melihat versi Git: git –version

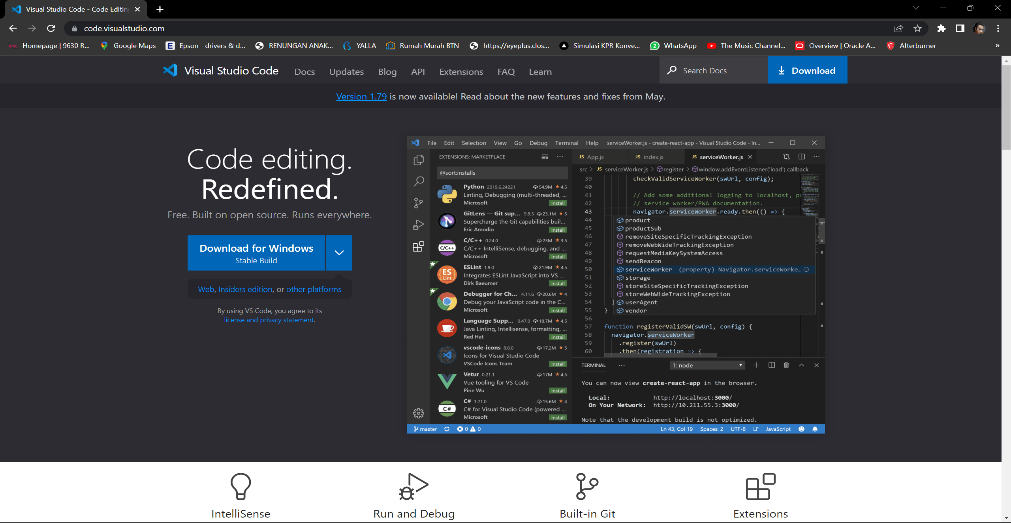


1. Jika versi Git ditampilkan, berarti instalasi berhasil dan kalian siap menggunakan Git melalui Git Bash.

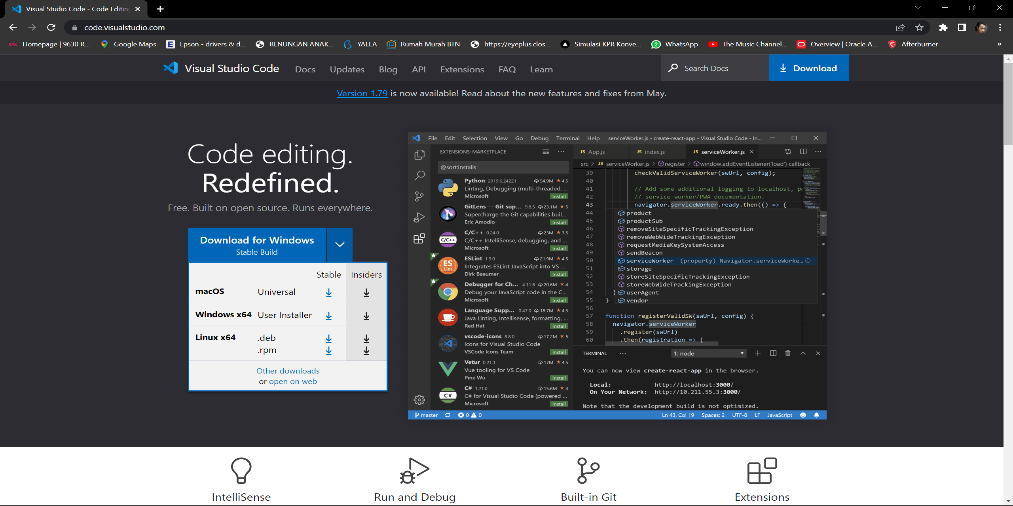
## **PEMASANGAN VISUAL STUDIO CODE**

Berikut adalah Langkah-langkah penginstalan GitBash di Windows 10:

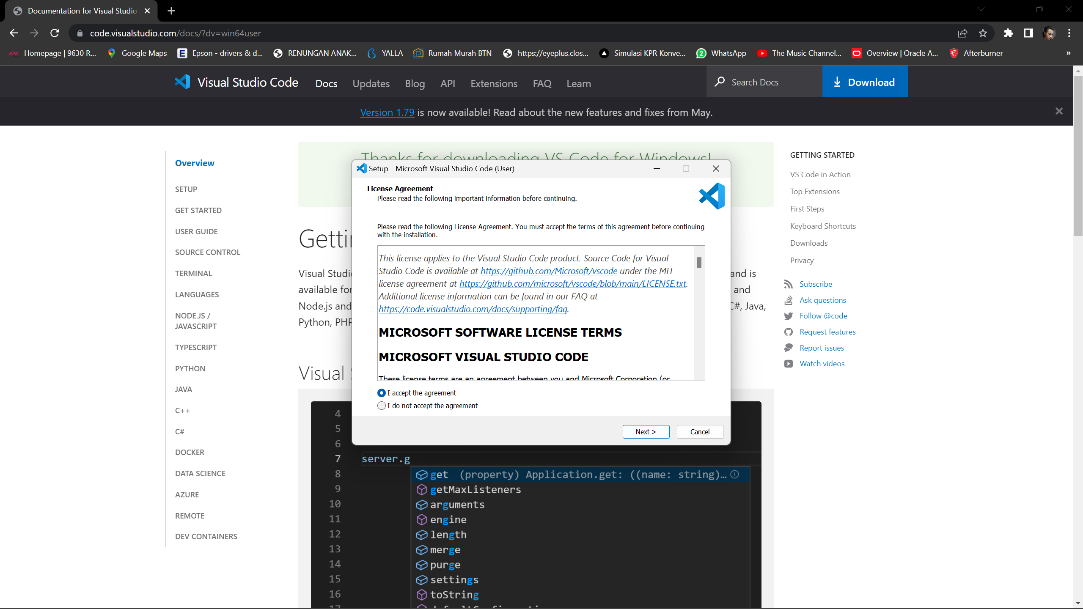
1. Buka browser web kalian dan kunjungi situs resmi Visual Studio Code di <https://code.visualstudio.com>.



1. Di halaman utama, kalian akan melihat tombol "Download for [sistem operasi]" sesuai dengan sistem operasi yang kalian gunakan (Windows, macOS, atau Linux). Klik tombol tersebut untuk mengunduh file instalasi Visual Studio Code.



1. Setelah file instalasi terunduh, buka file tersebut dan ikuti petunjuk instalasi yang diberikan pada layar. Perhatikan pengaturan yang disarankan, kecuali jika kalian memiliki kebutuhan khusus.

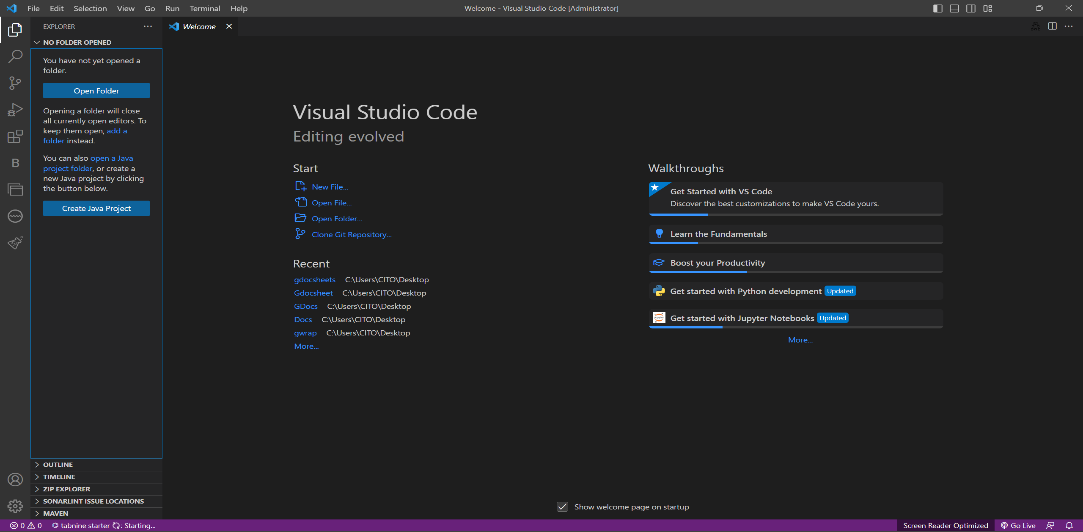


1. Selama proses instalasi, kalian akan disajikan beberapa pilihan konfigurasi. Jika kalian tidak yakin, biarkan pengaturan default dan klik "Next" atau "Continue" untuk melanjutkan instalasi.
2. Setelah instalasi selesai, buka Visual Studio Code dari menu atau melalui ikon yang muncul di desktop kalian.
3. Visual Studio Code akan membuka dengan tampilan awal yang sederhana. Kalian dapat mulai menggunakannya untuk mengedit dan mengelola file kode.
4. Visual Studio Code juga menyediakan fitur ekstensi yang dapat memperluas fungsionalitasnya. Kalian dapat menginstal ekstensi tambahan sesuai dengan kebutuhan kalian. Untuk melakukannya, klik pada ikon ekstensi di sebelah kiri, cari ekstensi yang kalian inginkan, dan klik "Install" untuk menginstalnya.
5. Setelah ekstensi terinstal, kalian dapat mulai menggunakan Visual Studio Code untuk pengembangan perangkat lunak. Dengan antarmuka pengguna yang intuitif dan fitur-fitur yang kuat, Visual Studio Code dapat membantu kalian dalam menulis, mengedit, dan mengelola kode dengan lebih efisien.

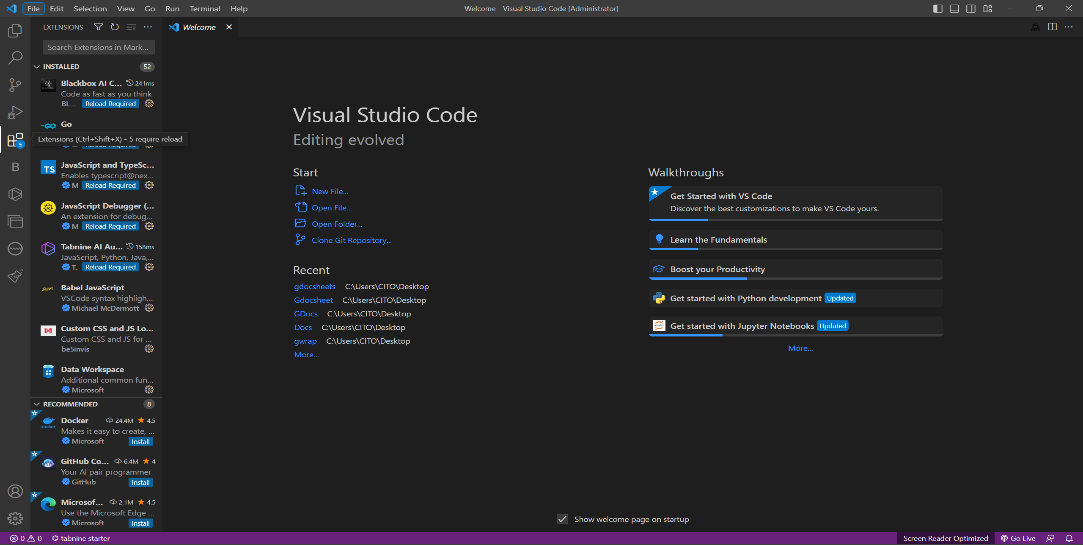
## **PEMASANGAN EKSTENSI GO-LANGUAGE DI VISUAL STUDIO CODE**

Agar kita dapat menggunakan Bahasa pemrograman golang di Visual studio code,kita perlu menambahkan Ekstensi berikut adalah caranya:

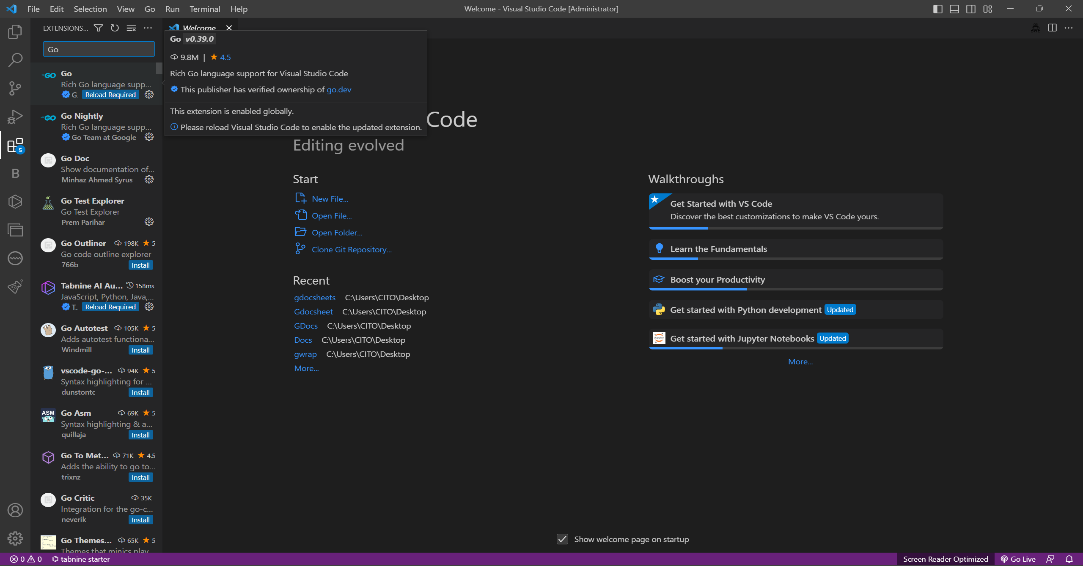
1. Buka Visual Studio Code yang sudah di Install sebelumnya.



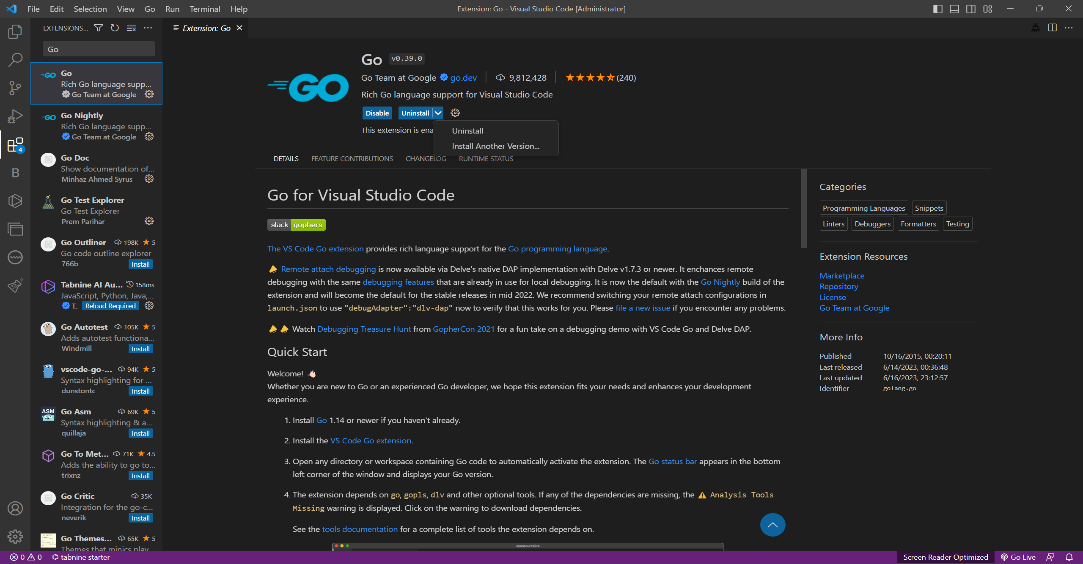
1. Di menu bar bagian kiri,tekan simbol yang berbentuk 4 persegi yang bernamakan Extensions.



1. Lalu silahkan ketik kata kunci “Go” di bagian searchbar/menu pencarian.



1. Setelah itu klik Extensi yang bernama ”Go” untuk membukanya.



1. Setelah menu Ekstensi ”Go” terbuka cari dan tekan tombol Install,tunggu hingga penginstalan selesai.

Jika Anda menggunakan Teks editor lainnya,Anda dapat mencari tombol Ekstensi yang tersedia.

# **BAB 3**

# **PEMROGRAMAN GO**



Untuk memahami pemrograman go, perlu banyak hal yang perlu diketahui oleh kita. Diantaranya adalah konsep dasar pemrograman,Pada bab ini, kita akan memahami konsep-konsep dasar dalam bahasa pemrograman Golang dengan cara yang mudah dipahami.

## **KONSEP DASAR PEMROGRAMAN**

Sebelum mempelajari bahasa pemrograman, kita harus mengetahui konsep dasar pemorgaman. Konsep ini menggambarkan bagaimana sebuah program berjalan. Konsep ini adalah hal fundamental sebelum mempelajari bahasa pemrograman tertentu.

### Algoritma

Dalam dunia matematika ada seseorang yang dikenal sebagai bapak dari aljabar dan algoritma. Beliau adalah Al-Khawarizmi. Nama lengkapnya adalah Abu Abdullah muhammad Ibn Musa Al-Khawarizmi. Beliau dikenal sebagai matematikawan yang memperkenalkan aljabar. Akan tetapi selain aljabar, beliau juga memperkenalkan konsep Algoritma. Para sarjana Eropa mengenal Al-Khawarizmi sebagai Algoritm, Algorismus, atau Algoritma (Nandy, t.thn.).

Algoritma sendiri adalah langkah-langkah logis dalam memecahkan masalah yang terstruktur secara sistematis (Nandy, t.thn.). Contoh dalam kehidupan kita sehari-hari secara sederhana adalah proses memasak mi. Untuk memakan mi kita harus mengikuti langkah-langkah yang tertera dalam kemasan mi, seperti memanaskan air terlebih dahulu, lalu masukkan mi ke dalam air yang sudah dipanaskan, kemudian ditiriskan, lalu diberi bumbu, barulah mi bisa dimakan. Proses logis dalam memasak mi tersebut merupakan contoh sederhana dari algoritma. Algoritma juga berlaku pada pemrograman, karena komputer berjalan sesuai dengan urutan perintah. Untuk memahami lebih dalam tentang algoritma, kalian bisa mencari referensi di internet maupun di buku cetak.

### Variabel, konstanta dan Tipe Data

Variabel dan tipe data merupakan 2 hal yang saling beriringan, dimana suatu variabel pasti mempunyai 1 tipe data. Variabel adalah tempat menampung suatu data yang nilainya dapat berubah sesuai dengan data yang diberikan (Roni, 2021). Data adalah kumpulan fakta berupa huruf, angka, suara, ataupun gambar. Dalam pemrograman, umumnya data diklasifikasikan ke dalam 6 tipe umum. Data yang diklasifikasi inilah yang disebut tipe data. Adapun 6 tipe data yang umum adalah:

1. Bilangan bulat (*integer*). Contoh: 1, 2, 10, 100
2. Bilangan pecahan (float). Contoh: 1.1, 2.5, 10.5
3. Karakter (char). Contoh: “a”,”c”,”1”,”&”
4. Boolean. Contoh: True, False
5. String. Contoh: “aku”,”kamu”,”dia”
6. Himpunan(Array): {1,2,3}, {“aku”,”kamu”,”dia”}

Contoh variabel dan tipe data:

|  |
| --- |
| Var x int = 20 // Artinya: Variabel x adalah bilangan bulat yang bernilai 20 |

Selain variabel ada juga konstanta, yang hampir mirip dengan variabel, hanya saja data yang dimasukkan ke dalam konstanta tidak bisa kita ubah.

### Struktur Kontrol.

Program komputer dijalankan secara berurutan, dari atas ke bawah, dari kiri ke kanan, seperti halnya kita membaca. Namun adakalanya suatu program perlu membuat keputusan, seperti menjalankan kode lain, atau memulai program dari awal. Keputusan yang mempengaruhi alur program ini disebut Struktur kontrol (Yse, 2020).

Pada dasarnya struktur kontrol dibagi menjadi 2:

1. Kondisional: untuk menjalankan suatu perintah jika suatu kondisi terpenuhi
2. Perulangan: untuk menjalankan perintah berulang kali selama suatu kondisi terpenuhi.

Dalam dunia pemrograman kedua hal tersebut sering dikenal dengan istilah if-else dan for-while.

Contoh Struktur Kontrol dalam pemrograman.

|  |
| --- |
| Var x int = 19  If(x >= 20){  fmt.Println(x)  }else{  fmt.Println(“nilai kurang dari 20”)  } |

Dalam kode tersebut, ada variabel x yang merupakan bilangan bulat dengan nilai 19. Jika variabel x lebih dari atau sama dengan 20, maka program harus menampilkan nilai dari variabel x. Jika tidak, maka program harus menampilkan “nilai kurang dari 20”.

### Fungsi dan Metode

Fungsi itu seperti resep. Jika kita memesan burger, sebenarnya kita memerintahkan untuk melakukan serangkaian kegiatan untuk membuat burger. Seperti menambahkan roti, memasak daging, memasukkan daging ke dalam roti, menambahkan saus, dan menutupnya dengan roti lagi. Kemudian pelayan akan mengantarkan kepada kita hasil jadi dari pembuatan Burger tadi. Dari analogi ini, fungsi adalah kumpulan instruksi untuk melakukan suatu tugas (Codecademy Team, t.thn.).

Sama halnya dengan fungsi method juga merupakan kumpulan instruksi untuk melakukan suatu tugas, hanya saja method terikat pada objek. Objek merujuk pada instansi dari suatu class. Karena terkait dengan objek, method biasanya ada dalam pemrograman berorientasi objek (OOP).

Contoh fungsi dalam bahasa pemrograman PHP:

|  |
| --- |
| function salam() {  echo "Halo, selamat pagi!";  }  salam(); // Output: Halo, selamat pagi! |

### Struktur Data

Struktur data merupakan suatu format khusus yang digunakan untuk mengatur dan menyimpan data dengan cara yang terstruktur. Fungsinya adalah memudahkan pengguna dalam mengakses dan memanipulasi data sesuai kebutuhan. Kita bisa mengibaratkan struktur data seperti cara kita mengatur dan menyimpan barang-barang dalam kehidupan sehari-hari, hanya saja struktur data digunakan untuk mengatur dan menyimpan data di dalam komputer. Terdapat beberapa jenis struktur data yang digunakan, antara lain:Array,List,Tree,Queue,Stack

### Input dan Output

Agar pengguna dapat berinteraksi dengan program, mereka perlu memberikan input data. Input ini biasanya dilakukan melalui keyboard dan mouse, atau menggunakan pengontrol permainan dalam kasus game. Sebagai contoh, dalam sebuah game, karakter dapat bergerak maju atau mundur dengan menggunakan tombol "w" dan "a" pada keyboard. Ketika karakter bergerak maju, game tersebut juga memberikan output berupa gambar animasi yang menunjukkan karakter sedang bergerak maju.

Dalam pemrograman, kita perlu memasukkan data yang diperlukan ke dalam kode program. Hal ini bisa dilakukan langsung dalam kode program atau melalui input yang diberikan saat program sedang berjalan. Output dari program dapat bervariasi tergantung pada pembuat programnya. Input dan output dalam program memiliki peran penting sebagai bentuk interaksi antara komputer dan manusia.

Berikut ini adalah contoh penggunaan input dan output dalam bahasa Python:

|  |
| --- |
| print("What is your name?")  name = input()  print(f"Hello {name}") |

Program tersebut memiliki fungsionalitas yang menerima satu input berupa variabel yang disebut "name" dan menghasilkan dua output. Output pertama adalah pertanyaan "what is your name?", sedangkan output kedua adalah pesan "Hello (variabel name)". Dengan demikian, program ini berinteraksi dengan pengguna untuk mendapatkan nama mereka dan memberikan salam menggunakan nama tersebut.

### Paradigma Pemrograman

Paradigma Pemrograman merupakan pendekatan yang digunakan dalam memecahkan masalah dengan menggunakan bahasa pemrograman tertentu. Hal ini disebabkan oleh kenyataan bahwa setiap bahasa pemrograman memiliki kecocokan yang berbeda dalam menyelesaikan masalah yang ada dalam suatu program.

Terdapat dua tipe paradigma dalam bahasa pemrograman, yaitu:

1. Paradigma Imperatif: Paradigma ini berfokus pada urutan instruksi yang harus dieksekusi untuk mencapai tujuan yang diinginkan. Beberapa contoh bahasa pemrograman yang menggunakan paradigma ini adalah C, C++, dan Java. Dalam paradigma ini, program akan menjalankan instruksi satu per satu sesuai dengan urutannya.
2. Paradigma Deklaratif: Paradigma ini lebih berfokus pada deskripsi mengenai apa yang ingin dicapai oleh program, bukan bagaimana program tersebut dijalankan. Contohnya adalah bahasa pemrograman Prolog, SQL, dan Haskell. Dalam paradigma ini, program mendeskripsikan pernyataan atau aturan yang kemudian akan dijalankan sesuai dengan kebutuhan.

Dengan adanya dua tipe paradigma ini, kita dapat memilih bahasa pemrograman yang paling sesuai untuk menyelesaikan masalah yang dihadapi dalam suatu program.

## **MEMBUAT PROGRAM “HELLO WORD” DI GO**

Kita akan memulai dengan membuat program pertama yang sederhana yaitu menampilkan tulisan "Hello World". Terdapat beberapa langkah yang perlu kita lakukan, di antaranya adalah inisialisasi proyek, pembuatan program, dan menjalankan program tersebut.

### Inisialisasi Proyek

Untuk memulai, kita perlu melakukan inisialisasi dengan membuat folder terlebih dahulu. Di sini, kami akan menunjukkan contoh pembuatan folder di desktop. Terdapat dua metode yang dapat digunakan untuk membuat folder tersebut, yaitu melalui antarmuka baris perintah atau menggunakan fitur "New" pada Windows. Untuk menggunakan fitur "New" pada Windows, Anda dapat mengikuti langkah-langkah berikut:

1. Klik kanan pada area kosong di desktop.
2. Pilih opsi "New" atau "Baru" (jika menggunakan bahasa Indonesia).
3. Selanjutnya, pilih opsi "Folder".
4. Berikan nama "hello-world" pada folder yang baru dibuat.

Alternatifnya, jika Anda ingin menggunakan baris perintah, berikut adalah langkah-langkahnya:

1. Buka folder desktop.
2. Tekan dan tahan tombol "Shift" pada keyboard, lalu klik kanan pada area kosong di dalam folder desktop.
3. Pilih opsi "Open terminal" (untuk Windows 11) atau "Command window here" (untuk Windows 10).

Setelah jendela terminal terbuka, masukkan perintah berikut:

|  |
| --- |
| mkdir hello-world  cd hello-world  go mod init hello-world  code . |

Kemudian teks editor VSCode akan terbuka

### Membuat file program dan menjalankan hello world

Untuk memulai pembuatan program menggunakan bahasa Go, langkah pertama adalah membuat sebuah file baru dengan nama "main.go" dan membukanya. Selanjutnya, Anda dapat menuliskan kode program berikut ke dalam file tersebut

|  |
| --- |
| package main  import "fmt"  func main(){  fmt.Println("Hello World")} |

Setelah Anda menulis kode tersebut, Anda dapat menjalankannya dengan mengetik perintah "go run main.go" di terminal atau command prompt. Perintah ini akan menjalankan program Go dan menghasilkan output yang sesuai.

Dalam kode yang Anda berikan, terdapat beberapa kata kunci yang perlu dipahami:

1. Package: Keyword "package" digunakan untuk menginisialisasi suatu package atau modul dalam bahasa Go. Pada contoh kode tersebut, nama package yang digunakan adalah "main".
2. Import: Keyword "import" digunakan untuk memasukkan package lain ke dalam file program, sehingga isi package yang di-import dapat digunakan. Pada contoh kode tersebut, package yang di-import adalah "fmt".
3. fmt.Println: Ini adalah contoh penggunaan fungsi "Println" dari package "fmt". Fungsi ini digunakan untuk mencetak atau menampilkan teks "Hello World" pada output program.
4. func main(): Fungsi "main" adalah fungsi khusus yang dieksekusi pertama kali saat program dijalankan. Semua program Go harus memiliki fungsi "main" yang merupakan titik awal eksekusi program.

Dengan menjalankan perintah "go run main.go", program akan mengeksekusi fungsi "main" dan mencetak output "Hello World" ke layar.

## **MEMAHAMI STRUKTUR SERTA KETENTUAN PENULISAN GOLANG**

Setiap bahasa pemrograman memiliki struktur dan aturan penulisan yang berbeda, baik dalam hal sintaks dasar maupun sintaks yang lebih kompleks. Memahami sintaksis dalam bahasa pemrograman sangat penting agar program dapat berjalan dengan lancar dan mengurangi kemungkinan terjadinya kesalahan.

Ketika baru mempelajari bahasa pemrograman, sering kali muncul pertanyaan tentang makna dari kode program yang ditulis. Hal ini sangat wajar karena memahami arti dan tujuan dari setiap baris kode adalah langkah awal yang penting dalam menguasai bahasa pemrograman tersebut.

Untuk mengatasi kebingungan tentang arti kode program, ada beberapa sumber yang dapat digunakan, seperti:

1. Dokumentasi resmi: Merujuk pada dokumentasi resmi bahasa pemrograman yang Anda pelajari. Dokumentasi ini menyediakan penjelasan rinci tentang setiap elemen sintaksis, fungsi, dan cara penggunaan yang benar.
2. Buku dan tutorial: Membaca buku panduan dan mengikuti tutorial online yang berfokus pada bahasa pemrograman yang dipelajari. Buku-buku ini biasanya memberikan penjelasan mendalam dan contoh penggunaan untuk membantu pemahaman konsep dan sintaksis bahasa pemrograman.
3. Forum dan komunitas: Bergabung dengan forum atau komunitas online yang terkait dengan bahasa pemrograman yang dipelajari. Di sana, Anda dapat bertanya langsung kepada pengembang berpengalaman dan mendapatkan jawaban serta penjelasan tentang kode program yang bingung.
4. Praktek dan eksperimen: Melakukan latihan dan eksperimen langsung dengan menulis kode program dan melihat hasilnya. Dengan mencoba sendiri, Anda dapat memahami secara praktis bagaimana kode program berinteraksi dan menghasilkan output yang diinginkan.

Mengerti arti kode program adalah proses yang berkelanjutan. Semakin sering Anda berlatih dan semakin banyak pengalaman yang Anda dapatkan, maka pemahaman tentang sintaksis dan arti kode program akan semakin meningkat.

|  |
| --- |
| package main import (  "fmt" ) func main(){  fmt.Println("Belajar Golang") } |

Dalam kode yang telah disebutkan, terdapat beberapa konsep yang perlu dipahami, antara lain:

1. Package: Package adalah konsep dalam bahasa pemrograman Golang yang digunakan untuk mengorganisir kode dalam unit terpisah. Setiap file Go harus dimulai dengan deklarasi paket menggunakan pernyataan package. Paket ini mencerminkan struktur folder dan bertindak sebagai namespace untuk kode yang ada dalam file tersebut.
2. Import: Import digunakan untuk mengimpor library atau paket yang akan digunakan dalam program. Pada kode yang telah disebutkan, baris 2-5 adalah pernyataan import yang mengimpor library fmt. Library ini menyediakan fungsi-fungsi untuk melakukan operasi cetak-mencetak dan membaca masukan pada program.
3. Main dan Penulisannya: Fungsi utama main merupakan fungsi khusus dalam bahasa pemrograman Golang yang dieksekusi pertama kali saat program dijalankan. Fungsi ini harus ada dalam paket dengan nama main. Penulisan main dengan huruf kecil adalah aturan penulisan yang diikuti dalam Golang. Jika fungsi utama ditulis dengan huruf kapital (Main), maka kompilator tidak akan mengenali fungsi tersebut sebagai fungsi utama.

Dengan memahami konsep-konsep tersebut, kita dapat memahami kerangka dasar dalam bahasa pemrograman Golang, yang terdiri dari deklarasi paket, import library, dan fungsi utama. Bagian-bagian tersebut sangat penting dalam struktur program Golang.

|  |
| --- |
| package main  import (  "fmt") func main(){  fmt.Println("Belajar Go!") } |

Berikut adalah penjelasan untuk setiap baris kode yang disebutkan:

1. Baris 1: Ini adalah deklarasi paket yang memberitahu bahwa file ini termasuk dalam paket dengan nama yang diberikan. Contohnya, package main menunjukkan bahwa ini adalah file untuk paket utama dalam program.
2. Baris 2 - 5: Ini adalah pernyataan impor (import) yang digunakan untuk memasukkan library fmt ke dalam program. Library ini menyediakan fungsi-fungsi untuk mencetak keluaran dan membaca masukan.
3. Baris 7: Ini adalah fungsi utama (main) yang akan dieksekusi pertama kali saat program dijalankan. Semua perintah yang ingin dieksekusi harus ditempatkan di dalam fungsi utama ini.

Dengan demikian, deklarasi paket, impor library, dan fungsi utama adalah komponen penting dalam struktur program Golang yang umum digunakan.

### Deklarasi Package

Package adalah cara untuk mengorganisir file atau dapat dianggap sebagai nama folder yang digunakan. Dalam bahasa pemrograman Golang, sebuah file dapat dieksekusi jika memiliki paket utama (package main).

Namun, bagaimana jika file tersebut berada di root proyek kita? Untuk mendeklarasikan paket ketika berada di proyek utama, kita menggunakan deklarasi paket dengan nama "main" (package main):

|  |
| --- |
| package main |

Jika proyek kita memiliki banyak modul dan kita ingin menjaga struktur program yang teratur dan mudah dipahami, kita dapat menggunakan folder untuk mengatur paket-paket dalam proyek tersebut. Misalnya, jika kita memiliki folder bernama "model" dalam proyek kita dan di dalam folder tersebut terdapat file bernama "posts.go", kita dapat menuliskan deklarasi paket sebagai berikut:

|  |
| --- |
| package model |

Apabila kita tidak menyertakan deklarasi paket yang diperlukan dalam program Golang, maka akan terjadi kesalahan (error) dan program tidak dapat dieksekusi. Dalam Golang, setiap file harus dimulai dengan pernyataan package yang menentukan nama paket yang digunakan dalam file tersebut. Tanpa deklarasi paket yang benar, kompilator akan menghasilkan kesalahan dan program tidak dapat dijalankan.

### Bagian Import

Import digunakan untuk memanggil library yang akan digunakan dalam kode program. Sebagai contoh pada kode di atas, kita menggunakan impor untuk memanggil library dengan nama fmt.

Apa itu library? Library adalah kumpulan fungsi-fungsi yang telah diproses secara terpisah dan siap digunakan dalam program. Dengan menggunakan library, kita dapat lebih mudah dalam membuat program dengan memanfaatkan fungsi-fungsi yang telah disediakan.

Di Golang, terdapat perbedaan dalam penggunaan impor library dibandingkan dengan bahasa pemrograman lainnya. Pada Golang, seringkali setiap perintah memerlukan impor library, bahkan untuk tugas yang sederhana seperti mencetak string.

Ada dua cara yang umum digunakan dalam menulis sintaks impor dalam Golang.

|  |
| --- |
| import "nama librarynya satu"  import "nama librarynya dua" |

Atau

|  |
| --- |
| import(  "library satu"  "library dua" ) |

Sebagai contoh:

|  |
| --- |
| Import “fmt” |

Yang berarti file main.go akan mengimport library fmt.

## **METHOD MAIN**

Dalam bahasa Golang, terdapat kebutuhan akan adanya sebuah fungsi utama yang disebut "main". Fungsi "main" ini akan dieksekusi pertama kali saat program dijalankan. Tanpa adanya fungsi "main", program tidak akan dapat dijalankan.

Berikut ini adalah contoh dari fungsi "main":

|  |
| --- |
| func main(){  fmt.Println(“Belajar Golang dasar go Docs dan Sheet”)  } |

Fungsi fmt.Println digunakan untuk mencetak kata atau nilai ke layar dalam sebuah perintah.

## **EKSPRESI**

Bahasa lain mengenal istilah "statement" sebagai pengganti "ekspresi". Statement merupakan unit kode terkecil yang dapat diinterpretasikan sebagai suatu tindakan atau kegiatan.

Pada contoh di atas, Anda menggunakan sebuah ekspresi untuk menuliskan string, yaitu:

|  |
| --- |
| fmt.println("Belajar Golang dasar Go Docs dan Sheet”) |

Kode di atas akan mencetak teks "Belajar Golang dasar Go Docs dan Sheet" di perangkat Anda.

Ekspresi dapat dianggap sebagai sebuah baris kode. Di Golang, berbeda dengan bahasa pemrograman lainnya, Anda tidak perlu menambahkan titik koma (;) di akhir setiap baris kode.

## **BLOK PROGRAM GO-LANGUAGE**

Blok kode adalah tanda yang digunakan untuk mengelompokkan kode lain. Blok kode ditandai dengan tanda { di awal dan tanda } di akhir sebuah fungsi.

Berikut adalah contohnya:

|  |
| --- |
| //Blok Program If pada G-lang  If i<i{  } |

Blok dan kode dalam Golang memiliki banyak bentuk. Intinya, setiap pembukaan kurung kurawal { harus diikuti dengan penutupan } yang sesuai.

## **PENULISAN BAGIAN KOMENTAR PADA GOLANG**

Komentar adalah teks atau tulisan yang tidak dieksekusi saat program dijalankan. Komentar memiliki manfaat dalam menghasilkan dokumentasi dan mematikan ekspresi atau fungsi tertentu dalam kode program.

Gaya penulisan komentar dalam bahasa pemrograman Golang dibagi menjadi dua, yaitu inline dan multiline. Komentar inline ditandai dengan menggunakan dua garis miring (//...) dalam satu baris. Sedangkan komentar multiline menggunakan garis miring diikuti dengan bintang (/.../) untuk komentar yang meluas lebih dari satu baris kode program.

Berikut ini adalah contohnya:

|  |
| --- |
| func main() {  // fmt.Println("Belajar Golang Go Docs dan Sheet") | inline  fmt.Println(" Belajar Golang Go Docs dan Sheet ")  /\*  Belajar Pemrograman Golang  di Kodingin / multiline  \*/  } |

Variabel adalah tempat untuk menyimpan nilai dalam program. Kita dapat memberikan nama pada variabel sehingga kita dapat merujuknya nanti. Misalnya, kita dapat membuat variabel "umur" untuk menyimpan umur seseorang.

Dalam Golang, kita perlu mendeklarasikan tipe data variabel, seperti int (untuk bilangan bulat), string (untuk teks), float32 (untu menyimpan bilangan desimal) dan Boolean (menyimpan nilai dengan dua status: benar atau salah).

Contoh penggunaan variabel dalam Golang:

|  |
| --- |
| func main() {  var a string  var b int  var c bool    fmt.Println(a)  fmt.Println(b)  fmt.Println(c)  } |

Contoh penggunaan banyak variabel:

|  |
| --- |
| func main() {  var a, b, c, d int = 1, 3, 5, 7    fmt.Println(a)  fmt.Println(b)  fmt.Println(c)  fmt.Println(d)  } |

Contoh Variable dengan banyak kata:

1. Camel Case:

|  |
| --- |
| MyVariableName = “Thea” |

Setiap kata, selain yang pertama, dimulai dengan huruf kapital

1. Pascal Case:

|  |
| --- |
| MyVariableName = “Thea” |

Setiap kata diawali dengan huruf kapital

1. Snake Case:

|  |
| --- |
| My\_variable\_name = “Thea” |

Setiap kata dipisahkan dengan underscore.

## **KONSTANTA(CONSTANS)**

Konstanta adalah nilai yang tetap dan tidak dapat diubah selama program berjalan. Dalam Go, kita dapat mendefinisikan konstanta menggunakan kata kunci const. Konstanta sangat berguna untuk menyimpan nilai-nilai yang tidak berubah dan perlu dipergunakan secara konsisten dalam program.

Dalam deklarasi konstanta, kita harus menentukan tipe data dan memberikan nilai awal yang tetap. Berikut adalah contoh deklarasi konstanta dalam Go:

|  |
| --- |
| func main() {  const pi = 3.14  const appName = "MyApp"  const maxRetry = 3    fmt.Println("Nama aplikasi:", appName)  fmt.Println("Nilai pi:", pi)  fmt.Println("Batas retry:", maxRetry)  } |

Pada contoh di atas, kita mendeklarasikan konstanta pi dengan nilai 3.14, appName dengan nilai "MyApp", dan maxRetry dengan nilai 3. Perlu diingat bahwa tipe data konstanta ditentukan oleh nilai yang diberikan.

Konstanta juga dapat digunakan dalam ekspresi matematika atau operasi lainnya. Misalnya:

|  |
| --- |
| func main() {  const radius = 5  const circumference = 2 \* pi \* radius    fmt.Println("Jari-jari lingkaran:", radius)  fmt.Println("Keliling lingkaran:", circumference)  } |

Dalam contoh tersebut, kita menggunakan konstanta radius dengan nilai 5 untuk menghitung circumference (keliling) lingkaran dengan menggunakan rumus matematika yang melibatkan konstanta pi.

Keuntungan penggunaan konstanta adalah memastikan konsistensi nilai yang digunakan di berbagai bagian program. Jika nilai konstanta perlu diubah, kita hanya perlu mengubahnya di satu tempat, yaitu saat deklarasi konstanta. Hal ini memudahkan pemeliharaan kode.

Namun, perlu diingat bahwa konstanta hanya dapat diberikan nilai saat deklarasi dan nilai tersebut tidak dapat diubah selama program berjalan. Jika kita mencoba mengubah nilai konstanta, maka akan terjadi kesalahan kompilasi.

Dalam penggunaan konstanta, sebaiknya gunakan nama yang jelas dan deskriptif agar mudah dimengerti oleh pembaca kode. Hindari penggunaan konstanta dengan nama yang terlalu umum atau ambigu.

Dengan pemahaman tentang konstanta dalam Go, kita dapat menggunakan nilai-nilai tetap ini secara konsisten dan menjaga konsistensi nilai dalam program yang kita tulis.

## **OUTPUT FUNCTION**

Output function adalah fungsi yang digunakan untuk menampilkan hasil keluaran (output) dari program kita. Dalam bahasa pemrograman Go, kita dapat menggunakan fungsi fmt.Println() untuk mencetak nilai-nilai ke layar. Fungsi ini akan menampilkan teks dan nilai-nilai yang ingin kita tampilkan.

Contoh penggunaan fungsi fmt.Println():

|  |
| --- |
| func main() {  fmt.Println("Halo, dunia!")  fmt.Println("Ini adalah contoh output function.")  fmt.Println("Selamat belajar Go!")  } |

Pada contoh di atas, kita menggunakan fmt.Println() untuk mencetak beberapa teks ke layar. Setiap pemanggilan fungsi fmt.Println() akan mencetak teks pada baris baru.

## **FORMATTING WEB**

Formatting verb adalah cara kita memformat nilai-nilai yang akan dicetak menggunakan fungsi fmt.Println() atau fungsi lainnya dari paket fmt. Formatting verb digunakan untuk mengatur tampilan dan tata letak keluaran yang akan ditampilkan.

Berikut adalah beberapa contoh penggunaan formatting verb dalam Go:

|  |
| --- |
| func main() {  nama := "John"  umur := 25  tinggi := 175.5    fmt.Printf("Nama: %s\n", nama)  fmt.Printf("Umur: %d tahun\n", umur)  fmt.Printf("Tinggi: %.2f cm\n", tinggi)  } |

Pada contoh di atas, kita menggunakan fungsi fmt.Printf() untuk mencetak nilai-nilai dengan menggunakan formatting verb.

1. %s digunakan untuk mencetak nilai string.
2. %d digunakan untuk mencetak nilai integer.
3. %.2f digunakan untuk mencetak nilai float dengan 2 digit desimal.

Dalam contoh tersebut, kita mencetak nilai dari variabel nama, umur, dan tinggi dengan menggunakan formatting verb yang sesuai. Tanda % diikuti oleh huruf atau simbol menunjukkan jenis data yang akan dicetak, dan argumen setelah tanda koma adalah nilai-nilai yang ingin dicetak.

Dengan menggunakan formatting verb, kita dapat mengatur tampilan dan format keluaran sesuai kebutuhan, seperti menentukan jumlah digit desimal, menambahkan tanda khusus, dan lain sebagainya.

## **TIPE DATA**

Tipe data adalah klasifikasi atau kategori yang digunakan untuk mendefinisikan jenis nilai yang dapat disimpan dan dioperasikan dalam program. Dalam bahasa pemrograman Go, terdapat beberapa tipe data bawaan yang dapat kita gunakan untuk menyimpan nilai, seperti tipe data string, integer, float, boolean, dan sebagainya.

Dalam Go, setiap variabel harus memiliki tipe data yang sesuai. Tipe data memberikan informasi tentang cara variabel tersebut akan digunakan dan bagaimana operasinya dapat dilakukan.

Berikut adalah contoh penggunaan beberapa tipe data dalam Go:

|  |
| --- |
| func main() {  var angka int = 10  var decimal float64 = 3.14  var status bool = true  var kata string = "Halo, dunia!"    fmt.Println("Angka:", angka)  fmt.Println("Decimal:", decimal)  fmt.Println("Status:", status)  fmt.Println("Kata:", kata)  } |

Pada contoh di atas, kita mendeklarasikan beberapa variabel dengan tipe data yang berbeda. Variabel nama memiliki tipe data string dan menyimpan nilai "John Doe". Variabel umur memiliki tipe data integer dan menyimpan nilai 25. Variabel tinggi memiliki tipe data float64 dan menyimpan nilai 175.5. Variabel sudahMenikah memiliki tipe data boolean dan menyimpan nilai false.

Dalam penggunaan tipe data, perlu diperhatikan bahwa kita harus menginisialisasi variabel dengan nilai yang sesuai dengan tipe datanya. Hal ini penting agar Go dapat melakukan pengecekan tipe data saat kompilasi program.

Dengan pemahaman tentang tipe data dalam Go, kita dapat memilih tipe data yang sesuai dengan nilai yang akan disimpan dan memastikan operasi yang dilakukan pada variabel berjalan dengan benar.

## **PENGENALAN OPERATOR**

Operator adalah simbol atau tanda khusus yang digunakan untuk melakukan operasi atau manipulasi pada nilai-nilai dalam program. Dalam bahasa pemrograman Go, terdapat berbagai jenis operator yang dapat digunakan, seperti operator aritmatika, operator perbandingan, operator logika, dan lain sebagainya. Operator memungkinkan kita untuk melakukan perhitungan, perbandingan, atau menggabungkan nilai-nilai dengan cara yang telah ditentukan.

Dengan pemahaman tentang operator dalam Go, kita dapat menggunakannya dengan tepat untuk memanipulasi dan memproses nilai-nilai dalam program dengan cara yang diinginkan.

### Aritmatika (Arithmetic)

Aritmatika adalah cabang matematika yang berkaitan dengan operasi matematika dasar seperti penjumlahan, pengurangan, perkalian, dan pembagian. Dalam bahasa pemrograman Go, terdapat operator aritmatika yang memungkinkan kita untuk melakukan operasi-aritmatika pada nilai-nilai numerik.Berikut adalah contoh penggunaan Operator dalam Aritmatika:

|  |
| --- |
| func main() {  // Penjumlahan  a := 5 + 3  fmt.Println("Penjumlahan:", a) |
| // Pengurangan  b := 10 - 4  fmt.Println("Pengurangan:", b)  // Perkalian  c := 6 \* 2  fmt.Println("Perkalian:", c)  // Pembagian  d := 15 / 3  fmt.Println("Pembagian:", d)  // Sisa bagi  e := 17 % 5  fmt.Println("Sisa bagi:", e)  } |

Pada contoh di atas, kita menggunakan operator aritmatika untuk melakukan operasi penjumlahan, pengurangan, perkalian, pembagian, dan sisa bagi. Operator + digunakan untuk penjumlahan, - untuk pengurangan, \* untuk perkalian, / untuk pembagian, dan % untuk sisa bagi.

Dengan menggunakan operator aritmatika, kita dapat melakukan perhitungan matematika dalam program Go dengan mudah dan efisien.

### Assignment

Assignment adalah proses memberikan nilai pada variabel dalam bahasa pemrograman Go. Dalam Go, kita dapat menggunakan operator assignment (=) untuk memberikan nilai pada variabel.

Berikut adalah contoh penggunaan operator assignment dalam Go:

|  |
| --- |
| func printNameAndAge() {  name := "John Doe"  age := 25  fmt.Println("Nama:", name)  fmt.Println("Umur:", age)  }  func main() {  printNameAndAge()  } |

Dengan menggunakan operator assignment, kita dapat dengan mudah memberikan nilai pada variabel dalam program Go. Hal ini memungkinkan kita untuk menyimpan dan mengolah data dengan lebih fleksibel.

### Comparison

Comparison (perbandingan) adalah proses membandingkan dua nilai untuk menentukan hubungan antara keduanya, seperti kesamaan, ketidaksamaan, atau relasi lainnya. Dalam bahasa pemrograman Go, kita menggunakan operator perbandingan untuk melakukan operasi perbandingan.

|  |
| --- |
| func main() {  // Perbandingan nilai  x := 5  y := 10    // Operator equality (==)  fmt.Println("x == y:", x == y)    // Operator inequality (!=)  fmt.Println("x != y:", x != y)    // Operator greater than (>)  fmt.Println("x > y:", x > y)    // Operator less than (<)  fmt.Println("x < y:", x < y) |
| // Operator greater than or equal (>=)  fmt.Println("x >= y:", x >= y)    // Operator less than or equal (<=)  fmt.Println("x <= y:", x <= y)  } |

Pada contoh di atas, kita menggunakan operator perbandingan untuk membandingkan nilai variabel x dan y. Operator == digunakan untuk memeriksa kesamaan, != untuk memeriksa ketidaksamaan, > untuk memeriksa lebih besar dari, < untuk memeriksa lebih kecil dari, >= untuk memeriksa lebih besar dari atau sama dengan, dan <= untuk memeriksa lebih kecil dari atau sama dengan.

Dengan menggunakan operator perbandingan, kita dapat dengan mudah membandingkan nilai-nilai dalam program Go dan mengambil keputusan berdasarkan hasil perbandingan tersebut.

### Logical

Logical (logika) adalah konsep dalam pemrograman untuk menggabungkan dan memanipulasi kondisi atau ekspresi logika. Dalam bahasa pemrograman Go, kita menggunakan operator logika untuk mengendalikan alur program berdasarkan kondisi-kondisi logika.

|  |
| --- |
| func main() {  // Variabel-variabel logika  a := true  b := false |
| // Operator logika AND (&&)  fmt.Println("a && b:", a && b)    // Operator logika OR (||)  fmt.Println("a || b:", a || b)    // Operator logika NOT (!)  fmt.Println("!a:", !a)  fmt.Println("!b:", !b)  } |

Dengan menggunakan operator logika, kita dapat mengontrol alur program berdasarkan kondisi-kondisi logika yang kita tentukan. Hal ini memungkinkan kita untuk membuat keputusan dan menjalankan blok kode tertentu berdasarkan kondisi yang terpenuhi.

### Bitwise

Bitwise adalah operasi yang dilakukan pada level bit individu dari suatu nilai. Dalam bahasa pemrograman Go, kita dapat melakukan operasi bitwise untuk memanipulasi dan mengontrol bit-bit dalam nilai-nilai bilangan.

Berikut adalah contoh penggunaan operasi bitwise dalam Go:

|  |
| --- |
|  |
| func main() {  // Variabel-variabel integer  a := 5 // 00000101 dalam biner  b := 3 // 00000011 dalam biner    // Operator bitwise AND (&)  fmt.Println("a & b:", a&b) // 00000001 dalam biner    // Operator bitwise OR (|)  fmt.Println("a | b:", a|b) // 00000111 dalam biner    // Operator bitwise XOR (^)  fmt.Println("a ^ b:", a^b) // 00000110 dalam biner    // Operator bitwise shift left (<<)  fmt.Println("a << 2:", a<<2) // 00010100 dalam biner    // Operator bitwise shift right (>>)  fmt.Println("a >> 2:", a>>2) // 00000001 dalam biner  } |

Pada contoh di atas, kita menggunakan operator bitwise untuk melakukan operasi pada variabel integer a dan b. Operator & (AND) digunakan untuk melakukan operasi bitwise AND pada bit-bit kedua nilai. Operator | (OR) digunakan untuk melakukan operasi bitwise OR pada bit-bit kedua nilai. Operator ^ (XOR) digunakan untuk melakukan operasi bitwise XOR pada bit-bit kedua nilai. Operator << (shift left) digunakan untuk menggeser bit ke kiri, sedangkan operator >> (shift right) digunakan untuk menggeser bit ke kanan.

Dengan menggunakan operasi bitwise, kita dapat memanipulasi dan mengontrol bit-bit dalam nilai-nilai bilangan, yang berguna dalam berbagai aplikasi seperti manipulasi flag, enkripsi, atau pengkodean.

## **KONDISI(CONDITIONS)**

Conditions (kondisi) adalah konsep dalam pemrograman yang digunakan untuk mengatur alur program berdasarkan keputusan yang dibuat berdasarkan kondisi tertentu. Dalam bahasa pemrograman Go, kita menggunakan struktur kendali if, else if, dan else untuk mengimplementasikan kondisi.

Berikut adalah contoh penggunaan struktur kendali if-else dalam Go:

|  |
| --- |
| func main() {  // Variabel umur  umur := 18  // Struktur kendali if-else  if umur >= 18 {  fmt.Println("Anda sudah cukup umur")  } else {  fmt.Println("Anda masih di bawah umur")}  } |

Pada contoh di atas, kita menggunakan struktur kendali if-else untuk memeriksa apakah nilai variabel umur lebih besar atau sama dengan 18. Jika kondisi tersebut terpenuhi, maka blok kode dalam if akan dieksekusi. Jika tidak, maka blok kode dalam else akan dieksekusi.

### If Statement

If statement adalah struktur kendali dalam bahasa pemrograman Go yang digunakan untuk melakukan pengecekan kondisi tertentu. Dengan if statement, kita dapat menjalankan blok kode tertentu jika kondisi yang diberikan terpenuhi.

Berikut adalah contoh penggunaan if statement dalam Go:

|  |
| --- |
| package main    import "fmt"    func main() {  // Variabel umur  umur := 20    // if statement  if umur >= 18 {  fmt.Println("Anda sudah cukup umur")  }  }  \ |

Pada contoh di atas, kita menggunakan if statement untuk memeriksa apakah nilai variabel umur lebih besar atau sama dengan 18. Jika kondisi tersebut terpenuhi, maka blok kode dalam if akan di-eksekusi dan output "Anda sudah cukup umur" akan ditampilkan.

### if else Statement

if-else statement adalah struktur kendali dalam bahasa pemrograman Go yang digunakan untuk melakukan pengecekan kondisi tertentu dan memberikan alternatif jalur eksekusi jika kondisi tersebut tidak terpenuhi. Dengan if-else statement, kita dapat menjalankan blok kode tertentu jika kondisi yang diberikan terpenuhi, dan menjalankan blok kode lainnya jika kondisi tersebut tidak terpenuhi.

Berikut adalah contoh penggunaan if-else statement dalam Go:

|  |
| --- |
| package main  import "fmt"  func main() {  // Variabel umur  umur := 16    // if-else statement  if umur >= 18 {  fmt.Println("Anda sudah cukup umur")  } else {  fmt.Println("Anda masih di bawah umur")  }  } |

Pada contoh di atas, kita menggunakan if-else statement untuk memeriksa apakah nilai variabel umur lebih besar atau sama dengan 18. Jika kondisi tersebut terpenuhi, maka blok kode dalam if akan di-eksekusi dan output "Anda sudah cukup umur" akan ditampilkan. Namun, jika kondisi tersebut tidak terpenuhi, maka blok kode dalam else akan di-eksekusi dan output "Anda masih di bawah umur" akan ditampilkan.

Dengan menggunakan if-else statement, kita dapat memberikan alternatif jalur eksekusi dalam program. Jika kondisi dalam if terpenuhi, blok kode dalam if akan dieksekusi. Namun, jika kondisi tersebut tidak terpenuhi, blok kode dalam else akan dieksekusi.

### if-else-if Statement

Selain menggunakan if-else statement, kita juga dapat menggunakan if-else-if statement untuk menguji beberapa kondisi secara berurutan. Berikut adalah contoh penggunaan if-else-if statement dalam Go:

|  |
| --- |
| package main    import "fmt"    func main() {  // Variabel nilai  nilai := 85    // if-else-if statement  if nilai >= 90 {  fmt.Println("Anda mendapatkan nilai A")  } else if nilai >= 80 {  fmt.Println("Anda mendapatkan nilai B")  } else if nilai >= 70 {  fmt.Println("Anda mendapatkan nilai C")  } else {  fmt.Println("Anda mendapatkan nilai D")  }  } |

Pada contoh di atas, kita menggunakan if-else-if statement untuk memeriksa nilai variabel nilai secara berurutan. Program akan mengevaluasi setiap kondisi dari atas ke bawah. Jika kondisi pertama terpenuhi, maka blok kode dalam if pertama akan di-eksekusi dan output "Anda mendapatkan nilai A" akan ditampilkan. Jika tidak, maka program akan melanjutkan ke kondisi berikutnya, dan seterusnya.

Dengan menggunakan if-else-if statement, kita dapat memberikan alternatif jalur eksekusi yang lebih kompleks berdasarkan kondisi-kondisi yang berbeda.

### Else-if Statement

Else if statement adalah bagian dari struktur kendali dalam bahasa pemrograman Go yang digunakan untuk mengevaluasi beberapa kondisi secara berurutan. Jika kondisi pertama tidak terpenuhi, maka program akan memeriksa kondisi berikutnya menggunakan else if statement. Jika kondisi pada else if terpenuhi, blok kode yang terkait akan di-eksekusi.

Berikut adalah contoh penggunaan else if statement dalam Go:

|  |
| --- |
| package main    import "fmt"    func main() {  nilai := 80    if nilai >= 90 {  fmt.Println("Nilai Anda A")  } else if nilai >= 80 {  fmt.Println("Nilai Anda B")  } else if nilai >= 70 {  fmt.Println("Nilai Anda C")  } else if nilai >= 60 {  fmt.Println("Nilai Anda D")  } else {  fmt.Println("Nilai Anda E")  }  } |

Dalam contoh di atas, kita menggunakan else if statement untuk mengevaluasi beberapa kondisi berdasarkan nilai variabel nilai. Program akan mengevaluasi kondisi secara berurutan, dan jika suatu kondisi terpenuhi, blok kode yang terkait akan di-eksekusi. Jika tidak ada kondisi yang terpenuhi, maka blok kode pada else akan di-eksekusi.

Dalam contoh tersebut, kita memeriksa apakah nilai lebih besar atau sama dengan 90, 80, 70, dan 60 secara berurutan. Jika suatu kondisi terpenuhi, maka output yang sesuai akan ditampilkan.

### Nested-if

Nested if statement adalah struktur kendali dalam bahasa pemrograman Go yang memungkinkan kita untuk menyatukan beberapa pernyataan if di dalam pernyataan if lainnya. Dalam nested if, kita dapat menempatkan satu pernyataan if di dalam blok kode pernyataan if lainnya. Hal ini memungkinkan kita untuk membuat pengecekan kondisi yang lebih kompleks dan berlapis.

Berikut adalah contoh penggunaan nested if statement dalam Go:

|  |
| --- |
| package main    import "fmt"    func main() {  umur := 20  tinggi := 165    if umur >= 18 {  fmt.Println("Anda sudah cukup umur")  if tinggi >= 170 {  fmt.Println("Anda juga cukup tinggi")  } else {  fmt.Println("Anda kurang tinggi") |
| }  } else {  fmt.Println("Anda masih di bawah umur")  }  } |

Dalam contoh di atas, kita menggunakan nested if statement untuk menguji dua kondisi secara berurutan. Pertama, kita memeriksa apakah umur lebih besar atau sama dengan 18. Jika kondisi ini terpenuhi, blok kode dalam if pertama dieksekusi. Di dalamnya, kita melakukan pengecekan tinggi. Jika tinggi lebih besar atau sama dengan 170, maka pesan "Anda juga cukup tinggi" ditampilkan. Jika tinggi tidak memenuhi kondisi tersebut, maka pesan "Anda kurang tinggi" ditampilkan. Jika umur kurang dari 18, blok kode dalam else akan dieksekusi dan pesan "Anda masih di bawah umur" ditampilkan.

Dengan menggunakan nested if statement, kita dapat membuat pengecekan kondisi yang lebih terperinci dan berlapis. Hal ini memungkinkan kita untuk mengambil tindakan yang sesuai berdasarkan kombinasi dari beberapa kondisi.

## **SWITCH**

Nested if statement adalah struktur kendali dalam bahasa pemrograman Go yang memungkinkan kita untuk menyatukan beberapa pernyataan if di dalam pernyataan if lainnya. Dalam nested if, kita dapat menempatkan satu pernyataan if di dalam blok kode pernyataan if lainnya. Hal ini memungkinkan kita untuk membuat pengecekan kondisi yang lebih kompleks dan berlapis

### Switch single-case

Switch single-case statement dalam Go digunakan ketika kita hanya memiliki satu kasus (case) yang ingin kita evaluasi dalam struktur switch. Ini memungkinkan kita untuk melakukan pengecekan tunggal dan menjalankan blok kode terkait jika kondisi terpenuhi.

Contoh Kode:

|  |
| --- |
| package main    import "fmt"    func main() {  var nilai int = 90    switch nilai {  case 90:  fmt.Println("Nilai sangat baik!")  }  } |

Penjelasan:

1. Pada contoh di atas, kita memiliki sebuah variabel nilai yang bernilai 90.
2. Dalam struktur switch, kita mengevaluasi nilai tersebut dengan menggunakan switch nilai.
3. Dalam kasus ini, kita hanya memiliki satu kasus yaitu case 90. Jika nilai variabel sama dengan 90, maka blok kode di dalam case akan dijalankan.
4. Dalam contoh ini, akan mencetak "Nilai sangat baik!" ke konsol karena nilai variabel nilai adalah 90.

Pada dasarnya, switch single-case statement digunakan ketika kita ingin melakukan pengecekan sederhana terhadap satu kondisi dan menjalankan blok kode terkait jika kondisi terpenuhi.

### Switch multi-case

Switch multi-case statement dalam Go digunakan ketika kita ingin mengevaluasi beberapa kasus (case) dalam struktur switch dan menjalankan blok kode terkait jika salah satu kondisi terpenuhi. Hal ini memungkinkan kita untuk menangani beberapa kemungkinan nilai dengan cara yang lebih terstruktur.

Contoh Kode:

|  |
| --- |
| package main    import "fmt"    func main() {  var nilai int = 80    switch nilai {  case 90:  fmt.Println("Nilai sangat baik!")  case 80, 85:  fmt.Println("Nilai baik!")  case 70, 75:  fmt.Println("Nilai cukup baik.")  default:  fmt.Println("Nilai standar.")  }  } |

Penjelasan:

1. Pada contoh di atas, kita memiliki sebuah variabel nilai yang bernilai 80.
2. Dalam struktur switch, kita mengevaluasi nilai tersebut dengan menggunakan switch nilai.
3. Kita memiliki beberapa kasus yang dievaluasi, yaitu:

* Pada case 90, jika nilai variabel sama dengan 90, maka blok kode di dalam case tersebut akan dijalankan.
* Pada case 80, 85, jika nilai variabel sama dengan 80 atau 85, maka blok kode di dalam case tersebut akan dijalankan.
* Pada case 70, 75, jika nilai variabel sama dengan 70 atau 75, maka blok kode di dalam case tersebut akan dijalankan.

1. Jika tidak ada kasus yang terpenuhi, maka blok kode di dalam default akan dijalankan.
2. Dalam contoh ini, karena nilai variabel nilai adalah 80, maka akan mencetak "Nilai baik!" ke konsol.

Pada dasarnya, switch multi-case statement digunakan ketika kita ingin mengevaluasi beberapa kasus dan melakukan tindakan berbeda tergantung pada nilai yang dievaluasi. Ini memberi kita fleksibilitas dalam menangani berbagai kemungkinan nilai dengan cara yang lebih terstruktur.

## **STRUCT**

Struct dalam Go adalah tipe data yang memungkinkan kita untuk mengelompokkan beberapa variabel terkait menjadi satu entitas tunggal. Dalam struktur, kita dapat mendefinisikan properti-properti atau field-field yang akan menyimpan nilai-nilai tersebut. Hal ini memungkinkan kita untuk mengatur data dengan cara yang lebih terstruktur dan mempermudah manipulasinya.

Contoh Kode:

|  |
| --- |
| func main() {  // Membuat objek dari struct Person  p := Person{  Name: "John Doe",  Age: 25,  Gender: "Male",  }    // Menampilkan nilai-nilai dari objek p  fmt.Println("Name:", p.Name)  fmt.Println("Age:", p.Age)  fmt.Println("Gender:", p.Gender)  } |

Penjelasan:

1. Pada contoh di atas, kita mendefinisikan sebuah struct bernama Person.
2. Struct Person memiliki tiga field, yaitu Name (string), Age (int), dan Gender (string).
3. Di dalam fungsi main(), kita membuat objek baru dari struct Person dengan nilai-nilai tertentu.
4. Objek tersebut diinisialisasi dengan menggunakan sintaks Person{} dan memberikan nilai kepada setiap field.
5. Setelah objek dibuat, kita dapat mengakses nilai-nilai field menggunakan sintaks namaObjek.namaField.
6. Pada contoh ini, kita mencetak nilai-nilai dari objek p, yaitu nama, usia, dan jenis kelamin.

Dengan menggunakan struct, kita dapat menggabungkan beberapa variabel terkait menjadi satu kesatuan yang disebut objek. Objek tersebut dapat memiliki properti atau field yang merepresentasikan nilai-nilai yang berbeda. Ini mempermudah pengelolaan dan manipulasi data dalam program Go dengan cara yang lebih terstruktur dan mudah dipahami.

## **MAPS**

Maps dalam Go adalah struktur data yang digunakan untuk memetakan atau mengaitkan nilai-nilai dengan kunci-kunci yang unik. Maps berfungsi sebagai koleksi pasangan kunci-nilai, di mana setiap kunci berbeda dan memetakan ke nilai tertentu. Dengan menggunakan maps, kita dapat dengan mudah mencari nilai berdasarkan kunci yang diberikan.

Contoh Kode:

|  |
| --- |
| func main() {  // Membuat map dengan kunci bertipe string dan nilai bertipe int  studentGrades := make(map[string]int)    // Menambahkan data ke map  studentGrades["John"] = 90  studentGrades["Jane"] = 95  studentGrades["Mike"] = 85    // Mengakses nilai dari map  fmt.Println("John's Grade:", studentGrades["John"])  fmt.Println("Jane's Grade:", studentGrades["Jane"])  fmt.Println("Mike's Grade:", studentGrades["Mike"])    // Menghapus data dari map  delete(studentGrades, "Mike")    // Menggunakan fungsi len untuk menghitung jumlah pasangan kunci-nilai dalam map  fmt.Println("Number of Students:", len(studentGrades))  } |

Penjelasan:

1. Pada contoh di atas, kita menggunakan fungsi make untuk membuat sebuah map dengan kunci bertipe string dan nilai bertipe int. Map tersebut diberi nama studentGrades.
2. Kemudian, kita menambahkan beberapa pasangan kunci-nilai ke dalam map studentGrades. Setiap kunci (nama siswa) dihubungkan dengan nilai (nilai siswa).
3. Untuk mengakses nilai dari map, kita menggunakan sintaks namaMap[namaKunci]. Contohnya, studentGrades["John"] mengembalikan nilai 90.
4. Kita juga dapat menghapus data dari map menggunakan fungsi delete. Pada contoh ini, kita menghapus data siswa "Mike" dari map.
5. Fungsi len digunakan untuk menghitung jumlah pasangan kunci-nilai dalam map. Pada contoh ini, kita mencetak jumlah siswa dalam map.

Maps sangat berguna dalam menyimpan data yang membutuhkan pengaitan antara kunci dan nilai. Dengan maps, kita dapat dengan mudah mencari nilai berdasarkan kunci dan memanipulasi data dengan efisien.

## **STRUKTUR KENDALI**

Struktur kendali digunakan untuk mengendalikan alur eksekusi program. Salah satu struktur kendali yang umum adalah pernyataan if-else, yang memungkinkan kita untuk menjalankan blok kode tertentu jika kondisi tertentu terpenuhi. Contoh penggunaan if-else dalam Golang:

|  |
| --- |
| func main() {  // Contoh penggunaan if-else  angka := 10  if angka > 5 {  fmt.Println("Angka lebih besar dari 5")  } else {  fmt.Println("Angka tidak lebih besar dari 5") }    // Contoh penggunaan for loop  for i := 1; i <= 3; i++ {  fmt.Println("Perulangan ke", i)  }  // Contoh penggunaan switch  hari := "Senin"  switch hari {  case "Senin":  fmt.Println("Hari ini adalah Senin")  case "Selasa":  fmt.Println("Hari ini adalah Selasa")  default:  fmt.Println("Hari ini bukan Senin atau Selasa")}  } |

Dalam contoh ini, penggunaan if-else, for loop, dan switch lebih sederhana. Kita menggunakan if-else untuk memeriksa apakah angka lebih besar dari 5 atau tidak. Kemudian, menggunakan for loop untuk mencetak perulangan dari 1 hingga 3. Terakhir, kita menggunakan switch untuk memeriksa nilai dari variabel hari dan mencetak pesan yang sesuai.

## **PERULANGAN**

Perulangan digunakan untuk mengulangi blok kode tertentu sejumlah kali. Salah satu jenis perulangan yang umum adalah perulangan for. Misalnya, kita dapat menggunakan perulangan for untuk mencetak angka 1 hingga 5. Contoh penggunaan perulangan for dalam Golang:

|  |
| --- |
| func main() {  // Contoh penggunaan for loop  for i := 1; i <= 5; i++ {  fmt.Println(i)  }  // Contoh penggunaan while loop  j := 1  for j <= 3 {  fmt.Println(j)  j++  }  // Contoh penggunaan infinite loop  k := 1  for {  fmt.Println(k)  k++  if k > 5 {  break}}  } |

Dalam contoh ini, kita menggunakan tiga jenis perulangan yang umum digunakan dalam Golang dengan penjelasan yang lebih sederhana.

1. Pertama, kita menggunakan for loop untuk mencetak angka dari 1 – 5.
2. Kemudian, kita menggunakan while loop dengan for loop. Dalam hal ini, kita menginisialisasi variabel j di luar perulangan dan menggunakan kondisi j <= 3 sebagai syarat untuk terus melakukan perulangan. Di setiap iterasi, kita mencetak nilai j dan meningkatkan nilainya dengan j++.
3. Terakhir, kita menggunakan infinite loop dengan for loop. Dalam hal ini, kita tidak memberikan kondisi awal atau akhir pada for loop. Kita mencetak nilai k dan meningkatkan nilainya dengan k++. Kemudian, kita menggunakan if statement dan break untuk menghentikan perulangan ketika nilai k melebihi 5.

## **FUNGSI**

Fungsi adalah blok kode yang dapat dipanggil untuk melakukan tugas tertentu. Kita dapat membuat fungsi sendiri atau menggunakan fungsi bawaan (built-in) Golang. Contoh penggunaan fungsi dalam Golang:

|  |
| --- |
| // Membuat fungsi sederhana  func sayHello() {  fmt.Println("Halo, dunia!")  }    // Membuat fungsi dengan parameter  func greet(name string) {  fmt.Println("Halo,", name)  }  // Membuat fungsi dengan return value  func addNumbers(a, b int) int {  return a + b}  func main() {  // Panggil fungsi sayHello  sayHello()  // Panggil fungsi greet dengan argument  greet("John")  // Panggil fungsi addNumbers dengan argument dan cetak hasilnya  result := addNumbers(5, 3)  fmt.Println("Hasil penjumlahan:", result)} |

Dalam contoh di atas, kita menggunakan beberapa contoh fungsi dalam Golang.

1. Pertama, kita mendefinisikan fungsi sayHello yang tidak menerima parameter dan hanya mencetak pesan "Halo, dunia!".
2. Kemudian, kita mendefinisikan fungsi greet dengan parameter name yang bertipe string. Fungsi ini akan mencetak pesan "Halo, [name]".
3. Selanjutnya, kita mendefinisikan fungsi addNumbers dengan parameter a dan b yang bertipe int. Fungsi ini akan mengembalikan hasil penjumlahan a dan b.

Di dalam fungsi main, kita memanggil fungsi-fungsi tersebut dan melihat hasilnya. Kita memanggil fungsi sayHello tanpa argumen, memanggil fungsi greet dengan argument "John", dan memanggil fungsi addNumbers dengan argument 5 dan 3. Hasil penjumlahan kemudian dicetak.

## **ARRAY AND SLICES**

Array dan slices adalah struktur data yang digunakan untuk menyimpan sekumpulan nilai. Array memiliki ukuran tetap, sedangkan slices memiliki ukuran yang dapat berubah. Misalnya, kita dapat membuat array atau slices untuk menyimpan daftar angka. Contoh penggunaan array dan slices dalam Golang.

|  |
| --- |
| func main() {  // Contoh Array  var fruits [3]string  fruits[0] = "Apple"  fruits[1] = "Banana"  fruits[2] = "Orange"    fmt.Println(fruits) // Output: [Apple Banana Orange]    // Contoh penggunaan slices  numbers := []int{1, 2, 3, 4, 5}  fmt.Println(numbers) // Output: [1 2 3 4 5]  fmt.Println(numbers[1:3]) // Output: [2 3]  fmt.Println(numbers[:4]) // Output: [1 2 3 4]  fmt.Println(numbers[3:]) // Output: [4 5]  fmt.Println(numbers[:]) // Output: [1 2 3 4 5]    // Mengubah elemen dalam slices  numbers[0] = 10  fmt.Println(numbers) // Output: [10 2 3 4 5]    // Menggunakan fungsi append untuk menambahkan elemen ke slices  numbers = append(numbers, 6)  fmt.Println(numbers) // Output: [10 2 3 4 5 6]} |

Dalam contoh di atas, kita menggunakan array dan slices dalam Golang. Pertama, kita mendefinisikan sebuah array fruits dengan panjang 3. Kemudian, kita mengisi elemen-elemennya dengan "Apple", "Banana", dan "Orange". Cetak array fruits akan menghasilkan [Apple Banana Orange].

Selanjutnya, kita menggunakan slices dengan variabel numbers. Kita mendefinisikan slices dengan elemen-elemen awal [1, 2, 3, 4, 5]. Kemudian, kita mencetak slices numbers secara keseluruhan dan juga mencetak beberapa potongan (slice) dari slices tersebut. Misalnya, numbers[1:3] menghasilkan [2 3], numbers[:4] menghasilkan [1 2 3 4], dan numbers[3:] menghasilkan [4 5]. Potongan [1:3] berarti kita mengambil elemen dari indeks 1 hingga sebelum indeks 3.

Selanjutnya, kita juga dapat mengubah elemen dalam slices dengan mengaksesnya melalui indeks. Misalnya, numbers[0] = 10 mengubah nilai elemen pertama menjadi 10.

Terakhir, kita menggunakan fungsi append untuk menambahkan elemen baru ke dalam slices. Misalnya, numbers = append(numbers, 6) menambahkan angka 6 ke dalam slices numbers.

# **BAB 4**

# **MEMBUAT PACKAGE**



Dalam bab ini kita akan mempelajari cara membuat package kita sendiri. Package yang akan kita buat adalah package untuk memanggil API dari Google Sheets dan Google Docs. Adapun alur pemorgamannya adalah sebagai berikut.

## **INISIALISASI PROYEK**

Sebelum inisialisasi proyek dengan go mod, kita harus buat repository github terlebih dahulu. Untuk nama repository-nya kita sesuaikan dengan proyek kita. Misalkan dalam buku ini, nama repository-nya adalah gdocsheets. Setelah membuat repository di github, kita clone repository kita ke PC/laptop. Masuk ke folder repo yang sudah di clone tadi dan lakukan inisialisasi go mod. Sebagai contoh, disini kita mempunyai repo bernama “PKGSheet”. Berikut CLI-nya.

|  |
| --- |
| go mod init PKGSheet |

Setelah inisialisasi go mod, kita akan mengambil package dari Google Sheets dan Google Docs.

|  |
| --- |
| go get google.golang.org/api/sheets/v4  go get google.golang.org/api/docs/v1  go get golang.org/x/oauth2/google |

## **OTENTIKASI PROYEK**

Untuk dapat terhubung dengan Google API, kita memerlukan proses otentikasi agar proyek kita mendapat otoritas untuk terhubung dengan google API. Untuk melakukan otentikasi kita harus membuat login ke google cloud console. Kemudian membuat Project baru. Beri nama proyek baru di google cloud dengan nama “latihan”. Lalu masuk ke bagian API dan pilih enable API & service. Kemudian cari google sheets dan enable API.

Sekarang setelah mengaktifkan API kita harus membuat kredensial untuk memberikan otoritas kepada program yang akan kita buat. Pilih menu API, lalu pilih menu *Credentials.* Untuk membuat kredensial kita pilih *create credentials* dan pilih OAuth Client ID. Kemudian pilih External untuk akses program kita. Isi informasi aplikasi dengan nama latihan, lalu email support dan email developer silakan gunakan email masing-masing. Untuk scope, test user,dan selanjutnya tidak perlu diisi. Setelah itu publish APP.

Kembali ke menu credentials, pilih *create credentials* lalu pilih OAuth client ID. Untuk application type pilih *Desktop App* Lalu biarkan nama terisi otomatis. Setelah itu download file JSON dan simpan di folder proyek yang telah dibuat. Ganti nama file JSON tersebut menjadi “credentials.json”.

## **MENULIS KODE**

Dalam tahan ini,kita akan mempelajari apa saja hal-hal yang diperlukan dalam penulisan kode,dalam setiap package pastilah memiliki cara dan karakteristik penulisan yang berbeda-beda,berikut adalah contohnya:

### Google Sheet``

Untuk penulisan kode buka folder proyek yang telah dibuat, lalu buka VSCode di folder tersebut.

Pertama, kita akan buat program untuk men-generate token untuk akses ke Google Sheets API. Token ini digunakan untuk otentikasi proyek kita yang sudah mendapatkan kredensial dari google cloud. Buat folder dengan nama token lalu buat file go dengan nama “token.go”. Untuk kodenya sendiri berada di repo github kami, di Bab 4. Dalam kode tersebut kita akan membuat 3 fungsi yaitu ‘GetTokenFromWeb’,’TokenFromFile’, dan ‘SaveToken’.

Fungsi **GetTokenFromWeb** bertujuan untuk mendapatkan token dari web menggunakan alur otorisasi OAuth2. Berikut adalah penjelasan mengenai kode tersebut:

1. Fungsi ini menerima satu argumen, yaitu config yang merupakan objek **‘\*oauth2.Config’** yang digunakan untuk melakukan otorisasi OAuth2. Kode ini akan menghasilkan URL otorisasi dengan menggunakan **‘config.**AuthCodeURL**("state-token", oauth2.AccessTypeOffline)’**. URL ini akan digunakan untuk mengarahkan pengguna ke halaman otorisasi di web.
2. Selanjutnya, kode akan mencetak URL tersebut ke konsol agar pengguna dapat membukanya di browser. Pengguna akan diminta untuk mengetikkan kode otorisasi setelah mengunjungi URL tersebut.
3. Kode ini akan membaca masukan dari pengguna menggunakan **‘fmt.Scan(&**authCode**)’** untuk mendapatkan kode otorisasi yang dimasukkan oleh pengguna.
4. Setelah mendapatkan kode otorisasi, kode ini akan menukarkan kode otorisasi dengan token menggunakan **‘config.Exchange(context.TODO(), authCode)’**. Proses ini melibatkan pengiriman permintaan ke server otorisasi untuk mendapatkan token akses dan token penyegar (refresh token).
5. Jika terjadi kesalahan selama proses pertukaran, kode ini akan menghentikan program dengan menggunakan **‘log.Fatalf()’** dan menampilkan pesan kesalahan yang sesuai.
6. Terakhir, fungsi ini akan mengembalikan token yang didapatkan.

Dengan demikian, fungsi ini digunakan untuk mempermudah alur otorisasi OAuth2 dengan menghasilkan URL otorisasi, meminta pengguna untuk memasukkan kode otorisasi, dan menukarkan kode tersebut dengan token akses.

Fungsi **TokenFromFile** digunakan untuk mengambil token dari sebuah file lokal. Berikut adalah penjelasan mengenai kode tersebut:

1. Fungsi ini menerima satu argumen, yaitu file yang merupakan string yang menunjukkan path file tempat token disimpan.
2. Pertama, kode akan membuka file menggunakan **‘os.Open(file)’**. Jika terjadi kesalahan saat membuka file, maka kode akan mengembalikan **‘nil’** dan error yang terjadi.
3. Setelah file berhasil dibuka, kode akan menutup file secara otomatis setelah selesai digunakan dengan menggunakan **‘defer f.Close()’**.
4. Selanjutnya, kode akan membuat objek **‘oauth2.Token’** kosong yang akan digunakan untuk menyimpan token yang dibaca dari file.
5. Kode akan menggunakan **‘json.NewDecoder(f).Decode(tok)’** untuk mendekode data JSON yang ada di file ke dalam objek token yang telah dibuat sebelumnya. Jika terjadi kesalahan saat mendekode data JSON, maka kode akan mengembalikan token yang telah dibaca dan error yang terjadi.
6. Jika tidak ada kesalahan, kode akan mengembalikan token yang telah dibaca dan error **‘nil’**.

Dengan demikian, fungsi ini bertujuan untuk membaca token dari file yang ditentukan dan mengembalikan token tersebut beserta error yang terjadi selama proses pembacaan.

Fungsi **SaveToken** digunakan untuk menyimpan token ke dalam sebuah file pada path yang ditentukan. Berikut ini adalah penjelasan dari kode tersebut:

1. Fungsi ini menerima dua parameter: **‘path’** yang merupakan string yang menunjukkan path file tujuan, dan **‘token’** yang merupakan pointer ke objek **‘oauth2.Token’** yang akan disimpan.
2. Pertama, kode akan mencetak pesan ke konsol yang menunjukkan path file tujuan yang akan digunakan untuk menyimpan token.
3. Kemudian, kode akan membuka file pada path yang ditentukan menggunakan **‘os.OpenFile()’**. Flag **‘os.O\_RDWR’** digunakan untuk membuka file dengan akses read-write, **‘os.O\_CREATE’** digunakan untuk membuat file jika tidak ada, dan **‘os.O\_TRUNC’** digunakan untuk mengosongkan file jika file sudah ada. Mode 0600 menunjukkan bahwa file hanya dapat diakses oleh pemiliknya.
4. jika terjadi kesalahan saat membuka file, kode akan menghentikan program dengan menggunakan **‘log.Fatalf()’** dan menampilkan pesan kesalahan yang sesuai.
5. Setelah membuka file, kode akan menutup file secara otomatis setelah selesai digunakan dengan menggunakan **‘defer f.Close()’**.
6. Terakhir, kode akan menggunakan **‘json.NewEncoder(f).**Encode**(token)’** untuk mengkodekan token menjadi format JSON dan menyimpannya ke dalam file yang telah dibuka sebelumnya.

Dengan demikian, fungsi ini bertujuan untuk menyimpan token ke dalam file pada path yang ditentukan sehingga dapat digunakan kembali di masa mendatang.

Kemudian kita buat Folder Baru dengan nama Client lalu buat file baru dengan nama “client.go” yang fungsinya untuk otorisasi REST client kita. Untuk kodenya sendiri berada di repo github kami, di Bab 4. Dalam kode ini terdapat 1 fungsi yaitu ‘GetClient’. Fungsi ini menerima dua argumen, yaitu config yang merupakan objek \*oauth2.Config yang digunakan untuk melakukan autentikasi OAuth2, dan filename yang merupakan string yang menunjukkan path file token yang akan digunakan. Berikut adalah penjelasan kodenya.

1. Pertama, kode akan mencoba untuk memperoleh token dari file menggunakan token.TokenFromFile(filename). Jika token berhasil dibaca dari file, maka token tersebut akan digunakan untuk membuat http.Client yang terautentikasi.
2. Jika terjadi kesalahan saat membaca token dari file (misalnya file tidak ada atau tidak valid), maka kode akan menjalankan langkah-langkah berikut:

* Pertama, kode akan memperoleh token dari web menggunakan token.GetTokenFromWeb(config). Fungsi GetTokenFromWeb akan meminta pengguna untuk memasukkan kode otorisasi dan menukarkannya dengan token.
* Selanjutnya, kode akan menyimpan token yang baru didapatkan ke dalam file menggunakan token.SaveToken(filename, tok). Fungsi SaveToken akan mengkodekan token dalam format JSON dan menyimpannya dalam file yang ditentukan.

1. Akhirnya, kode akan mengembalikan http.Client yang terautentikasi menggunakan config.Client(context.Background(), tok). Client ini dapat digunakan untuk melakukan permintaan HTTP yang memerlukan autentikasi OAuth2.

Dengan demikian, fungsi ini bertujuan untuk memperoleh http.Client yang terautentikasi dengan token OAuth2. Jika token tersedia dalam file, maka akan digunakan token tersebut. Jika tidak, maka pengguna akan diminta untuk memasukkan kode otorisasi dan token baru akan disimpan dalam file untuk digunakan di masa mendatang.

Selanjutnya buat folder dengan nama Config lalu tambah file dalam folder tersebut dengan nama “config.go”. Masukkan kode berikut ke dalam file “config.go”. Untuk kodenya sendiri berada di repo github kami, di Bab 4. Dalam kode tersebut ada 1 fungsi yaitu ‘NewConfigGoogle’.

Fungsi **NewConfigGoogle** adalah fungsi publik yang didefinisikan dalam package **‘config’**. Fungsi ini menerima argumen **‘filePath’** yang merupakan path file JSON yang berisi konfigurasi, dan **‘scope’** yang merupakan daftar string yang menunjukkan scope OAuth2 yang diminta. Untuk rinciannya sebagai berikut:

1. Di dalam fungsi, kode membaca isi file menggunakan **‘os.ReadFile(**filePath**)’**. Jika terjadi kesalahan saat membaca file, fungsi akan langsung mengembalikan.
2. Jika file berhasil dibaca, kode memanggil **‘google.ConfigFromJSON(b, scope...)’** untuk memparsing data JSON dari file dan membuat konfigurasi OAuth2. Fungsi **‘google.ConfigFromJSON’** disediakan oleh package **‘golang.org/x/oauth2/google’**.
3. Jika tidak terjadi kesalahan saat pembuatan konfigurasi OAuth2, fungsi ini mengembalikan konfigurasi tersebut.
4. Jika terjadi kesalahan, fungsi mengembalikan **‘nil’** untuk konfigurasi dan error yang terjadi.

Dengan demikian, package **‘config’** ini kemungkinan digunakan untuk mengelola konfigurasi OAuth2 dan menyediakan fungsi **‘NewConfigGoogle’** yang membantu membuat konfigurasi OAuth2 khusus untuk integrasi dengan layanan Google berdasarkan file JSON yang diberikan.

Selanjutnya kita akan masukkan kode utama kita untuk menjalankan fungsi tambah data, baca data, sunting data, dan hapus data dalam google sheets. Buat folder baru dengan nama “Gsheet” lalu buat file dengan nama “sheet.go”. Masukkan kode berikut ke dalam “sheet.go”. Untuk kodenya ada di repository kami di bab 4. Dalam kode tersebut ada 4 fungsi yaitu ‘Buat’,’Baca’,’Update’ dan ‘Hapus’. Berikut adalah penjelasan dari fungsi-fungsi tersebut.

Fungsi **Buat** adalah sebuah fungsi yang didefinisikan dalam package gsheets. Fungsi ini menerima beberapa parameter:

1. srv adalah objek sheets.Service yang digunakan untuk berkomunikasi dengan Google Sheets API.
2. IdSpreadsheet adalah string yang menyimpan ID dari spreadsheet Google Sheets yang akan digunakan.
3. SheetRange adalah string yang menunjukkan range sheet pada spreadsheet yang akan ditulis data.
4. gw adalah objek sheets.ValueRange yang berisi data yang akan ditulis ke Google Sheets.

Di dalam fungsi, kode ini menggunakan beberapa parameter dan nilai untuk melakukan operasi Append pada spreadsheet.

1. Kode ini menggunakan **‘srv.Spreadsheets.Values.Append’** untuk menambahkan data ke spreadsheet dengan menggunakan parameter yang sudah ditentukan.
2. Setelah operasi Append selesai, kode mencetak respons dari operasi tersebut menggunakan fmt.Println(resp).
3. Jika terjadi kesalahan selama operasi Append, kode menggunakan log.Fatal(err) untuk mencetak pesan kesalahan dan menghentikan program.

Secara keseluruhan, fungsi **Buat** digunakan untuk menambahkan data kedalam sebuah spreadsheet di Google Sheets berdasarkan ID spreadsheet, rentang sel, dan data yang akan dimasukkan. Fungsi ini menggunakan metode **‘srv.Spreadsheets.Values.Append’** dari Google Sheets API untuk melakukan permintaan dan mengambil response-nya.

Kemudian ada fungsi **Baca** digunakan untuk mengambil nilai-nilai dari sebuah spreadsheet di Google Sheets menggunakan Google Sheets API. Berikut ini penjelasan dari kode tersebut:

1. Fungsi Baca menerima beberapa parameter: **‘srv’** (sebuah pointer ke objek sheets.Service yang digunakan untuk berkomunikasi dengan Google Sheets API), **‘**IdSpreadhseet’ (sebuah string yang mewakili ID dari spreadsheet yang akan dibaca), **‘SheetRange’** (sebuah string yang menunjukkan rentang sel yang akan dibaca), **‘valueouttype’** (sebuah string yang menentukan opsi rendering nilai), dan **‘dateouttype’** (sebuah string yang menentukan opsi rendering tanggal/waktu).
2. Di dalam fungsi, variabel **‘spreadsheetId’** diisi dengan nilai **‘IdSpreadhseet’**, dan variabel **‘range2’** diisi dengan nilai **‘SheetRange’**.
3. Variabel **‘valueRenderOption’** diisi dengan nilai **‘valueouttype’**, dan variabel **‘dateTimeRenderOption’** diisi dengan nilai **‘dateouttype’**.
4. Kemudian, kode memanggil **‘srv.**Spreadsheets**.Values.Get(spreadsheetId, range2)’** untuk mengambil nilai dari spreadsheet yang ditentukan dan rentang yang ditentukan. Metode **‘ValueRenderOption’** dan **‘DateTimeRenderOption’** digunakan untuk menentukan opsi rendering nilai yang diambil.
5. Metode **‘Do()’** dipanggil untuk menjalankan permintaan API dan mengambil response-nya. Response-nya disimpan dalam variabel **‘resp’**, dan jika terjadi kesalahan selama permintaan, kode akan mencetak pesan kesalahan menggunakan **‘log.Fatal(err)’**.
6. Terakhir, fungsi ini mencetak responsenya (**‘resp’**) ke konsol menggunakan **‘fmt.Println(resp)’**.

Secara keseluruhan, fungsi **Baca** digunakan untuk mengambil nilai-nilai dari sebuah spreadsheet di Google Sheets berdasarkan ID spreadsheet, rentang sel, opsi rendering nilai, dan opsi rendering tanggal/waktu yang diberikan. Fungsi ini menggunakan metode **‘srv.Spreadsheets.Values.Get’** dari Google Sheets API untuk melakukan permintaan dan mengambil response-nya.

Kemudian ada Fungsi **Update** digunakan untuk memperbarui nilai-nilai di dalam sebuah spreadsheet di Google Sheets menggunakan Google Sheets API. Berikut ini penjelasan dari kode tersebut:

1. Fungsi Update menerima beberapa parameter: **‘srv’** (sebuah pointer ke objek **‘sheets.Service’** yang digunakan untuk berkomunikasi dengan Google Sheets API), **‘IdSpreadsheet’** (sebuah string yang mewakili ID dari spreadsheet yang akan diperbarui), **‘SheetRange’** (sebuah string yang menunjukkan rentang sel yang akan diperbarui), dan **‘gw’** (sebuah pointer ke objek sheets.ValueRange yang berisi data yang akan diperbarui).
2. Di dalam fungsi, variabel **‘spreadsheetId’** diisi dengan nilai **‘**IdSpreadsheet’, dan variabel **‘range2’** diisi dengan nilai **‘SheetRange’**.
3. Variabel **‘valueInputOption’** diisi dengan nilai "USER\_ENTERED", yang menandakan bahwa nilai-nilai yang akan diperbarui akan diinterpretasikan berdasarkan aturan pengguna.
4. Variabel **‘rb’** diisi dengan nilai **‘gw’**, yaitu objek **‘sheets.ValueRange’** yang berisi data yang akan diperbarui.
5. Kemudian, kode memanggil **‘srv.Spreadsheets.Values.Update(spreadsheetId, range2, rb)’** untuk memperbarui nilai-nilai di dalam spreadsheet yang ditentukan dan rentang yang ditentukan. Metode **‘ValueInputOption’** digunakan untuk menentukan opsi input nilai, dalam hal ini "USER\_ENTERED".
6. Metode **‘Do()’** dipanggil untuk menjalankan permintaan API dan mengambil response-nya. Response-nya disimpan dalam variabel **‘resp’**, dan jika terjadi kesalahan selama permintaan, kode akan mencetak pesan kesalahan menggunakan **‘log.Fatal(err)’**.
7. Terakhir, fungsi ini mencetak responsenya (**‘resp’**) ke konsol menggunakan **‘fmt.Print(resp)’**.

Secara keseluruhan, fungsi **Update** digunakan untuk memperbarui nilai-nilai di dalam sebuah spreadsheet di Google Sheets berdasarkan ID spreadsheet, rentang sel, dan data yang akan diperbarui. Fungsi ini menggunakan metode srv.Spreadsheets.Values.Update dari Google Sheets API untuk melakukan permintaan dan mengambil responsenya.

Terakhir ada fungsi **Hapus**. Fungsi **Hapus** digunakan untuk menghapus nilai-nilai di dalam sebuah spreadsheet di Google Sheets menggunakan Google Sheets API. Berikut ini penjelasan dari kode tersebut:

1. Fungsi Hapus menerima beberapa parameter: srv (sebuah pointer ke objek sheets.Service yang digunakan untuk berkomunikasi dengan Google Sheets API), IdSpreadhseet (sebuah string yang mewakili ID dari spreadsheet yang akan dihapus nilainya), dan SheetRange (sebuah string yang menunjukkan rentang sel yang akan dihapus nilainya).
2. Di dalam fungsi, variabel spreadsheetId diisi dengan nilai IdSpreadhseet, dan variabel range2 diisi dengan nilai SheetRange.
3. Variabel rb diisi dengan objek sheets.ClearValuesRequest{}, yang digunakan untuk memberikan permintaan penghapusan nilai-nilai.
4. Kemudian,kodememanggil srv.Spreadsheets.Values.Clear(spreadsheetId, range2, rb) untuk menghapus nilai-nilai di dalam spreadsheet yang ditentukan dan rentang yang ditentukan. Metode Do() digunakan untuk menjalankan permintaan API.
5. Metode Do() mengembalikan responsen dan kesalahan (jika ada). Jika terjadi kesalahan, kode akan mencetak pesan kesalahan menggunakan log.Fatal(err).
6. Terakhir, fungsi ini mencetak responsen (resp) ke konsol menggunakan fmt.Println(resp).

Secara keseluruhan, fungsi Hapus digunakan untuk menghapus nilai-nilai di dalam sebuah spreadsheet di Google Sheets berdasarkan ID spreadsheet dan rentang sel yang ditentukan. Fungsi ini menggunakan metode srv.Spreadsheets.Values.Clear dari Google Sheets API untuk melakukan permintaan penghapusan dan mengambil responsenya.

Dengan demikian, package gsheets ini kemungkinan digunakan untuk menyediakan fungsi-fungsi yang mempermudah interaksi dengan Google Sheets menggunakan Google Sheets API, dan fungsi Buat digunakan untuk menambahkan data ke spreadsheet Google Sheets.

### Google Docs

Untuk membuat kode google docs kita sudah tidak perlu membuat file untuk otorisasi dan generate token lagi selama kita masih membuatnya di dalam folder proyek yang sudah kita buat. Cukup buat folder baru dengan nama “GDocs”, lalu buat file dengan nama “docs.go”. Masukkan kode berikut untuk membuat dan menghapus dokumen google docs. Untuk kode ada di repository kami bab 4 folder Gdocs. Kode ini memiliki 2 fungsi yaitu ‘DeleteDocument’ dan ‘Filebaru’.

Fungsi **DeleteDocument** ini digunakan untuk menghapus dokumen dalam Google Docs. Fungsi menerima parameter **‘srv’** (pointer ke objek **‘docs.Service’** yang digunakan untuk berkomunikasi dengan Google Docs API), **‘documentID’** (ID dokumen yang akan dihapus), dan **‘credentialsFile’** (path ke file kredensial untuk otentikasi). Fungsi ini membuat permintaan **‘BatchUpdateDocumentRequest’** untuk menghapus semua isi dokumen menggunakan **‘DeleteContentRangeRequest’**. Jika terjadi kesalahan selama permintaan, fungsi akan mengembalikan error.

Fungsi **Filebaru** ini digunakan untuk mendapatkan informasi tentang dokumen dengan ID tertentu dari Google Docs. Fungsi menerima parameter **‘srv’** (pointer ke objek **‘docs.Service’** yang digunakan untuk berkomunikasi dengan Google Docs API) dan **‘docsID’** (ID dokumen yang ingin diperoleh informasinya). Fungsi ini menggunakan metode **‘Documents.Get’** untuk mendapatkan data tentang dokumen tersebut. Jika terjadi kesalahan selama permintaan, fungsi akan mencetak pesan kesalahan menggunakan **‘log.Fatalf’**, dan jika berhasil, akan mencetak judul dokumen tersebut.

Secara keseluruhan, package gdocs menyediakan fungsi-fungsi untuk menghapus dokumen dalam Google Docs dan mendapatkan informasi tentang dokumen dari Google Docs menggunakan Google Docs API.

## **MENJALANKAN *PACKAGE* GOOGLE DOCS DAN GOOGLE SHEETS**

Untuk Menjalankan Program dari package GSheet dan Gdocs yang telah kita buat, kita harus membuat file baru di folder proyek kita yang menjalankan package main dan fungsi main. Sebelumnya kita harus pastikan kredensial yang telah kita unduh dalam format json ada di dalam folder proyek kita, kemudian ganti nama file kredensial tersebut dengan “credentials.json”. Berikut adalah kode untuk menjalankan package **‘gsheet’**. Pastikan untuk membuat satu file sheet untuk di eksekusi.

|  |
| --- |
| func main() {  ctx := context.Background()  filePath := "credentials.json"  conf, err := config.NewConfigGoogle(filePath, "https://www.googleapis.com/auth/spreadsheets")  if err != nil {  log.Fatalf("Unable to parse client secret file to config: %v", err)  }  client := client.GetClient(conf, "token.json")  srv, err := sheets.NewService(ctx, option.WithHTTPClient(client))  if err != nil {  log.Fatalf("Unable to retrieve Sheets client: %v", err)  }  gw := &sheets.ValueRange{  Range: "Sheet1!A1:D9",  MajorDimension: "COLUMNS",  Values: [][]interface{}{  {"Melinda", "meraih", "mimpi"},  {"kamu", "meraih", "pahala"},  {true, false, true},  },  }  gsheets.Baca(srv, "masukkan id file sheet kalian disini!", gw.Range, "FORMATTED\_VALUE", "FORMATTED\_STRING")  } |

1. Di dalam fungsi main(), konteks (ctx) dibuat menggunakan context.Background() untuk digunakan dalam operasi-operasi yang melibatkan Google Sheets API.
2. Variabel filePath diisi dengan path ke file kredensial (credentials.json) yang digunakan untuk mengotentikasi permintaan ke Google Sheets API.
3. Selanjutnya, fungsi NewConfigGoogle() dari package config digunakan untuk memuat file kredensial dan mengembalikan konfigurasi (conf). Jika terjadi kesalahan, pesan kesalahan akan dicetak menggunakan log.Fatalf().
4. Fungsi GetClient() dari package client digunakan untuk mendapatkan http.Client yang akan digunakan dalam komunikasi dengan Google Sheets API. Fungsi ini menggunakan konfigurasi (conf) dan nama file token (token.json) untuk memperoleh token akses. Jika token tidak tersedia, fungsi GetTokenFromWeb() akan dipanggil untuk mendapatkan token dari web.
5. Setelah itu, objek sheets.Service (srv) dibuat menggunakan sheets.NewService() dengan menggunakan http.Client yang telah diperoleh sebelumnya. Jika terjadi kesalahan, pesan kesalahan akan dicetak menggunakan log.Fatalf().
6. Selanjutnya, sebuah objek sheets.ValueRange (gw) dibuat dengan rentang sel, dimensi utama, dan nilai-nilai yang akan digunakan dalam operasi pembacaan data dari Google Sheets.
7. Fungsi Baca() dari package gsheets dipanggil untuk membaca data dari Google Sheets. Parameter yang diberikan adalah srv (objek sheets.Service), ID file sheet yang diinginkan, rentang sel, jenis tampilan nilai, dan jenis tampilan tanggal. Fungsi ini akan mencetak responsen yang berisi data dari Google Sheets ke konsol.

Pastikan untuk mengganti "masukkan id file sheet kalian disini!" dengan ID file sheet yang sesuai sebelum menjalankan program ini. Setelah itu coba jalankan kode tersebut dengan menggunakan perintah CLI:

|  |
| --- |
| go run main.go |

Maka Anda berhasil membaca nilai dalam file google sheet Anda.

Untuk menjalankan Google docs, masukkan kode berikut.

|  |
| --- |
| func main() {  ctx := context.Background()  filePath := "credentials.json"  conf, err := config.NewConfigGoogle(filePath, "https://www.googleapis.com/auth/documents")  if err != nil {  log.Fatalf("Unable to parse client secret file to config: %v", err)  }  client := client.GetClient(conf, "token.json")  srv, err := docs.NewService(ctx, option.WithHTTPClient(client))  if err != nil {  log.Fatalf("Unable to retrieve Sheets client: %v", err)  }  gdocs.Baca(srv, "masukkan id docs kalian")  } |

Pada contoh kode yang diberikan, terdapat beberapa perubahan dan penambahan dibandingkan dengan kode sebelumnya. Berikut ini penjelasan dari kode tersebut:

1. Package main merupakan package utama dalam program Go.
2. Di dalam fungsi main(), konteks (ctx) dibuat menggunakan context.Background() untuk digunakan dalam operasi-operasi yang melibatkan Google Docs API.
3. Variabel filePath diisi dengan path ke file kredensial (credentials.json) yang digunakan untuk mengotentikasi permintaan ke Google Docs API.
4. Selanjutnya, fungsi NewConfigGoogle() dari package config digunakan untuk memuat file kredensial dan mengembalikan konfigurasi (conf). Jika terjadi kesalahan, pesan kesalahan akan dicetak menggunakan log.Fatalf().
5. Fungsi GetClient() dari package client digunakan untuk mendapatkan http.Client yang akan digunakan dalam komunikasi dengan Google Docs API. Fungsi ini menggunakan konfigurasi (conf) dan nama file token (token.json) untuk memperoleh token akses. Jika token tidak tersedia, fungsi GetTokenFromWeb() akan dipanggil untuk mendapatkan token dari web.
6. Setelah itu, objek docs.Service (srv) dibuat menggunakan docs.NewService() dengan menggunakan http.Client yang telah diperoleh sebelumnya. Jika terjadi kesalahan, pesan kesalahan akan dicetak menggunakan log.Fatalf().
7. Selanjutnya, fungsi Baca() dari package gdocs dipanggil untuk membaca data dari Google Docs. Parameter yang diberikan adalah srv (objek docs.Service) dan ID dokumen Google Docs yang diinginkan. Pastikan untuk mengganti "masukkan id docs kalian" dengan ID dokumen yang sesuai sebelum menjalankan program ini.

Perhatikan juga perubahan pada import package, yaitu PKGSheet/GDocs menjadi PKGSheet/GDocs (tanpa spasi) dan PKGSheet /client menjadi PKGSheet/client (tanpa spasi). Pastikan nama package dan path package sudah benar dan sesuai dengan struktur folder dan file yang ada.

Dengan Kode tersebut kita dapat membaca title dari dokumen kita di google docs.

# **LATIHAN-GO**

Latihan:

1. Isi bagian dari code yang kosong,Output “Hello, World!” .

|  |
| --- |
| package main  import ("fmt")    func main() {  \_\_\_\_\_\_\_\_\_\_("Hello, World!")  } |

1. Komentar dalam Go di tulis menggunakan Karakter spesial,Yaitu?

|  |
| --- |
| package main  import ("fmt")    func main() {    \_\_\_ this is a comment  fmt.Println("Hello World!")  } |

1. Komentar multibaris di Go ditulis dengan karakter khusus, yaitu?

|  |
| --- |
| package main  import ("fmt")    func main() {  \_\_ These lines  are comments \_\_    fmt.Println("Hello World!")  } |

/\* dan \*/

1. Buat variabel bernama “myNum” dan beri nilai 50.

|  |
| --- |
| package main import ("fmt")  func main() {  var \_\_\_ = \_\_  fmt.Println(\_\_\_) } |

1. Tambahkan tipe data yang benar untuk variabel berikut:

|  |
| --- |
| package main import ("fmt")  func main() {  var myNum \_\_= 90  var myWord \_\_\_= "Hello"  var myBool \_\_= true } |

1. Buat array tipe string yang disebut mobil.

|  |
| --- |
| package main import ("fmt")  func main() {  var \_\_\_ = [4]\_\_\_\_{"Volvo", "BMW", "Ford", "Mazda"}  fmt.Print(\_\_\_) } |

1. Kalikan 10 dengan 5,dan print hasilnya.

|  |
| --- |
| package main import ("fmt")  func main() {  fmt.Print(10\_5) } |

1. Print “Hello World” Jika x lebih besar dari y.

|  |
| --- |
| package main import ("fmt")  func main() {  var x = 50  var y = 10  \_\_ x \_ y {  fmt.Print("Hello World")  } } |

1. Masukan bagian yang kosong untuk menyelesaikan statement “switch” berikut.

|  |
| --- |
| package main import ("fmt")  func main() {  var day = 2  switch \_\_ {  \_\_\_(1):  fmt.Print("Saturday")  \_\_\_(2):  fmt.Print("Sunday")   } } |

1. Print “i” selama “i” kurang dari 6.

|  |
| --- |
| package main import ("fmt")  func main() {  \_\_ i:=0; i < 6; \_\_ {  fmt.Println(i)  } } |

**Kunci Jawaban**

Jawaban akan ditulis dengan garis bawah “Seperti ini”.

1.kode lengkap soal nomor satu.

|  |
| --- |
| package main  import ("fmt")    func main() {  Fmt.Println ("Hello, World!")  } |

2.kode lengkap soal nomor dua,

|  |
| --- |
| package main  import ("fmt")    func main() {    // this is a comment  fmt.Println("Hello World!")  } |

3.kode lengkap soal nomor tiga.

|  |
| --- |
| package main  import ("fmt")    func main() {  /\*These lines  are comments \*/    fmt.Println("Hello World!")  } |

4.kode lengkap soal nomor empat.

|  |
| --- |
| package main import ("fmt")  func main() {  var MyNum = 50  fmt.Println(myNum)  } |

5.Kode lengkap soal nomor lima. (Eh ada wildan oawkoawkokawokawok)

|  |
| --- |
| package main import ("fmt")  func main() {  var myNum int= 90  var myWord Hello= "Hello"  var myBool boolean= true } |

6.kode lengkap soal nomor enam.

|  |
| --- |
| package main import ("fmt")  func main() {  var Cars = [4]String{"Volvo", "BMW", "Ford", "Mazda"}  fmt.Print(Cars) } |

7.Kode engkap soal nomor tujuh.

|  |
| --- |
| package main import ("fmt")  func main() {  fmt.Print(10\*5) } |

8.kode lengkap soal nomor delapan.

|  |
| --- |
| package main import ("fmt")  func main() {  var x = 50  var y = 10  if x > y {  fmt.Print("Hello World")  } } |

9.kode lenkap soal nomor sembilan.

|  |
| --- |
| package main import ("fmt")  func main() {  var day = 2  switch Day {  case(1):  fmt.Print("Saturday")  case(2):  fmt.Print("Sunday")   } } |

10.kode lengkap soal nomor sepuluh.

|  |
| --- |
| package main import ("fmt")  func main() {  for i:=0; i < 6; i++ {  fmt.Println(i)  } } |

# **DAFTAR PUSTAKA**

Codecademy Team. (t.thn.). *Method and Function | Codecademy*. Diambil kembali dari codecademy.com: https://www.codecademy.com/article/fwd-js-methods-functions

Fielding, R., & Taylor, R. (2002). Principled Design of the Modern. *ACM Transactions on Internet Technology*, 115-150.

Google. (t.thn.). *Google Docs | Google For Developer*. Diambil kembali dari developer.google.com: https://developers.google.com/docs

Google. (t.thn.). *Google Sheets | Google For Developer*. Diambil kembali dari developers.google.com: https://developers.google.com/sheets

Google LLC. (t.thn.). *Google Cloud APIs*. Diambil kembali dari cloud.google.com: https://cloud.google.com/apis/docs/overview

Google LLC. (t.thn.). *Products and Services*. Diambil kembali dari cloud.google.com: https://cloud.google.com/products/

Loshin, D. (2021, Maret). *What are Data Structures? - Definition from WhatIs.com*. Diambil kembali dari techtarget.com: https://www.techtarget.com/searchdatamanagement/definition/data-structure#:~:text=A%20data%20structure%20is%20a,to%20suit%20a%20specific%20purpose.

Nandy. (t.thn.). *Kenalan Dengan Penemu Aljabar dan Algoritma*. Diambil kembali dari gramedia.com: https://www.gramedia.com/literasi/penemu-aljabar-algoritma/

Reddy, M. (2011). *API Design for C++.* Burlington: Morgan Kaufmann.

Riko. (2022, Juni 23). *Pengertian Git,Git Bash dan Cara Menginstalnya*. Diambil kembali dari plimbi.com: https://www.plimbi.com/article/177691/pengertian-gitgit-bash-dan-cara-menginstalnya

Roni, A. (2021, April 29). *Pengertian Variabel Dalam Pemrograman Dasar*. Diambil kembali dari aderoni.com: https://aderoni.com/pemrograman/pengertian-variabel-dalam-pemrograman/

Wali, M., Sepriano, Indra, D. G., Nengsih, T. A., Antoni, Hasanuddin, . . . Rachmat. (2023). *PENGANTAR 15 BAHASA PEMROGRAMAN TERBAIK DI MASA DEPAN.* Kota Jambi: PT. Sonpedia Publishing Indonesia.

Yse, D. L. (2020, Januari 22). *Essential Program | Control Structure*. Diambil kembali dari towardsdatascience.com: https://towardsdatascience.com/essential-programming-control-structures-2e5e73285df4#:~:text=Control%20Structures%20are%20the%20blocks,if%20a%20condition%20is%20met.

# **TENTANG PENULIS**

Rolly Maulana Awangga, lahir di Kota Indramayu pada tanggal 10 November 1986. Pendidikan tingkat dasar hingga menengah ditempuh di Indramayu. Mulai merantau sejak SMA, melatih kemandiriannya di SMANDA Cirebon dengan aktif organisasi PPS Betako Merpati Putih, Pengurus OSIS dan Pendiri Dewan Keamanan Sekolah. Melanjutkan pendidikan S1 di STT Telkom, S2 di IT Telkom Bandung. Selama kuliah aktif sebagai TLH Telkom, pengurus Klub Linux Bandung, Pengurus Bandung Kota Blogger, Pendiri Saung IT dan wartawan Pikiran Rakyat. Menjadi tenaga ahli dan konsultan di aplikasi SDDKN Sekretariat Negara, Aplikasi Kementrian Hukum dan Ham, Team DevOps Pekan Olahraga Nasional, Cloud Architect Aplikasi Asesment Madrasah Kementrian Agama.

Waskitho Cito Adiwiguno, lahir di kota Jakarta pada tanggal 18 September 2004. Pendidikan tingkat dasar ditempuh di Sekolah Dasar Santalusia Bogor serta pendidikan tingkat mengah ditempuh di SMPN 9 Kota bekasi hingga pendidikan akhir ditempuh di SMAN 22 Kota Bekasi,Suka bermusik dan ikut berorganisasi kesiswaan seperti PMR(Palang Merah Remaja), GPS(Generasi Penggiat Seni) selama masa sekolah menengah akhir. Melanjutkan pendidikan D4 di Universitas Logistic & Bisnis Internasional 

( ULBI).

Muhammad Wildan Khalilurrahman, lahir di Bandung pada tanggal 19 Desember 2003. Pendidikan tingkat dasar ditempuh di SDN Sinarbahagia, Bandung dan SDN Cadasngampar, Sukabumi. Pendidikan Menengah ditempuh SMPT Darul ‘Amal dan SMAT Darul’Amal, Sukabumi. Selama sekolah menengah, aktif di Organisasi Santri Terpadu Darul ‘Amal. Saat ini sedang melanjutkan pendidikan D4 di Universitas Logistik dan Bisnis Internasional (ULBI), Bandung.

