

Tutorial Membangun Aplikasi Website Pendidikan Otomotif

Rachma Nurhaliza Parindra

Adam Ghafara

Roni Habibi

|  |  |
| --- | --- |
| 2023 | **PT. Penerbit Buku Pedia**  **2024** |

**TUTORIAL MEMBANGUN APLIKASI WEBSITE PENDIDIKAN OTOMOTIF**

Menggunakan Golang dan JavaScript

***Penulis:***

Rachma Nurhaliza Parindra

Adam Ghafara

Roni Habibi

***ISBN:***

***Editor:***

Roni Habibi

***Penyunting:***

Rachma Nurhaliza Parindra

Adam Ghafara

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

Rachma Nurhaliza Parindra

Adam Ghafara

***Font:***

Calibri

***Penerbit:***

PT.Penerbit Buku Pedia

***Redaksi:***

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

Kec. Parongpong, Kab. Bandung Barat 40559

Telp. 628-775-2000-300

Email : [penerbit@bukupedia.co.id](mailto:penerbit@bukupedia.co.id)

***Distributor:***

Informatics Research Center

Jl. Sariasih No.54

Bandung 40151

Email : [irc@ulbi.ac.id](mailto:irc@ulbi.ac.id)

Cetakan Pertama, 2024

Hak cipta dilindungi undang-undang

Dilarang memperbanyak karya tulis dari penerbit

# PRAKARTA

O

tomotif merupakan salah satu topik menarik bagi para pecinta mobil, motor, dan kendaraan bermesin lainnya. Selain itu, Otomotif menjadi bahan topik bagi mereka yang ingin mengenal lebih tentangnya, baik dalam perawatan kendaraan, tips dan trik, dan juga informasi terkini. Namun, informasi pada media internet yang membahas tentang otomotif terbilang sedikit dan sangat terbatas. Hal ini membuat beberapa dari mereka kesulitan mencari informasi tepat dan ter aktual dalam otomotif.

Buku ini bertujuan untuk memberikan pemahaman tentang pembuatan website berteknologi Golang dan JavaScript untuk mendukung pemahaman yang lebih mendalam tentang teknologi tersebut. Dengan menggunakan Golang sebagai bahasa pemrograman backend, kalian dapat menciptakan aplikasi dasar dengan kinerja yang tinggi, kehandalan, dan kemampuan untuk mengelola beban kerja dengan efisien. Meskipun demikian, penggunaan JavaScript pada sisi frontend memungkinkan pembuatan antarmuka pengguna yang dinamis dan interaktif. Kombinasi kedua teknologi ini menjadi landasan untuk membangun pengalaman belajar yang kuat, yang menggabungkan komponen fungsionalitas backend yang canggih dengan responsivitas antarmuka pengguna yang menarik.

Oleh karena itu, Tutorial ini bertujuan untuk memberikan wawasan mendalam tentang pembuatan website dengan mengambil tema Otomotif sebagai bahan tutorial nya. Selain itu, Tutorial ini juga bertujuan untuk mendorong inovasi dalam penggunaan teknologi dalam pendidikan.

Buku ini juga memiliki source code sebagai bahan ajar yang dapat kalian akses menggunakan link github berikut:

<https://github.com/berkatauto>

# DAFTAR ISI

[PRAKARTA I](#_Toc156522626)

[DAFTAR ISI II](#_Toc156522627)

[DAFTAR GAMBAR V](#_Toc156522628)

[BAB I PENDAHULUAN 1](#_Toc156522629)

[A. PENGENALAN APLIKASI 1](#_Toc156522630)

[B. TUJUAN DAN FUNGSI TUTORIAL 2](#_Toc156522631)

[C. MANFAAT TUTORIAL 3](#_Toc156522632)

[BAB II PENGENALAN OTOMOTIF 4](#_Toc156522633)

[A. KONSEP DASAR OTOMOTIF 4](#_Toc156522634)

[B. PERKEMBANGAN OTOMOTIF SAAT KINI 4](#_Toc156522635)

[C. KEUNGGULAN DARI WEBSITE PENDIDIKAN OTOMOTIF 5](#_Toc156522636)

[BAB III DASAR PEMROGRAMAN 6](#_Toc156522637)

[A. BAHASA PEMROGRAMAN 6](#_Toc156522638)

[a. HTML 6](#_Toc156522639)

[b. CSS 7](#_Toc156522640)

[c. JavaScript 7](#_Toc156522641)

[d. Golang (GO) 8](#_Toc156522642)

[BAB IV INSTALASI DAN PENYIAPAN APLIKASI PEMBANGUN 11](#_Toc156522643)

[A. APLIKASI PEMBANGUN YANG DIBUTUHKAN 11](#_Toc156522644)

[a. Visual Studio Code 11](#_Toc156522645)

[b. MongoDB Compass 13](#_Toc156522646)

[c. Postman 14](#_Toc156522647)

[d. Golang 16](#_Toc156522648)

[e. Git Bash 18](#_Toc156522649)

[B. REKOMENDASI APLIKASI PEMBANGUN LAINNYA 22](#_Toc156522650)

[BAB V PEMBUATAN APLIKASI WEBSITE 23](#_Toc156522651)

[A. PEMBUATAN REPOSITORY GITHUB 23](#_Toc156522652)

[B. PEMBUATAN FRONTEND 25](#_Toc156522653)

[a. Memasukkan Template Kedalam Repository 26](#_Toc156522654)

[b. Upload Repository 27](#_Toc156522655)

[c. Jalankan frontend menjadi Pages di github 29](#_Toc156522656)

[C. PEMBUATAN BACKEND 30](#_Toc156522657)

[a. Membuat Package 30](#_Toc156522658)

[b. Membuat Endpoint API 31](#_Toc156522659)

[c. Publish Repository ke Library Golang 31](#_Toc156522660)

[D. PEMBUATAN API CLOUD 31](#_Toc156522661)

[E. PEMBUATAN DATABASE 31](#_Toc156522662)

[a. Membuat Akun dan Membuat Database MongoDB 31](#_Toc156522663)

[b. Menyambungkan Database Dengan MongoDB Compass 32](#_Toc156522664)

[c. Menyambungkan MongoDB dengan Backend 35](#_Toc156522665)

[F. MEMBUAT JAVASCRIPT 40](#_Toc156522666)

[a. Membuat Variable 40](#_Toc156522667)

[b. Membuat 40](#_Toc156522668)

[c. Publish Versi Javascript 40](#_Toc156522669)

[d. Jalankan Javascript kedalam frontend 40](#_Toc156522670)

[DAFTAR PUSTAKA V](#_Toc156522671)

[GLOSARIUM VI](#_Toc156522672)

[INDEKS X](#_Toc156522673)

[TENTANG PENULIS XI](#_Toc156522674)

# DAFTAR GAMBAR

[Gambar 1 Halaman Depan Visual Studio Code 12](#_Toc156519154)

[Gambar 2 12](#_Toc156519155)

[Gambar 3 14](#_Toc156519156)

[Gambar 4 14](#_Toc156519157)

[Gambar 5 16](#_Toc156519158)

[Gambar 6 16](#_Toc156519159)

[Gambar 7 17](#_Toc156519160)

[Gambar 8 17](#_Toc156519161)

[Gambar 9 17](#_Toc156519162)

[Gambar 10 18](#_Toc156519163)

[Gambar 11 19](#_Toc156519164)

[Gambar 12 19](#_Toc156519165)

[Gambar 13 20](#_Toc156519166)

[Gambar 14 20](#_Toc156519167)

[Gambar 15 22](#_Toc156519168)

[Gambar 16 24](#_Toc156519169)

[Gambar 17 25](#_Toc156519170)

[Gambar 18 25](#_Toc156519171)

[Gambar 19 26](#_Toc156519172)

[Gambar 20 26](#_Toc156519173)

[Gambar 21 28](#_Toc156519174)

[Gambar 22 28](#_Toc156519175)

[Gambar 23 29](#_Toc156519176)

# BAB I PENDAHULUAN

## PENGENALAN APLIKASI

Dalam teknologi informasi dalam dunia otomotif, ada berbagai perilaku masyarakat yang berbeda dalam memandang dunia otomotif ini. Ada yang menggunakannya sebagai media informasi berbagi ilmu dalam otomotif, adapun mereka yang menggemari dunia otomotif ini hingga berbagi pengalaman mereka dalam hal tersebut, dan adapun mereka yang hanya mencari tahu bagaimana mereka dapat mencari jalan keluar dalam permasalahan kendaraan mereka melalui teknologi informasi teknologi tersebut. Oleh karenanya, adanya aplikasi website Pendidikan Otomotif sebagai bagian dari teknologi informasi diharapkan menjadi pemudah dalam hubungan masyarakat yang menggemari dunia otomotif, sekaligus mereka yang ingin menemukan jalan keluar dalam dunia otomotif tersebut.

Aplikasi Website Pendidikan Otomotif merupakan website inovatif yang bertujuan untuk mengefektifkan pendidikan otomotif di era digital saat ini. Mengingat tantangan yang dihadapi oleh para penggiat otomotif, seperti kurangnya akses terhadap ilmu dan informasi terkini dan kurangnya pengalaman praktis, website ini dirancang untuk mengatasi kendala tersebut. Berkat pendekatan gabungan teknologi Internet dan multimedia, aplikasi ini menyediakan platform pembelajaran yang kaya dan interaktif. Dengan cara ini, pengguna dapat dengan mudah mengakses materi pelatihan otomotif terkini dan relevan. Website ini tidak hanya memberikan pemahaman teoritis tentang konsep otomotif tetapi juga menawarkan simulasi realistis dan tes latihan yang memungkinkan pengguna mengasah keterampilan virtualnya.

Website memiliki beberapa fasilitas yang dapat di akses, seperti Beranda Berita yang dapat di akses oleh semua, Dashboard, serta akses untuk menulis satu artikel, dan sebagainya.

## TUJUAN DAN FUNGSI TUTORIAL

Tujuan dari tutorial ini yaitu untuk menyediakan panduan yang komprehensif dan terperinci bagi pengembang, pengguna, dan pihak terkait lainnya. Tutorial ini bertujuan untuk memfasilitasi pemahaman yang lebih baik tentang struktur, logika, dan fungsionalitas aplikasi, serta menyediakan panduan praktis bagi pengguna untuk memanfaatkan aplikasi secara efektif.

Fungsi utamanya adalah sebagai referensi yang jelas dan terstruktur yang mencakup informasi tentang struktur kode, penggunaan teknologi dan metode tertentu dalam implementasi aplikasi, petunjuk instalasi, dan konfigurasi.

Tujuan tutorial :

1. Paduan Komperehensif

Bertujuan untuk menjelaskan secara komprehensif tentang fungsi bagaimana aplikasi pendidikan otomotif yang dibangun menggunakan Golang dan JavaScript.

1. Panduan Pengguna

Membantu pengembang dalam pengembangan, pemeliharaan, dan peningkatan aplikasi. Tujuan utama lainnya adalah memberikan panduan praktis bagi pengguna yang ingin memanfaatkan aplikasi. Ini mencakup petunjuk instalasi, konfigurasi, serta cara penggunaan fitur-fitur yang ada dalam aplikasi.

1. Referensi Teknis

Menjadi sumber referensi yang terperinci bagi pengembang. Dokumentasi ini akan menjelaskan struktur kode, pemilihan bahasa pemrograman, penggunaan framework, dan teknologi terkait lainnya yang digunakan dalam pembuatan aplikasi.

Fungsi tutorial :

1. Panduan Implementasi

Berfungsi sebagai panduan implementasi yang membantu pengembang untuk memahami dan menerapkan aplikasi dengan efisien menggunakan Golang dan JavaScript.

1. Dukungan Pengguna Akhir

Berfungsi sebagai sumber informasi untuk pengguna akhir aplikasi. Ini memberikan pemahaman yang lebih baik tentang cara menggunakan aplikasi secara efektif.

## MANFAAT TUTORIAL

Tutorial menjadi sumber pengetahuan yang berharga bagi pembelajar, praktisi, pengembang, dan industri otomotif secara keseluruhan. Hal ini memungkinkan akses yang lebih baik terhadap informasi teknis terkait penerapan teknologi terkini dalam dunia otomotif.

Tutorial ini berperan dalam mendorong perkembangan industri dengan memberikan wawasan mendalam tentang perancangan website yang dapat dioptimalkan dalam aplikasi pendidikan. Tutorial ini dapat menginspirasi inovasi baru, pengembangan produk yang lebih canggih, dan juga memperluas cakupan pengetahuan bagi komunitas pengembang dan praktisi otomotif.

Berikut ini beberapa manfaat dari dokumentasi :

1. Menjadi sumber Pembelajaran

Memberikan akses kepada informasi yang komprehensif tentang aplikasi teknologi dalam industri otomotif. Hal ini membantu mereka dalam memahami konsep-konsep penting dan perkembangan terbaru dalam bidang ini. Menjadi sumber bahan ajar yang berharga dalam menyampaikan materi tentang integrasi teknologi dalam pembelajaran otomotif. Dengan dokumentasi ini, pendidik dapat menyajikan informasi yang relevan dan terkini.

1. Idustri Otomotif Secara Umum

Mendorong inovasi dan perkembangan dalam industri otomotif dengan memanfaatkan teknologi terbaru. Dokumentasi ini dapat menginspirasi pengembangan produk dan layanan baru yang lebih canggih dan efisien.

# BAB II PENGENALAN OTOMOTIF

## KONSEP DASAR OTOMOTIF

Otomotif mencakup serangkaian konsep dasar yang penting untuk memahami cara kerja kendaraan bermotor. Salah satunya adalah mesin, jantung dari mobil. Mesin menggunakan energi dari bahan bakar untuk menciptakan gerakan mekanis yang membantu kendaraan bergerak. Selain itu, terdapat sistem transmisi yang mentransfer tenaga dari mesin ke roda kendaraan dengan mengatur kecepatan dan torsi. Sistem pembakaran internal mesin bekerja dengan membakar campuran bahan bakar dan udara untuk menghasilkan energi. Komponen lainnya antara lain transmisi, suspensi, rem, bahan bakar, pendingin, kelistrikan, kemudi, intake, dan sistem transmisi yang semuanya bekerja sama untuk membantu kendaraan tampil maksimal. Memahami konsep-konsep ini membantu merawat, memperbaiki, dan mengoperasikan kendaraan dengan lebih efektif.

## PERKEMBANGAN OTOMOTIF SAAT KINI

Saat ini, industri otomotif sedang mengalami transformasi yang signifikan. Perkembangan teknologi telah melahirkan inovasi-inovasi besar di bidang kendaraan bermotor. Mobilitas listrik semakin menarik perhatian masyarakat seiring dengan semakin banyaknya jumlah mobil listrik yang diproduksi oleh banyak pabrikan besar. Selain itu, konsep kendaraan otonom atau self-driving juga semakin mendapat perhatian, dengan dikembangkannya sistem kecerdasan buatan yang memungkinkan kendaraan mengemudi sendiri melalui sensor canggih dan teknologi pemrosesan data. Aspek keselamatan juga ditekankan dengan penerapan teknologi pencegahan kecelakaan, seperti sistem pengereman darurat otomatis dan sistem bantuan pengemudi yang semakin canggih. Selain itu, konsep berbagi mobil dan transportasi berbasis layanan (seperti persewaan mobil jangka pendek melalui aplikasi) semakin populer, sehingga mengubah cara masyarakat menggunakan dan memiliki kendaraan. Hal ini merupakan bagian dari

perubahan besar dalam industri otomotif yang kini bertujuan untuk meningkatkan kenyamanan, efisiensi, keselamatan, dan keberlanjutan.

## KEUNGGULAN DARI WEBSITE PENDIDIKAN OTOMOTIF

Website pendidikan otomotif memiliki manfaat yang signifikan dalam menyediakan akses universal terhadap pengetahuan otomotif. Dengan kemudahan akses melalui Internet, website ini tidak hanya menyajikan materi pembelajaran terstruktur tetapi juga memungkinkan pembaruan konten secara cepat, mencakup inovasi dan perkembangan terkini di industri mobil. Interaktivitas konten, seperti video instruksional dan simulasi, memungkinkan pemahaman yang lebih baik, sementara platform online juga mendukung forum diskusi, kolaborasi, dan pengujian yang memperkaya pengalaman belajar. Fleksibilitas untuk belajar kapan saja, di mana saja, serta penyesuaiannya, menjadikan situs web pendidikan otomotif sebagai sumber daya yang sangat berharga bagi pelajar, profesional, dan penggemar otomotif.

1. Aksesibilitas di seluruh dunia

Melalui situs web, siapa pun dari berbagai belahan dunia dapat mengakses informasi otomotif dan materi pendidikan, selama mereka memiliki koneksi Internet. Hal ini memungkinkan penyebaran pengetahuan otomotif secara luas.

1. Konten interaktif

Situs web dapat menyediakan konten interaktif seperti video instruksional, simulasi, atau animasi yang membantu siswa atau pengguna lebih memahami konsep otomotif.

1. Pembaruan yang mudah dan ulasan

Situs ini dapat diperbarui secara berkala untuk memuat informasi terkini, teknologi terkini, atau perkembangan industri otomotif. Hal ini memastikan bahwa materi yang disampaikan tetap relevan dan terkini.

# BAB III DASAR PEMROGRAMAN

## BAHASA PEMROGRAMAN

Bahasa pemrograman adalah seperangkat aturan dan instruksi yang digunakan untuk berkomunikasi dengan komputer. Ini adalah cara manusia merancang algoritme, mengatur data, dan menginstruksikan komputer untuk melakukan serangkaian tindakan tertentu. Setiap bahasa pemrograman memiliki sintaks dan aturannya sendiri yang memungkinkan pemrogram mengekspresikan ide dan tujuan mereka dalam format yang dapat dipahami mesin.

Dari bahasa tingkat rendah seperti bahasa mesin dan perakitan hingga bahasa tingkat tinggi seperti Python, JavaScript, dan C++, masing-masing bahasa memiliki kegunaannya masing-masing dalam mengembangkan berbagai aplikasi, mulai dari perangkat lunak hingga pengembangan web dan buatan. Intelijen. Bahasa pemrograman adalah landasan inovasi teknologi modern, memungkinkan kita menciptakan solusi yang lebih kompleks dan efektif di dunia digital. Berikut ini merupakan bahasa pemrograman yang digunakan :

### HTML

Menurut Bimo Sunarfrihantono (2002), HTML (Hypertext Markup Language) adalah format yang digunakan untuk membuat dokumen dan aplikasi yang berjalan di halaman web. Oleh karena itu, sebelum Anda dapat membuat program aplikasi yang berjalan di halaman web, Anda harus mengenal dan menguasai HTML. “Hypertext Merkup Languange (HTML) merupakan bahasa standar yang digunakan untuk menampilkan halaman web,” kata Hidayatullah dan Kawistara (2015) dalam jurnal Fitri Ayu dan Nia Permata Sari (ISSN:2549-0222).

Berikut ini kelebihan yang dimiliki HTML :

* + - 1. Bahasa yang dipakai secara dan mempunyai banyak sumber serta komunitas yang besar.
      2. Dapat dijalankan setiap *web browser*.
      3. *Open-source* yang digunakan secara gratis.
      4. Bahasa *markup* yang tertata rapi dan konsisten.
      5. Dapat diintegrasikan dengan bahasa pemrograman yang dapat berjalan di *server side* seperti *PHP, Asp, Java* dan lain-lain.

Berikut ini kekurangan yang dimiliki HTML :

1. *HTML* dipakai untuk membuat halaman *website* statisdan sederhana.
2. *HTML* tidak dapat menjalankan *logic.*

### CSS

*Cascading Style Sheets* (*CSS)* merupakan bahasa pemrograman yang berfungsi sebagai web design. Penggunaan *CSS* untuk membuat tampilan web yang bagus dan menarik.

Berikut Ini merupakan kelebihan dari *CSS* yaitu :

* + 1. Memisahkan desain dengan konten situs web.
    2. Mengelola desain dengan cara efisien.
    3. Lebih mudah untuk mengunduh karena lebih ringan dalam ukuran file.
    4. CSS dapat digunakan di banyak halaman web.

Berikut kekurangan dari *CSS* yaitu :

* + - 1. Beberapa browser tidak mendukung css (browser lama).
      2. Butuh waktu lebih lama untuk membuat codingan css.
      3. Masih banyaknya bug atau error dalam css.

### JavaScript

Bahasa pemrograman JavaScript diresmikan pada tahun 1996 dan menjadi bahasa pemrograman ECMAScript. Dua tahun kemudian tepatnya tahun 1998, bahasa ECMAScript menjadi ECMAScript 2. Dan pada tahun 1999, ECMAScript 2 menjadi ECMAScript 3. ECMAScript 3 kembali dikembangkan menjadi bahasa pemrograman JavaScript yang kita kenal selama ini. Dari tahun ke tahun, bahasa JavaScript mengalami perkembangan yang pesat. Pada tahun 2016, 92% situs web di seluruh dunia menggunakan bahasa pemrograman JavaScript. Dalam 20 tahun terakhir bahasa ini memiliki banyak fungsi, apalagi bagi para pengembang website sering kita mendengar tentang bahasa pemrograman JavaScript.

JavaScript merupakan bahasa pemrograman tingkat tinggi yang kini menjadi bahasa pemrograman utama bagi pengembang web bersama dengan HTML (Hypertext Markup Language) dan CSS (Cascading Style Sheets). JavaScript adalah bahasa pemrograman yang digunakan dalam mengembangkan situs web agar lebih dinamis dan interaktif.

Berikut Kelebihan javascript :

1. Fleksibilitas tinggi

JavaScript adalah bahasa pemrograman yang sangat fleksibel. Hal ini memungkinkan pengembang menggunakan metode berbeda untuk menyelesaikan tugas yang sama.

1. Cocok untuk pengembangan web

Karena JavaScript terintegrasi dengan baik ke dalam pengembangan web, JavaScript memungkinkan interaksi langsung dengan elemen HTML, menjadikannya pilihan yang baik untuk membuat halaman web interaktif.

1. Berfungsi lintas platform

JavaScript tidak terbatas pada platform tertentu dan dapat berjalan di berbagai sistem operasi dan perangkat.

Berikut Kekurangan javascript :

1. Performa

Meskipun JavaScript terus mengalami peningkatan, performanya terkadang dapat menjadi masalah dalam aplikasi yang sangat besar atau kompleks.

1. Ketergantungan pada koneksi Internet

Beberapa pustaka JavaScript dan fitur web modern memerlukan koneksi Internet yang stabil, yang dapat membatasi penggunaan dalam situasi di mana konektivitas Internet terbatas.

1. Keterbatasan Akses ke Fungsi Perangkat Keras

JavaScript di browser memiliki keterbatasan dalam mengakses fungsi perangkat keras langsung (seperti membaca file dari sistem), yang mengharuskan penggunaan API yang disediakan browser.

### Golang (GO)

Go merupakan bahasa pemrograman yang dikembangkan oleh Google pada tahun 2009. Bahasa ini dirancang untuk fokus pada kesederhanaan, kejelasan, dan efisiensi dalam pengembangan perangkat lunak. Go menonjol karena konkurensi dan goroutinenya yang memungkinkan eksekusi paralel tugas-tugas ringan, mendukung pengembangan perangkat lunak secara bersamaan, terukur, dan efisien.

Keuntungan lainnya termasuk kompilasi yang cepat, dukungan untuk pemrograman berorientasi objek tanpa terlalu banyak kerumitan, dan kemampuan untuk membuat perangkat lunak yang dapat berjalan di berbagai platform dengan sedikit modifikasi. Go telah banyak digunakan dalam berbagai aplikasi termasuk pengembangan aplikasi web, server backend, pemrosesan data, dan banyak proyek teknologi tinggi lainnya.

Kelebihan dari Go yaitu :

1. Kinerja tinggi

Go memberikan kinerja yang cepat dan efisien, terutama saat mengompilasi kode.

1. Konkurensi tinggi

Dukungan goroutine bawaan memungkinkan eksekusi tugas paralel (paralel) dengan mudah, mendukung aplikasi yang memerlukan pemrosesan bersamaan.

1. Manajemen memori yang baik

Go memiliki manajemen memori yang baik, mengurangi risiko kebocoran memori ingat dan membuat aplikasi lebih stabil.

Kekurangan dari Go yaitu :

1. Kurangnya keragaman perpustakaan

Meskipun perpustakaan standarnya bagus, Go mungkin kekurangan keragaman perpustakaan dan alat dibandingkan dengan bahasa lain yang lebih mapan.

1. Sebuah bahasa yang masih dalam pengembangan

Sebagai bahasa yang relatif baru, ada adalah ekspektasi rendah terhadap fitur atau konvensi bahasa pemrograman yang mungkin belum tersedia atau didefinisikan sepenuhnya di Go.

1. Komunitas Lebih Kecil

Meskipun komunitas Go terus berkembang, ukurannya masih lebih kecil dibandingkan komunitas yang lebih mapan bahasa pemrograman. Sehingga mendapatkan bantuan atau dukungan mungkin memerlukan lebih banyak usaha.

1. Kurangnya fitur yang diharapkan

Beberapa fitur digunakan oleh pengembang yang akrab dengan bahasa tersebut Harapkan pemrograman lain kurang tersedia atau bahkan sedang dikembangkan di Go.

# BAB IV INSTALASI DAN PENYIAPAN APLIKASI PEMBANGUN

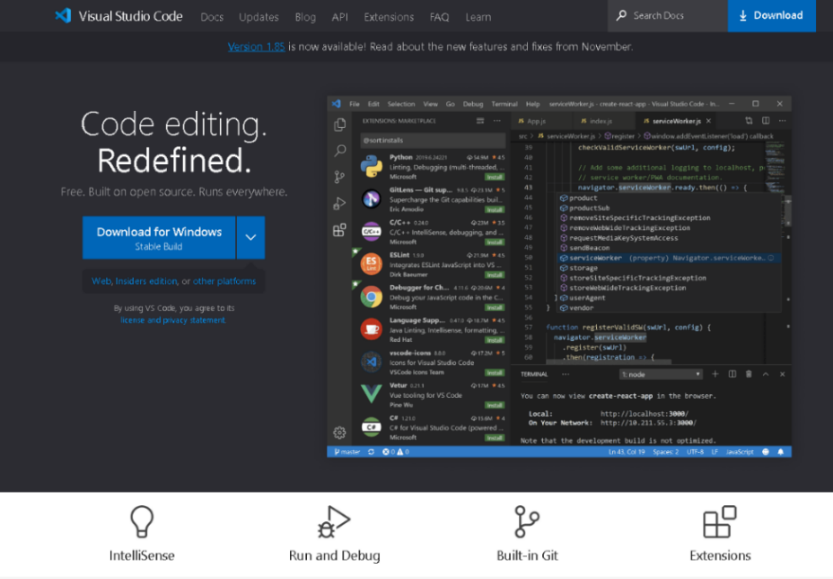
## APLIKASI PEMBANGUN YANG DIBUTUHKAN

Pada kegiatan pembuatan aplikasi ini, kita membutuhkan aplikasi yang akan digunakan sebagai pembuatan dalam aplikasi Pendidikan otomotif ini. Berikut beberapa aplikasi yang dibutuhkan.

**Catatan :** Aplikasi pembangun yang jelaskan berikut hanya digunakan pada sistem operasi Windows 64-bit saja, sehingga tidak ada gambaran untuk penginstalan pada operasi sistem lainnya atau menggunakan Windows 32-bit.

### Visual Studio Code

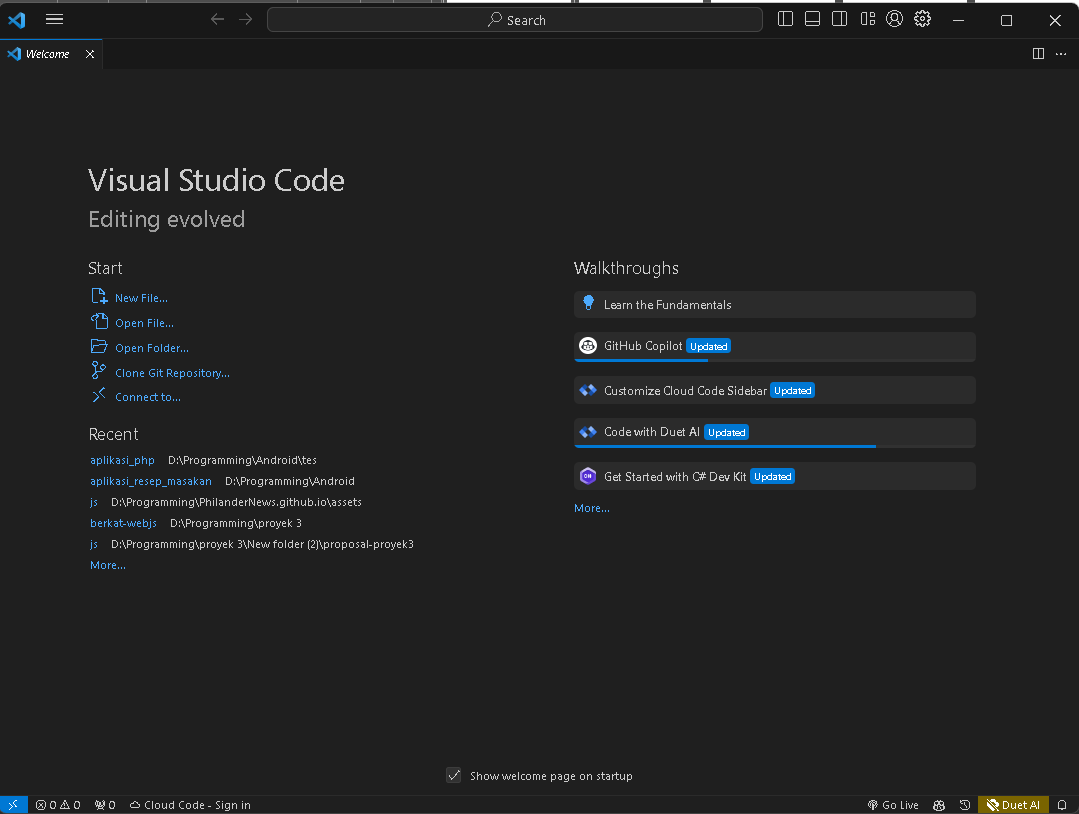
Visual Studio Code merupakan Aplikasi pengeditan kode yang umum digunakan oleh developer-developer sebagai aplikasi perancangan sistem. Untuk pemasangan aplikasi ini, silahkan download instalasi nya pada situs [code.visualstudio.com](https://code.visualstudio.com).



Gambar 1 Halaman Depan Visual Studio Code

Klik pada “*Download for Windows”*. Lalu secara otomatis akan mendownload instalasinya dan kalian langsung diarahkan pada halaman dokumentasi. Silahkan di ikuti tata cara instalasi nya selagi kalian menunggu mengunduh aplikasi tersebut.

Aplikasi yang telah di instal terlihat seperti berikut:



Gambar 2 Tampilan UI Visual Studio Code

Selanjutnya, kalian dapat menyambungkan akun github dengan aplikasi ini. Dengan begitu, proses commit dapat lebih mudah dan tidak memakan waktu banyak untuk memasukkan command untuk commiting.

Selain itu, kalian dapat menyambungkan akun github kalian untuk menjalankan Github Copilot yang merupakan Chat AI yang dapat memudahkan kegiatan ngoding kalian disini. Github Copilot membutuhkan biaya untuk dapat berlangganan menggunakannya, namun jika akun github kalian memiliki student membership didalamnya, Github Copilot bisa didapatkan secara gratis.

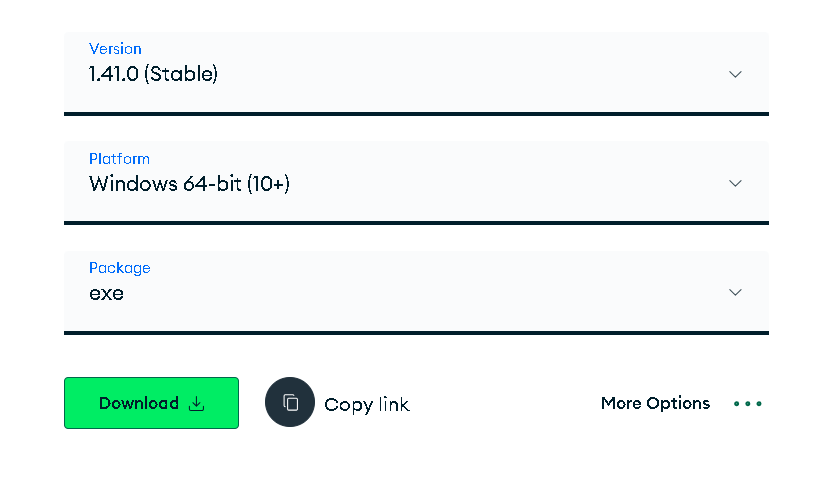
Adapun kalian dapat memasang extension-extension berikut yang dapat memudahkan kalian dalam melakukan Tutorial ini:

1. Live Server
2. Go
3. Open In Browser
4. Prettier
5. Intellisense for CSS
6. IntelliCode

### MongoDB Compass

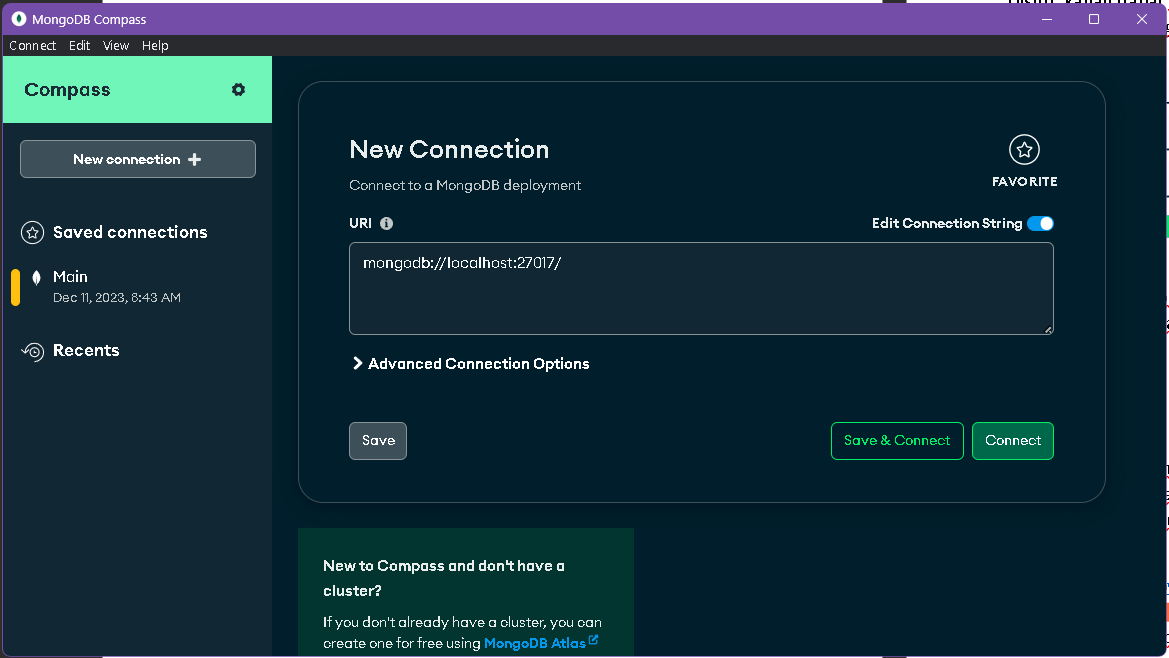
Mongo DB Compass merupakan aplikasi dari Mongo yang dapat mempermudah kita dalam menjangkau database noSQL pada Mongo. Untuk instalasi MongoDB Compass, silahkan masuk kedalam situs berikut <www.mongodb.com/try/download/compass> lalu cari pada bagian “MongoDB Compass Download (GUI)”.

Disitu, kalian dapat memilih versi Compass yang ingin kalian instal. Sebagai rekomendasi, silahkan gunakan 1.41.0 yang terbaru dan sudah stable.



Gambar 3 Pilihan Versi MongoDB Compass

Silahkan unduh aplikasi tersebut, setelahnya, jalankan dan ikuti tahap penginstalannya. Aplikasi yang dijalankan untuk pertama kali, terlihat seperti berikut:

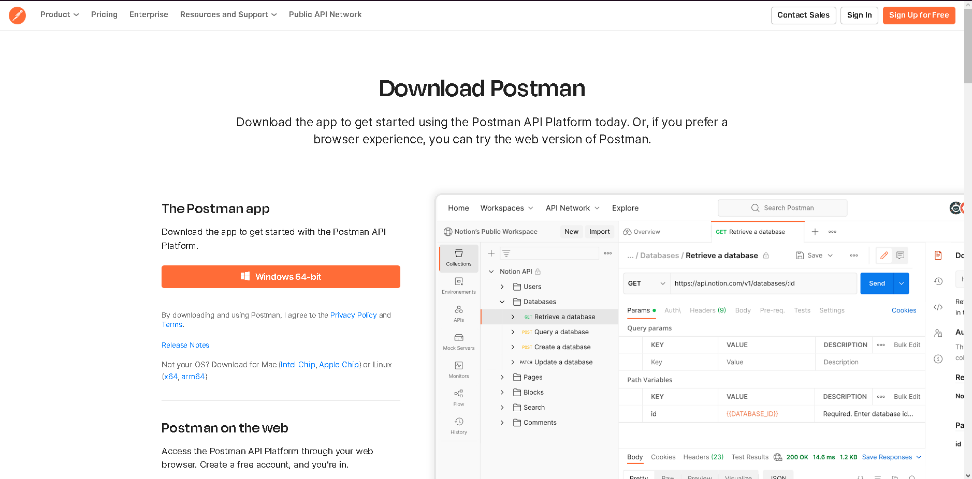


Gambar 4 Tampilan UI MongoDB Compass

### Postman

Postman digunakan sebagai pengujian pada API yang dibuatkan dalam Google Cloud sehingga didapatkan hasil yang dapat digunakan sebagai tolak ukur pada API yang dibuat.

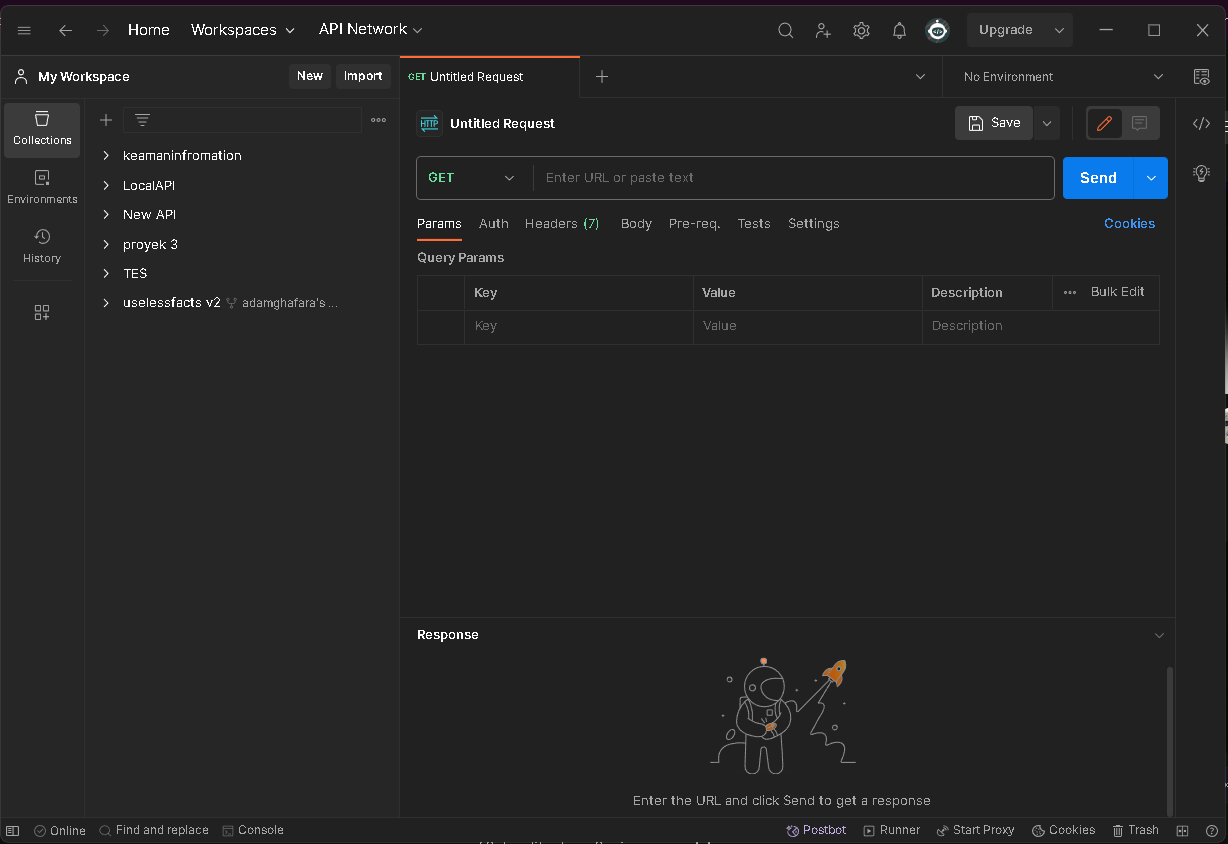
Untuk menginstal Postman, silahkan masuk kedalam situs berikut [www.postman.com/downloads/](https://www.postman.com/downloads/), dan silahkan unduh langsung dengan menekan tombol , silahkan tunggu beberapa saat dan silahkan ikuti tahap penginstalannya.



Gambar 5 Halaman Download Postman

Setelah penginstalan, silahkan jalankan aplikasi. kalian akan diminta untuk login terlebih dahulu untuk mengakses aplikasi Postman ini. Kalian dapat membuat akun Postman menggunakan akun Google atau GitHub kalian untuk di sambungkan dengan aplikasi ini, kemudian login kan pada aplikasi.

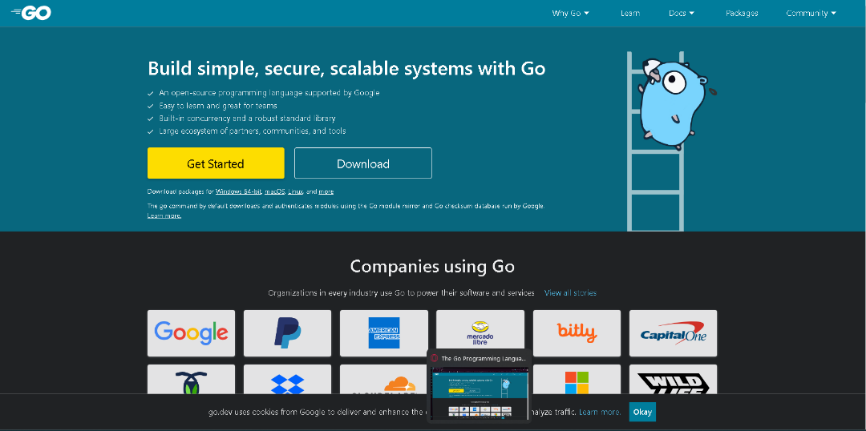
Tampilan aplikasi kurang lebih terlihat seperti berikut:



Gambar 6 Tampilan UI Postman

### Golang

Golang sebagai Bahasa pemrograman utama yang akan kita gunakan sebagai backend pada aplikasi ini. Untuk penginstalan aplikasi ini, silahkan masuk kedalam situs berikut [go.dev](https://go.dev), kemudian pilih download untuk mengunduh aplikasi golang.



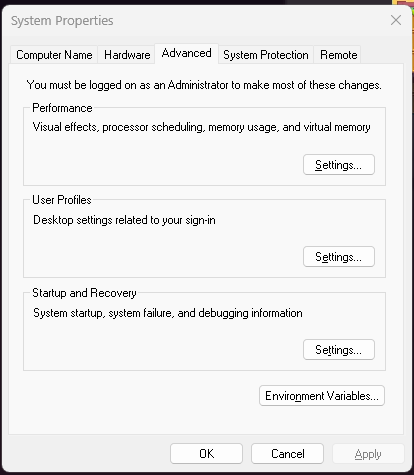
Gambar 7 Halaman Utama Golang

Pada tampilan berikut, silahkan pilih instalasi untuk windows, yang kemudian akan langsung di download oleh browser. Jika sudah, silahkan jalankan aplikasi instalasi golang nya, dan ikuti tahap penginstalannya. Untuk penginstalanya, pastikan kalian menginstalnya pada satu directory saja dan tidak ditempatkan dalam Program Files.

**Contoh: “C:\Go”**

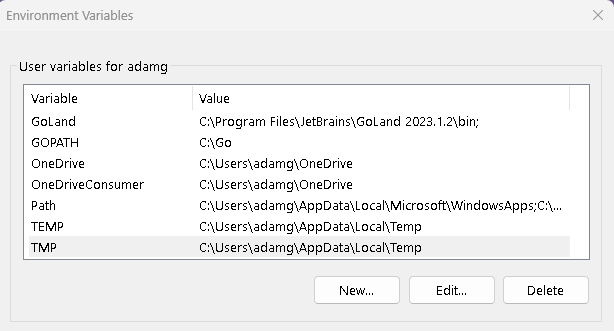
Selanjutnya, kalian perlu menambah environment untuk Golang. Silahkan ikuti tahap berikut:

1. Buka Search dan Cari “Edit the System Environment Variables”
2. Lalu buka “Environment Variables…”



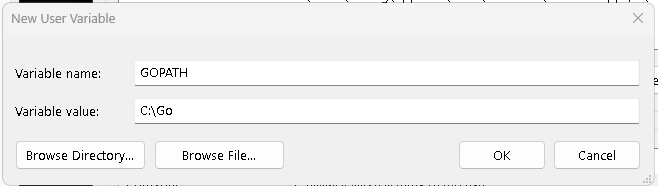
Gambar 8 Environment Variables

1. Pada bagian “User variables for (namauser)”. Silahkan pilih “New…”.



Gambar 9 New Variables

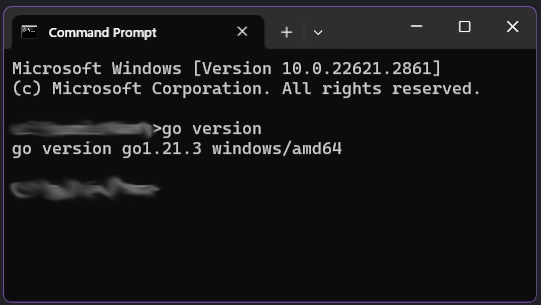
1. Akan muncul jendela baru untuk mengisi variables yang ingin dibuat. Isikan Nama variable dengan “GOPATH”, dan masukkan Value dengan directory.



Gambar 10 New User Variables

1. Save semua dan restart perangkat yang digunakan.

Untuk mengetes apakah Golang terinstal dengan tepat, kita dapat mengecek instalasinya menggunakan Command Prompt (cmd). Silahkan buka aplikasi Command Promt dengan menggunakan “Search” pada windows dan tulis “cmd”. Setelah aplikasi terbuka, silahkan masukkan command “go version” dan eksekusikan. Tampilan hasilnya kurang lebih terlihat seperti berikut:



Gambar 11 Cek Versi Golang

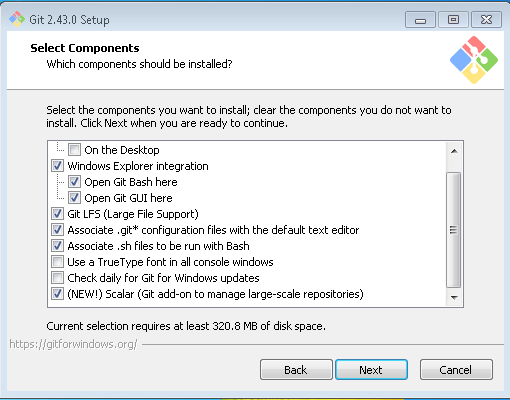
### Git Bash

Git Bash merupakan command prompt alternatif pengganti cmd Windows yang lebih . Git Bash dilengkapi dengan beberapa command yang dapat langsung berinteraksi dengan GitHub sehjngga menjadi aplikasi yang dibutuhkan programmer untuk mengatur repository nya.

Untuk menginstalnya, kalian dapat mengunduh penginstal nya di link situs berikut [www.git-scm.com/downloads](https://www.git-scm.com/downloads), kemudian klik Download, dan pilih Windows 64-bit pada versi Standalone.

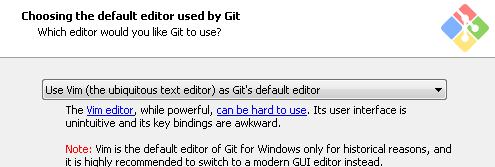
Pada saat penginstalan, pastikan kalian menyetel beberapa opsi penginstalan berikut:

1. Centang beberapa opsi seperti pada gambar berikut:



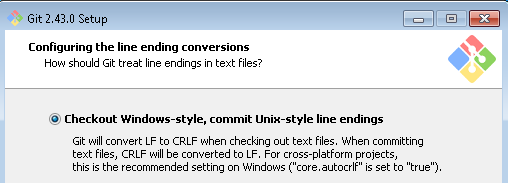
Gambar 12 Setelan Instalan Git Bash

1. Pilih Vim sebagai default Editor. Jangan khawatir, setelah ini kita masih bisa pakai VS code sebagai Code Editor.



Gambar 13 Setelan Default Editor

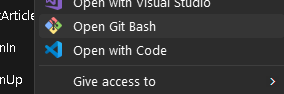
1. Untuk Line Ending Conversion, pilihlah Windows Style.



Gambar 14 Setelan Ending Conversions

1. Opsi lain selain yang dijelaskan sebelumnya, biarkan pada opsi default nya.

Untuk menjalankan Git Bash, silahkan pilih folder yang ingin kalian gunakan sebagai source code, kemudian klik kanan untuk memunculkan context menu, dan pilih Open Git Bash.



Gambar 15 Context Menu Git Bash

Jika tidak ada opsi Open Git Bash, kalian perlu menjalankan regedit untuk mengutak atik registry agar memunculkan opsi nya. Kalian dapat menjalankan registry mudahnya menggunakan codingan berikut:

Windows Registry Editor Version 5.00

; Open files

; Default Git-Bash Location C:\Program Files\Git\git-bash.exe

[HKEY\_CLASSES\_ROOT\\*\shell\Open Git Bash]

@="Open Git Bash"

"Icon"="C:\\Program Files\\Git\\git-bash.exe"

[HKEY\_CLASSES\_ROOT\\*\shell\Open Git Bash\command]

@="\"C:\\Program Files\\Git\\git-bash.exe\" \"--cd=%1\""

; This will make it appear when you right click ON a folder

; The "Icon" line can be removed if you don't want the icon to appear

[HKEY\_CLASSES\_ROOT\Directory\shell\bash]

@="Open Git Bash"

"Icon"="C:\\Program Files\\Git\\git-bash.exe"

[HKEY\_CLASSES\_ROOT\Directory\shell\bash\command]

@="\"C:\\Program Files\\Git\\git-bash.exe\" \"--cd=%1\""

; This will make it appear when you right click INSIDE a folder

; The "Icon" line can be removed if you don't want the icon to appear

[HKEY\_CLASSES\_ROOT\Directory\Background\shell\bash]

@="Open Git Bash"

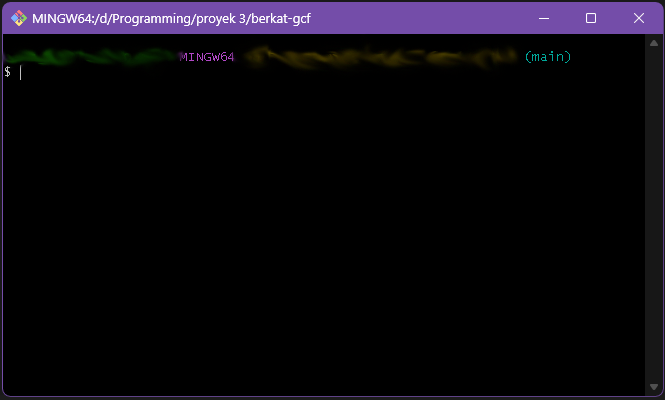
"Icon"="C:\\Program Files\\Git\\git-bash.exe"

[HKEY\_CLASSES\_ROOT\Directory\Background\shell\bash\command]

@="\"C:\\Program Files\\Git\\git-bash.exe\" \"--cd=%v.\""

Masukkan codingan kedalam notepad. Save codingan tersebut sebagai file .reg lalu jalankan. Cek Kembali apakah Open Git Bash muncul pada opsi context menu.

Tampilannya akan terlihat seperti berikut:



Gambar 16 Tampilan Utama Command Git Bash

## REKOMENDASI APLIKASI PEMBANGUN LAINNYA

Selain beberapa aplikasi yang dijelaskan tadi, adapun beberapa aplikasi yang dapat kalian instal untuk digunakan dalam pembuatan aplikasi website ini:

1. XAMPP, digunakan untuk menjalankan website secara lokal.
2. GoLand, aplikasi IDE seperti VSCode namun difokuskan untuk Bahasa pemrograma Golang.

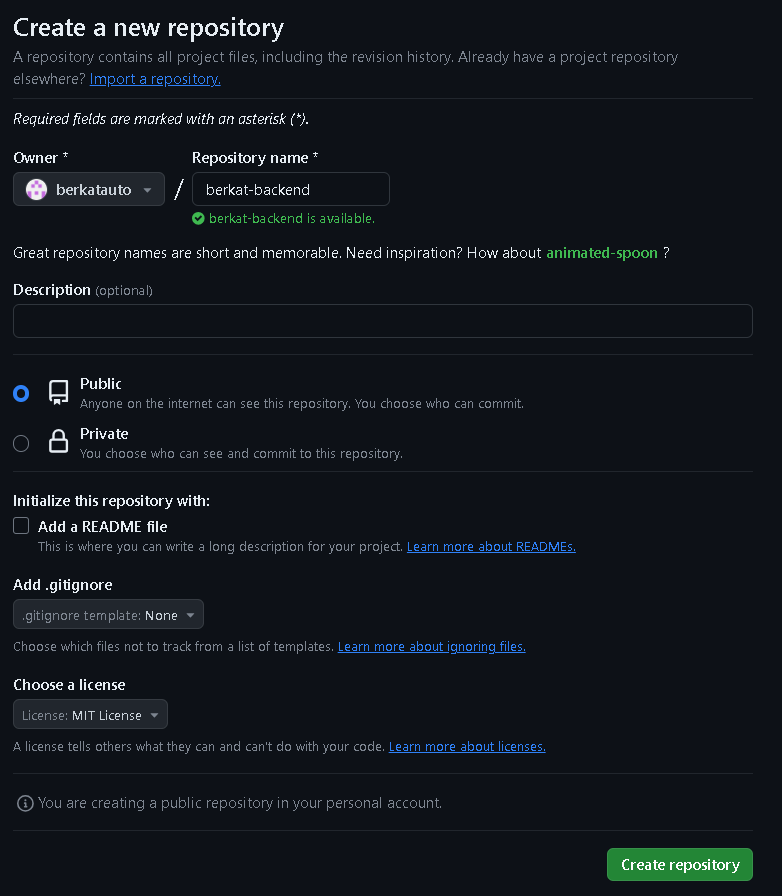
# BAB V PEMBUATAN APLIKASI WEBSITE

## PEMBUATAN REPOSITORY GITHUB

Github merupakan situs yang menyediakan source code, kalian dapat membuat repository nya disini. Dengan GitHub, kalian dapat menyimpan berbagai repository yang akan kalian gunakan dalam tutorial ini, kalian juga dapat menjalankan aplikasi web nya disini dengan menggunakan plugin pages yang disediakan github tanpa mengeluarkan banyak biaya.

Berikut adalah tahap pembuatan Repository GitHub.

1. Buat akun terlebih dahulu yang khusus digunakan sebagai akun aplikasi web. Kalian dapat membuat akun sendiri, atau menjadikannya sebagai akun Organization. Dalam Tutorial ini, kita hanya menggunakannya sebagai akun sendiri saja.
2. Pada Tampilan halaman depan, lihat pada navigation bar nya, dan klik pada tombol “+”, akan muncul dropdown opsi nya dan pilih “New Repository”.
3. Tampilan berikut adalah form untuk membuat Repository, silahkan beri nama Repository yang kalian buat dan isikan deskripsinya.



Gambar 17 Membuat Repository

Dalam hal ini, beberapa repository yang kita buat pada tutorial diperlukan License agar dapat di publikasi kan pada beberapa penyedia library pihak ketiga. Namun untuk kali ini kita tidak perlu membutuhkannya.

Untuk repository pertama, silahkan kalian beri namanya dengan (namaakun).github.io. Repository ini digunakan sebagai frontend yang akan kita buat dan menjadi domain utamanya.

## PEMBUATAN FRONTEND

Ada berbagai cara untuk membuat Front-end. Namun pada tutorial ini, kita akan menggunakan template sebagai Frontend. Pada tutorial ini, kita menggunakan Template sebagai Frontendnya. Kalian dapat menggunakan berbagai template yang ingin kalian gunakan, namun untuk tutorial ini, kita akan menggunakan template dari tailwind yang dapat kalian unduh pada link berikut:

Halaman Utama:

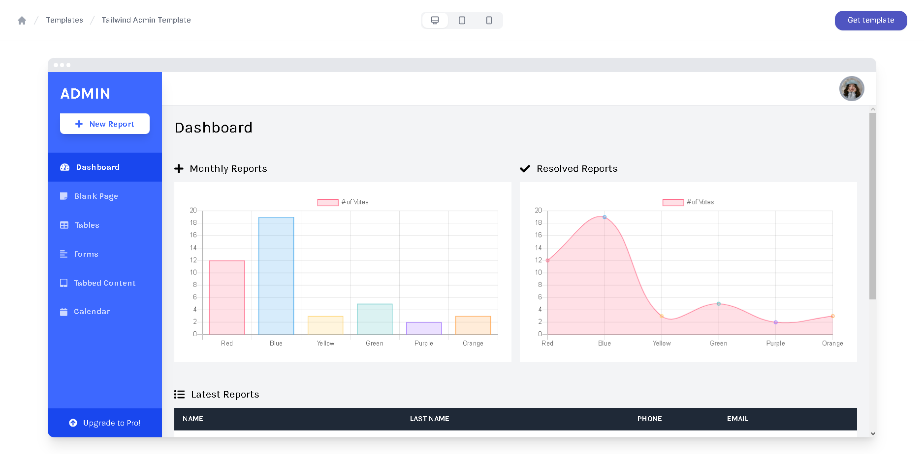
[uideck.com/templates/base-tailwind/](https://uideck.com/templates/base-tailwind/)



Gambar 18 Frontend Untuk Halaman Utama

Dashboard:

[www.tailwindawesome.com/resources/tailwind-admin-template](http://www.tailwindawesome.com/resources/tailwind-admin-template)



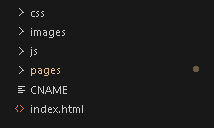
Gambar 19 Frontend Untuk Dashboard

Setelah itu, kita dapat langsung memasukkannya kedalam repository yang sudah kita buat. Ikuti tahap berikut:

### Memasukkan Template Kedalam Repository

Ikuti tahap berikut untuk memasukkan template kedalam repository.

1. Silahkan buat folder, beri nama sesuai keinginan untuk frontend.
2. Silahkan buatkan folder didalamnya seperti berikut:



Gambar 20 Foldering Frontend

1. Ekstrak kedua template frontend yang di unduh tadi, silahkan masukkan seperti pada gambar berikut.



Gambar 21 Isi Pages Untuk Frontend

Pastikan sesuaikan beberapa template berikut:

1. **Index.html** untuk halaman utama ditempatkan pada folder utama.
2. Untuk **signin, signup,** dan beberapa template halaman utama lainnya, silahkan masukkan kedalam folder pages, ganti Namanya untuk beberapa template menggunakan nama yang tertera pada gambar di atas.
3. Folder dsb adalah tempat dimana template dashboard ditempatkan, silahkan masukkan **index.html** dashboard kedalam folder tersebut. Ganti namanya menjadi **dashboard.html**.
4. Ada **blank.html** didalam template, silahkan masukkan juga untuk dijadikan bahan untuk membuat CRUD Form.
5. Dan untuk beberapa template dashboard lainnya, silahkan masukkan dan sesuaikan namanya.
6. Assets (CSS, Gambar dan beberapa JS) yang masuk sebagai pembangun template-template tersebut, masukkan kedalam folder-folder yang sudah dibuatkan tadi.

### Upload Repository

Tahap selanjutnya, kita akan memasukkan frontend yang telah kita buat kedalam repository. Berikut ini adalah tahap-tahapnya.

1. Masuk kedalam folder frontend utama yang telah kita buat.
2. Klik kanan, dan pilih Git Bash
3. Selanjutnya, silahkan masukkan dan jalankan command-command berikut secara berurutan:

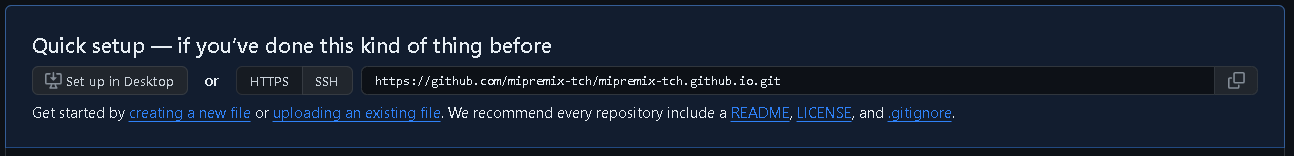
git init

git add .

git commit -m “Isikan pesan commit disini”

git branch -M main

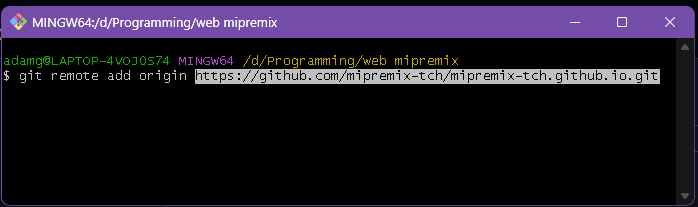
1. Kembali kedalam repository yang kita buat dalam GitHub. Perhatikan pada Quick Setupnya.



Gambar 22 Quick Setup

1. Silahkan pilih HTTPS dan copy link yang ada di sebelahnya.
2. Kembali kedalam Git Bash, silahkan masukkan command berikut dan paste link tadi.

git remote add origin “Link Repository tadi“



Gambar 23 Add Origin Pada Folder

Lalu Jalankan.

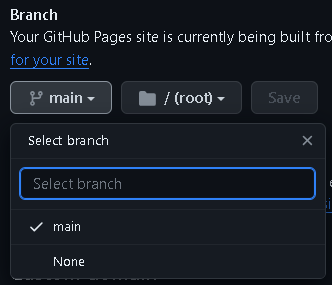
1. Terakhir, silahkan Push folder tersebut sehingga akan masuk kedalam repository GitHub.

git push -u origin main

### Jalankan Repository Frontend Menjadi Page

Untuk dapat menjalankan frontend yang tadi kita buat menjadi pages, silahkan ikuti tahap berikut:

1. Masuk ke repository frontend kalian di github, masuk kedalam menu Settings.
2. Pada pilihan Code and Automation, pilih menu Pages.
3. Akan muncul tampilan halaman GitHub Pages. Pada bagian branch, ubah branch None menjadi main.



Gambar 24 Branching Untuk Pages

1. Save dan tunggu beberapa saat.
2. Silahkan tes repository nya apakah sudah menjadi Pages. Silahkan masukkan link “(namaakun).github.io” kedalam browser, jalankan dan lihat hasilnya.

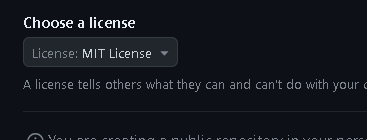
## PEMBUATAN BACKEND

Pada pembuatan Backend ini, kita akan menggunakan Golang sebagai Package nya. Silahkan ikuti beberapa tahap berikut untuk membuat Backend Aplikasi Website ini.

### Membuat Package

Berikut ini adalah tahap pembuatan Package Backend.

1. Buatlah satu Repository untuk Backend, gunakan nama yang dapat diingat. Pada pembuatan repository untuk package backend ini, kalian perlu menambahkan License, ini berguna agar kita dapat mempublikasi repository kita menjadi Library Golang. Pada pilihan License, Pilihlah MIT License.

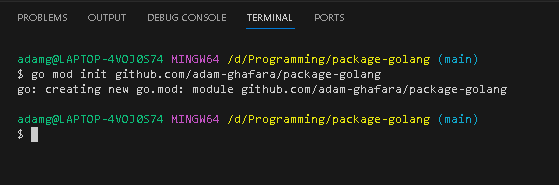


Gambar 25 Pilihan License

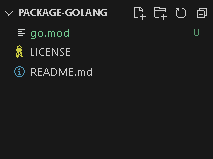
1. Disamping itu, kalian dapat mencentang “Add a Readme file” agar membuat markdown Readme secara otomatis. Kalian dapat menggunakan Readme tersebut sebagai dokumentasi package kalian. Setelahnya, salin link repository nya.
2. Setelah dibuat Repository, clone Repository tersebut kedalam computer lokal.
3. Silahkan buka folder repository menggunakan Visual Studio Code. Untuk membuka nya, kalian dapat membuka nya dengan Context menu. Jika tidak ada, kalian dapat menggunakan Git Bash agar dapat membuka folder nya di Visual Studio Code. Caranya dengan membuka folder repository tersebut dengan Git Bash, masukkan command code . kemudian jalankan, maka secara otomatis Visual Studio Code berjalan dengan folder yang tadi kita input.
4. Buka Terminal di Visual Code, caranya lihat pada menu bar terdapat Terminal, buka dan pilih New Terminal.
5. Jalankan command berikut untuk membuat file go.mod, ini digunakan sebagai module package pada backend yang kita buat.

go mod init “link github”

Contoh hasilnya seperti berikut:



Gambar 26 Hasil Terminal Pembuatan Package



Gambar 27 Hasil Pertama Pembuatan Package

### Membuat Endpoint API

Endpoint API dibuatkan untuk dapat digunakan pada API, Dimana sistem backend akan menjalankan proses sesuai setiap endpoint yang dibuat. Berikut ini adalah beberapa Endpoint yang dibuat dalam Backend.

1. Silahkan buat satu file .go untuk digunakan sebagai tempat endpoint.
2. Silahkan isikan file dengan codingan berikut:
3. Package dan Import

package berkatbepkg

import (

    "encoding/base64"

    "encoding/json"

    "fmt"

    "net/http"

    "os"

    "time"

    "github.com/whatsauth/watoken"

)

1. GCFHandler

func Handler(MONGOCONNSTRINGENV, dbname, collectionname string, r \*http.Request) string {

    mconn := SetConnection(MONGOCONNSTRINGENV, dbname)

    dataarticle := GetArticle(mconn, collectionname)

    return GCFReturnStruct(dataarticle)

}

1. LoginHandler

func LoginHandler(PASETOPRIVATEKEYENV, MONGOCONNSTRINGENV, dbname, collectionname string, r \*http.Request) string {

    var Response Credential

    Response.Status = false

    mconn := SetConnection(MONGOCONNSTRINGENV, dbname)

    var datauser User

    err := json.NewDecoder(r.Body).Decode(&datauser)

    if err != nil {

        Response.Message = "error parsing application/json: " + err.Error()

    } else {

        if IsPasswordValid(mconn, collectionname, datauser) {

            Response.Status = true

            tokenstring, err := watoken.Encode(datauser.Username, os.Getenv(PASETOPRIVATEKEYENV))

            if err != nil {

                Response.Message = "Gagal Encode Token : " + err.Error()

            } else {

                Response.Message = "Welcome!"

                Response.Token = tokenstring

            }

        } else {

            Response.Message = "Invalid Password"

        }

    }

    return GCFReturnStruct(Response)

}

1. CreateUserWToken

func GCFCreateUserWToken(PASETOPRIVATEKEYENV, MONGOCONNSTRINGENV, dbname, collectionname string, r \*http.Request) string {

    mconn := SetConnection(MONGOCONNSTRINGENV, dbname)

    var datauser User

    err := json.NewDecoder(r.Body).Decode(&datauser)

    if err != nil {

        return err.Error()

    }

    // Hash the password before storing it

    hashedPassword, hashErr := HashPassword(datauser.Password)

    if hashErr != nil {

        return hashErr.Error()

    }

    datauser.Password = hashedPassword

    CreateUserAndAddedToken(PASETOPRIVATEKEYENV, mconn, collectionname, datauser)

    fmt.Println("User Creation Succesfull. User Information : ", datauser)

    return GCFReturnStruct(datauser)

}

1. CreateUser

func GCFCreateUser(MONGOCONNSTRINGENV, dbname, collectionname string, r \*http.Request) string {

    mconn := SetConnection(MONGOCONNSTRINGENV, dbname)

    var datauser User

    err := json.NewDecoder(r.Body).Decode(&datauser)

    if err != nil {

        return err.Error()

    }

    // Hash the password before storing it

    hashedPassword, hashErr := HashPassword(datauser.Password)

    if hashErr != nil {

        return hashErr.Error()

    }

    datauser.Password = hashedPassword

    createErr := CreateNewUserRole(mconn, collectionname, datauser)

    fmt.Println(createErr)

    return GCFReturnStruct(datauser)

}

1. CreateHandler

func GCFCreateHandler(MONGOCONNSTRINGENV, dbname, collectionname string, r \*http.Request) string {

    mconn := SetConnection(MONGOCONNSTRINGENV, dbname)

    var datauser User

    err := json.NewDecoder(r.Body).Decode(&datauser)

    if err != nil {

        return err.Error()

    }

    // Hash the password before storing it

    hashedPassword, hashErr := HashPassword(datauser.Password)

    if hashErr != nil {

        return hashErr.Error()

    }

    datauser.Password = hashedPassword

    createErr := CreateNewUserRole(mconn, collectionname, datauser)

    fmt.Println(createErr)

    return GCFReturnStruct(datauser)

}

1. ReturnStruct

func ReturnStruct(DataStuct any) string {

    jsondata, \_ := json.Marshal(DataStuct)

    return string(jsondata)

}

1. PostArticle

func GCFPostArticle(MONGOCONNSTRINGENV, dbname, collectionname string, r \*http.Request) string {

    mconn := SetConnection(MONGOCONNSTRINGENV, dbname)

    var newarticle Article

    var fileConvert Content

    err := json.NewDecoder(r.Body).Decode(&newarticle)

    if err != nil {

        return err.Error()

    }

    response := GCFReturnStruct(newarticle)

    PostArticle(mconn, collectionname, newarticle)

    // Automatically If There is an Category

    // It will be added to the tags

    if newarticle.Category != "" {

        newarticle.Tags.Tag = newarticle.Category

    }

    // Add category at the first line of title

    newarticle.Title = newarticle.Category + " : " + newarticle.Title

    // Add the date

    newarticle.Date = time.Date(time.Now().Year(), time.Now().Month(), time.Now().Day(), time.Now().Hour(), time.Now().Minute(), time.Now().Second(), time.Now().Nanosecond(), time.UTC)

    // If Available, it will convert the image to base64

    fileConvert = newarticle.Content

    ConvertFileToBase64(fileConvert)

    UploadedVideoToBase64(fileConvert)

    return response

}

1. DeleteArticle

func DeleteArticle(MONGOCONNSTRINGENV, dbname, collectionname string, r \*http.Request) string {

    mconn := SetConnection(MONGOCONNSTRINGENV, dbname)

    var deleteArticle Article

    err := json.NewDecoder(r.Body).Decode(&deleteArticle)

    if err != nil {

        return err.Error()

    }

    response := GCFReturnStruct("Deleting Successful.")

    DeleteArticle(mconn, collectionname, deleteArticle)

    return response

}

1. UpdateArticle

func UpdateArticle(MONGOCONNSTRINGENV, dbname, collectionname string, r \*http.Request) string {

    mconn := SetConnection(MONGOCONNSTRINGENV, dbname)

    var updateArticle Article

    err := json.NewDecoder(r.Body).Decode(&updateArticle)

    if err != nil {

        return err.Error()

    }

    response := GCFReturnStruct(updateArticle)

    UpdateArticle(mconn, collectionname, updateArticle)

    return response

}

1. GetArticleByLastDate

func GetArticleByLastDate(MONGOCONNSTRINGENV, dbname, collectionname string) string {

    mconn := SetConnection(MONGOCONNSTRINGENV, collectionname)

    bylastdate := GetByLastDate(mconn, collectionname, Article{})

    return GCFReturnStruct(bylastdate)

}

1. ConvertFileToBase64

func ConvertFileToBase64(file Content) {

    // file.ImageHeader = base64.StdEncoding.EncodeToString([]byte(file.ImageHeader))

    file.Image = base64.StdEncoding.EncodeToString([]byte(file.Image))

}

1. UploadedVideoToBase64

func UploadedVideoToBase64(file Content) {

    file.VideoContent = base64.StdEncoding.EncodeToString([]byte(file.VideoContent))

}

1. Selanjutnya, buat file .go dengan nama type.go. Pada file ini, digunakan sebagai payload data yang akan keluar masuk lewat end point.
2. Isikan file dengan codingan struct berikut:

package berkatbepkg

import "time"

type articleID struct {

    randomArticleID int

}

type Article struct {

    Author   string    `json:"author" bson:"author"`

    Title    string    `json:"title" bson:"title"`

    Category string    `json:"category" bson:"category"`

    Tags     Tags      `json:"tags" bson:"tags"`

    Content  Content   `json:"content" bson:"content"`

    Date     time.Time `json:"date" bson:"date"`

}

type Tags struct {

    Tag string `json:"tag" bson:"tag"`

}

type Content struct {

    Paragraph    string `json:"paragraph" bson:"paragraph"`

    Image        string `json:"image" bson:"image"`

    VideoContent string `json:"video\_link,omiempty" bson:"video\_link,omiempty"` // If available, the video will automatically declared to be embedded.

}

type User struct {

    Fullname      string `json:"fullname" bson:"fullname"`

    Username      string `json:"username" bson:"username"`

    Password      string `json:"password" bson:"password"`

    JournalStatus bool   `json:"journal\_bool" bson:"journal\_bool"`

    Token         string `json:"token,omitempty" bson:"token,omitempty"`

    Role          string `json:"role,omitempty" bson:"role,omitempty"`

}

type Credential struct {

    Status  bool   `json:"status" bson:"status"`

    Token   string `json:"token,omitempty" bson:"token,omitempty"`

    Message string `json:"message,omitempty" bson:"message,omitempty"`

}

1. Buat satu file lagi dengan nama hash.go, Ini digunakan sebagai hashing untuk enkripsi password.
2. Isikan dengan codingan berikut:

package berkatbepkg

import "golang.org/x/crypto/bcrypt"

func HashPassword(password string) (string, error) {

    bytes, err := bcrypt.GenerateFromPassword([]byte(password), 14)

    return string(bytes), err

}

func CheckPasswordHash(password, hash string) bool {

    err := bcrypt.CompareHashAndPassword([]byte(hash), []byte(password))

    return err == nil

}

1. Setelah semua codingan lengkap, Kembali kedalam Terminal, dan masukkan command berikut:

go get -u all

go mod tidy

Command tersebut digunakan agar package golang yang kita buat akan mengunduh berbagai package yang di gunakan dalam Importing.

### Publish Repository ke Library Golang

Setelah kita membuat berbagai codingan untuk backend yang kita gunakan, kita perlu mempublish repository yang kita buat ke Library Golang. Ini agar Package Backend yang kita buat dapat berjalan menggunakan API.

Untuk mempublish Repository kita ke Library Golang, kita perlu melakukan tahap berikut:

1. Buatlah tag untuk Repository, caranya dapat menggunakan Terminal di Visual Studio Code. Dan masukkan command berikut:

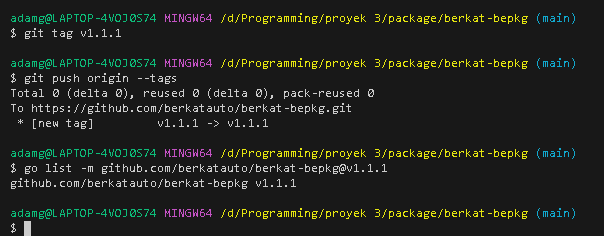
git tag                                                 # Cek version untuk package ini

git tag v1.0.0                                          # Set version untuk package ini

git push origin --tags                                  # Push Package dengan Version

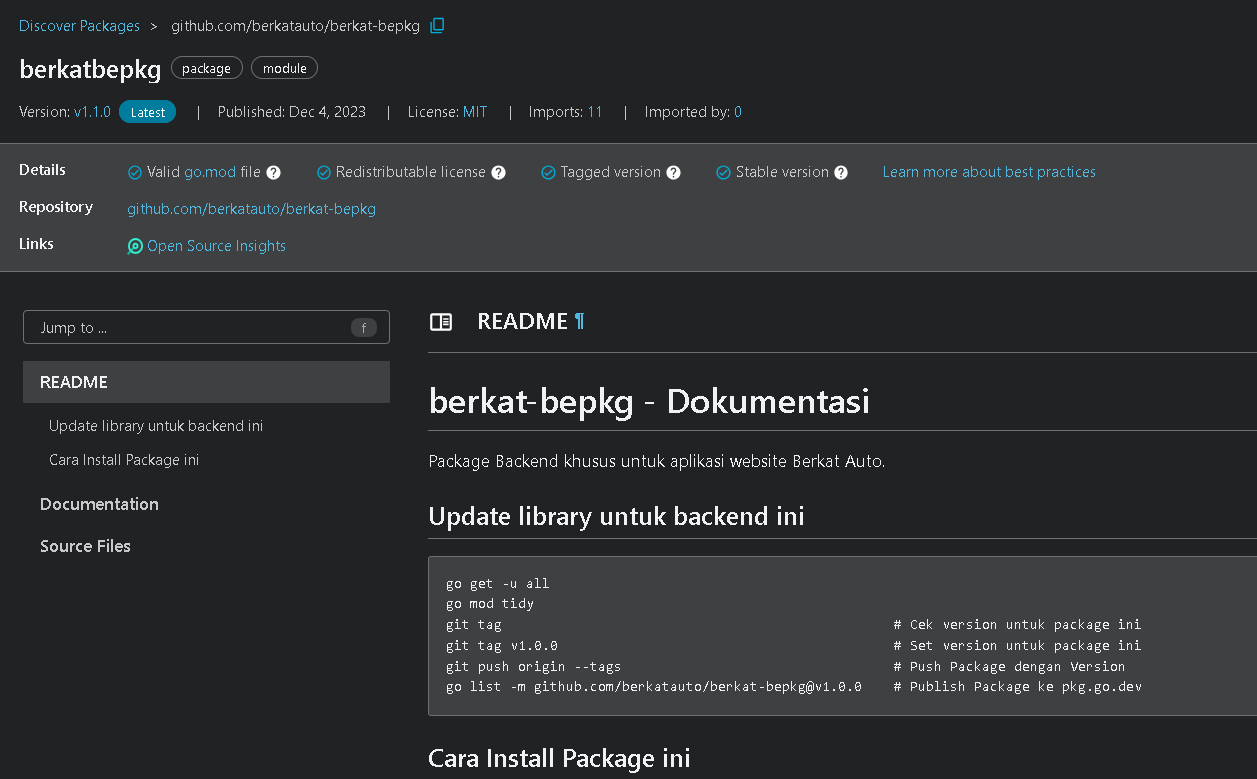
go list -m link\_repository\_github\_kalian@v1.0.0     # Publish Package ke pkg.go.dev

Contohnya seperti berikut:



Gambar 28 Hasil Dari Publish Repository Ke Library Golang

1. Setelahnya, kalian dapat mengeceknya di <pkg.go.dev>. Cari nama Package yang dipublish. Jika Repository telah dipublish, hasilnya terlihat seperti berikut:



Gambar 29 Tampilan Packagae Yang Kita Publish

Jika Package tidak ditemukan, besar kemungkinan Repository yang kalian publish masih dalam pengecekan. Setidaknya dibutuhkan 3 hari untuk dapat mempublish Package nya. Cek Kembali setelah 3 hari.

## PEMBUATAN API CLOUD

Setelah dibuatkan Backend, kita dapat melanjutkannya dengan membuat API Cloud. Untuk pembuatan API Cloud, kita menggunakan Google Cloud sebagai aplikasi yang mengatur sistem API nya.

### Membuat Akun Google Cloud

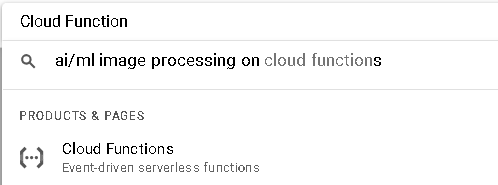
Google Cloud memiliki langganan gratis bagi pertama kali memakainya. Untuk membuat akun Google Cloud, pastikan memiliki kartu debit dan isikan Tabungan debit tersebut minimal RP.20.000. Setelah dilakukan. Silahkan lakukan tahap berikut:

1. Masuk ke halaman [cloud.google.com.](cloud.google.com/?hl=en) Sign in menggunakan akun google yang akan kalian gunakan dengan Cloud.
2. Pada halaman utama, silahkan klik pada “Get Started for Free” untuk membuat Google Cloud secara gratis.
3. Silahkan isikan beberapa data yang dibutuhkan dan lanjut dengan mengisikan billing nya.
4. Isikan data-data kartu debit yang diperlukan untuk memulai langganan dan buat akun Google Cloudnya.
5. Setelahnya, kalian diarahkan menuju halaman Console. Kalian juga mendapat 3 Basic Project yang dapat kalian pakai sebagai Cloud. Gunakan Project tersebut sehemat mungkin karena Project Basic tersebut memiliki Kuota yang terbatas.

### Membuat API

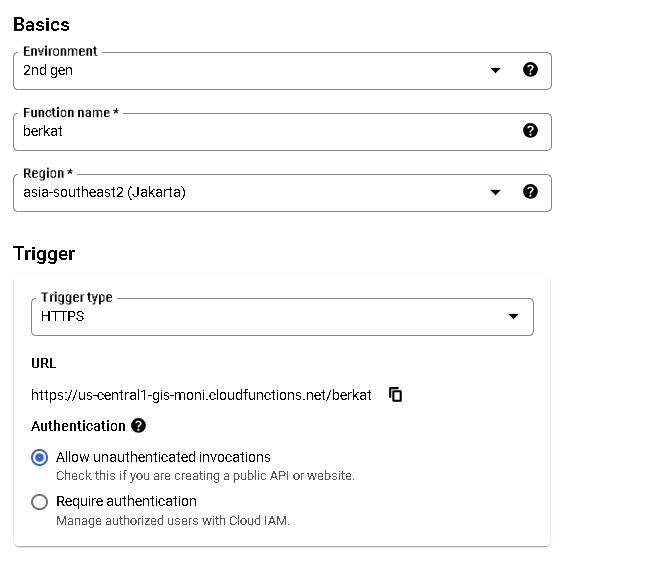
Untuk memulai membuat API, silahkan lakukan tahap berikut:

1. Pada console, terdapat search pada Menu bar, silahkan Cari Cloud Function.



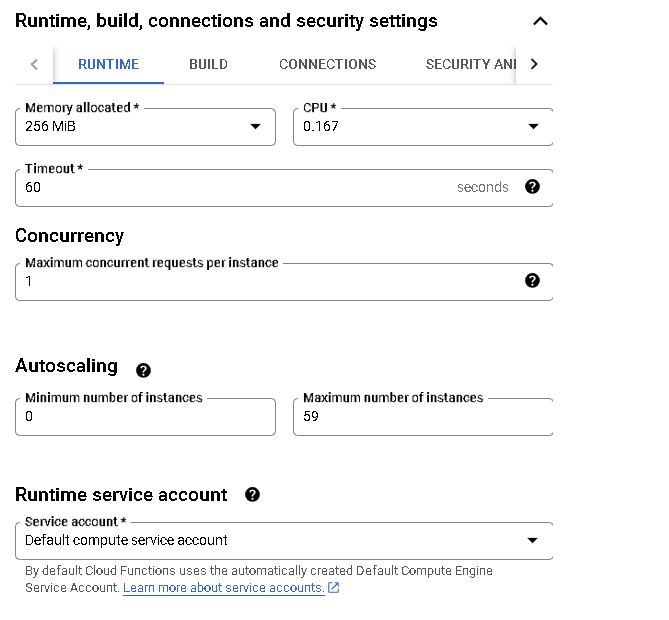
Gambar 30 Cloud Function Pada Pencarian

1. Klik pada Create Function untuk membuat satu Function API.
2. Kalian di arahkan pada Halaman Pembuatan Function. Silahkan buat pengaturan untuk pembuatan API seperti berikut:
3. Basics dan Trigger



Gambar 31 Pengaturan Basics dan Trigger

1. Runtime



Gambar 32 Pengaturan Runtime

1. Runtime Environment Variables



Gambar 33 Variables Untuk API

Variables MONGOSTRING menggunakan link yang disediakan pada Mongo Atlas yang digunakan dalam MongoDB Compass. Copy link tersebut dan masukkan kedalam Value.

Setelah semua di setel dengan benar, klik Next.

1. Kalian ditampilkan dengan halaman Code. Pada Runtime, silahkan pilih Golang 1.21, akan terlihat seperti berikut:
2. Isikan halaman Code dengan Codingan berikut:

package gcf

import ("fmt"

"net/http"

"github.com/GoogleCloudPlatform/functions-framework-go/functions"

**repositorybackend** **"link repository backend"**)

func init() {

functions.HTTP("HelloPostArticle", HelloPostArticle)

}

func HelloPostArticle(w http.ResponseWriter, r\*http.Request) {

w.Header().Set("Access-Control-Allow-Origin", "\*")

w.Header().Set("Access-Control-Allow-Methods", "**POST**")

w.Header().Set("Access-Control-Allow-Headers", "Content-Type, Authorization")

if r.Method == "OPTIONS" {

return

}

w.Header().Set("Access-Control-Allow-Origin", "\*")

response := **namapackage.namaEndPoint**("MONGOSTRING", "berkatauto", "articleSet", r)

fmt.Fprintf(w, response)

}

Perhatikan pada w.Header().Set. Pada Header keduanya, terdapat POST. Sesuaikan Header tersebut dengan End Point CRUD yang akan kalian gunakan. Jika End Point adalah untuk Delete, ganti POST tersebut menjadi DELETE. Jika End Point adalah untuk Mengambil Data, gati POST tersebut menjadi GET.

Jika bingung, kalian dapat mengkobinasi semua method End Point nya menjadi POST, GET, DELETE, PUT.

Perhatikan juga pada response, terdapat **namapackage.namaEndPoint**. Sesuaikan response tersebut dengan package yang kalian buat, dan End Point yang akan kalian gunakan. Misalnya untuk membuat End Point DELETE, maka gunakanlah End Point yang akan kalian gunakan untuk menghapus.

1. Buka pada codingan go.mod. dan Copy codingan berikut:

module **“link repository”**

go 1.21.3

require (

github.com/GoogleCloudPlatform/functions-framework-go v1.8.0

**github.com/namakaun/repository “versi repository”**

)

require (

aidanwoods.dev/go-paseto v1.5.0 // indirect

aidanwoods.dev/go-result v0.1.0 // indirect

github.com/aiteung/atdb v0.1.1 // indirect

github.com/cloudevents/sdk-go/v2 v2.14.0 // indirect

github.com/go-sql-driver/mysql v1.7.1 // indirect

github.com/golang-sql/civil v0.0.0-20220223132316-b832511892a9 // indirect

github.com/golang-sql/sqlexp v0.1.0 // indirect

github.com/golang/snappy v0.0.4 // indirect

github.com/google/uuid v1.3.0 // indirect

github.com/json-iterator/go v1.1.10 // indirect

github.com/klauspost/compress v1.17.2 // indirect

github.com/microsoft/go-mssqldb v1.6.0 // indirect

github.com/modern-go/concurrent v0.0.0-20180228061459-e0a39a4cb421 // indirect

github.com/modern-go/reflect2 v0.0.0-20180701023420-4b7aa43c6742 // indirect

github.com/montanaflynn/stats v0.7.1 // indirect

github.com/whatsauth/watoken v0.1.3 // indirect

github.com/xdg-go/pbkdf2 v1.0.0 // indirect

github.com/xdg-go/scram v1.1.2 // indirect

github.com/xdg-go/stringprep v1.0.4 // indirect

github.com/youmark/pkcs8 v0.0.0-20201027041543-1326539a0a0a // indirect

go.mongodb.org/mongo-driver v1.12.1 // indirect

go.uber.org/atomic v1.4.0 // indirect

go.uber.org/multierr v1.1.0 // indirect

go.uber.org/zap v1.10.0 // indirect

golang.org/x/crypto v0.14.0 // indirect

golang.org/x/sync v0.5.0 // indirect

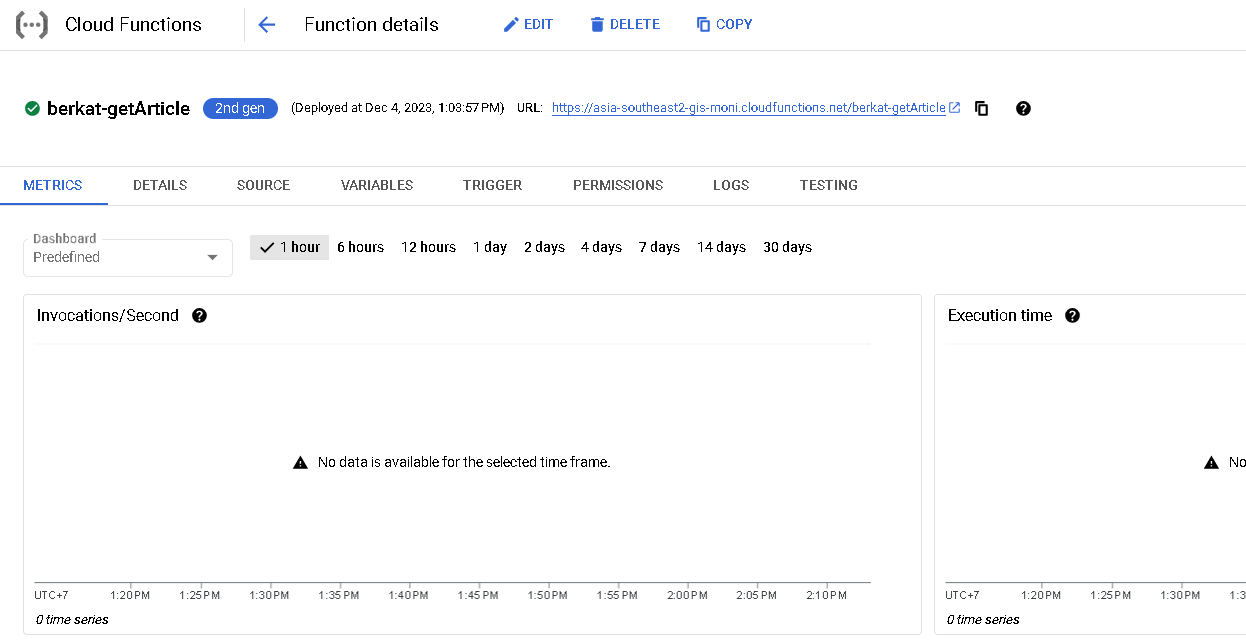
golang.org/x/sys v0.14.0 // indirect

golang.org/x/text v0.14.0 // indirect

)

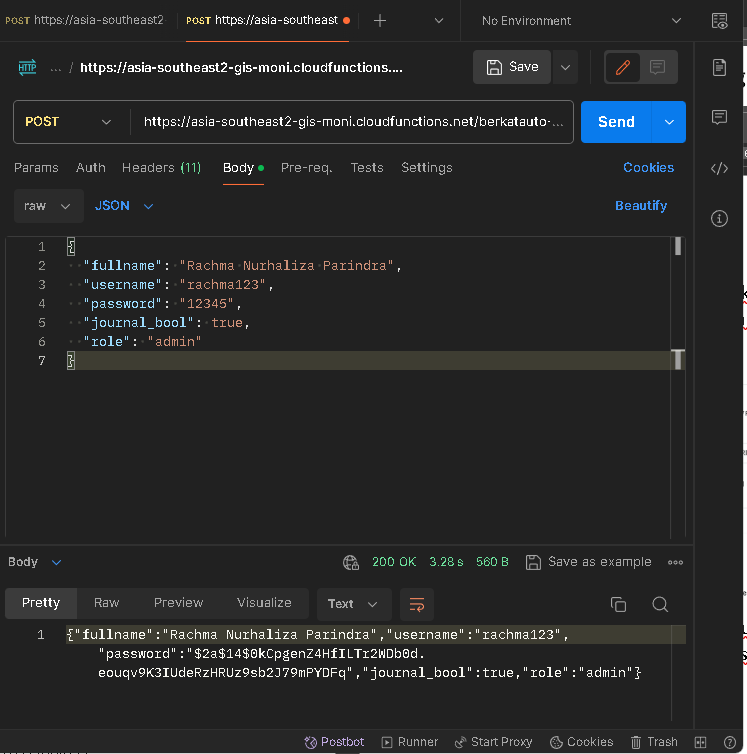
Pastikan mengubah “link repository” nya mengarah pada repository backend yang kalian buat sebelumnya yang akan digunakan untuk API.

1. Setelah semua sudah sesuai, klik pada “Deploy” dan tunggu beberapa saat untuk sistemnya menjalankan. Jika deploy berhasil, hasilnya terlihat seperti berikut:



Gambar 34

Sekarang kalian dapat menggunakan API tersebut, salin URL yang di sediakan API dan tes menggunakan Postman untuk mendapat hasil dari API nya.

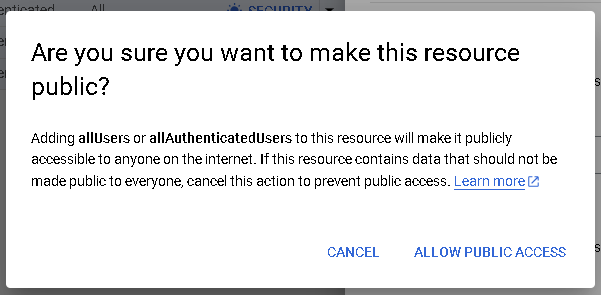


Gambar 35

### Setelan Tambahan Untuk Akses API

Adapun setelan tambahan untuk API yang dapat dilakukan untuk membuat Akses API secara terbuka.

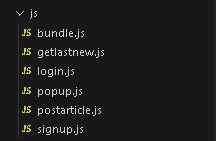
1. Cari Cloud Run. Jika sudah masuk halamannya, centang semua API yang dibuat.
2. Lihat pada “Permissions” di sebelah kanan halaman, Klik pada “Add Principal”.
3. Pada bagian Add Principals, tulis “allUsers”.
4. Pada Assign roles, Cari “Cloud Run”, dan pilih “Cloud Run Invoker”.
5. Jika sudah, Save. Dan jika muncul pop up berikut, pilihlah “Allow Public Access”.



Gambar 36 PopUp Public Access

API sudah menjadi Public.

Selanjutnya, kalian dapat menggunakan API ini kedalam frontend. Masukkan API tersebut kedalam Javascript untuk dapat dijalankan pada frontend.



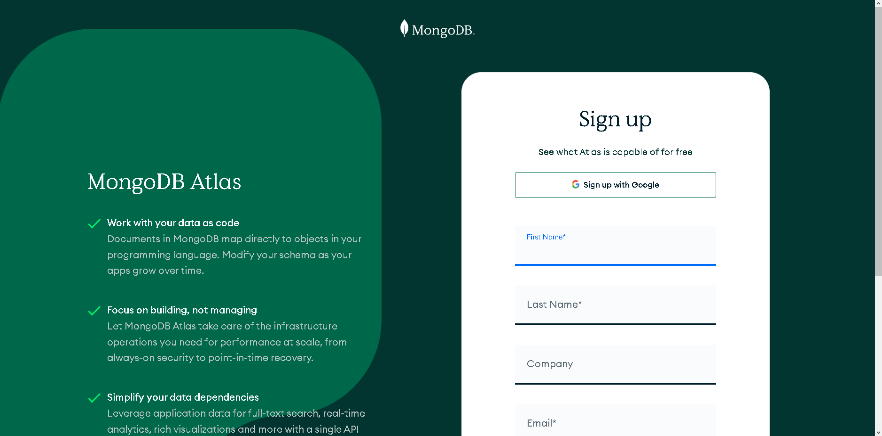
Gambar 37 Javascript Pada Frontend

## PEMBUATAN DATABASE

Pada tutorial selanjutnya, kita akan membuat Database menggunakan MongoDB. Berikut adalah tahap-tahapnya.

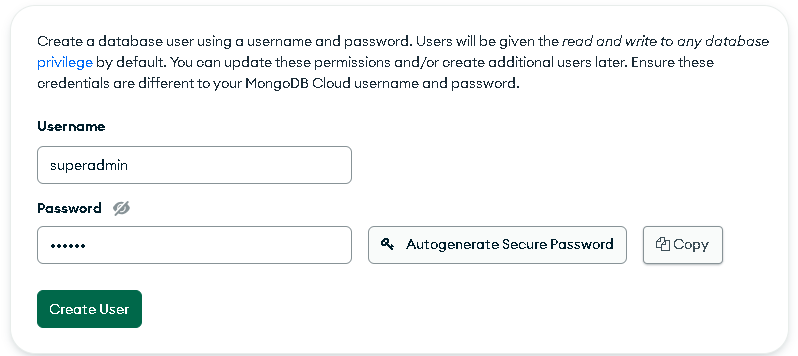
### Membuat Akun dan Membuat Database MongoDB

1. Silahkan masuk ke link berikut, silahkan isi data yang disediakan. Atau, kalian dapat menggunakan akun Google untuk digunakan membuat akun MongoDB.



Gambar 38 Halaman Daftar Akun MongoDB

1. Setelahnya, silahkan isi beberapa pertanyaan yang disediakan berikut.
2. Kalian akan langsung di arahkan ke halaman Deploy Database. Silahkan setel ke beberapa opsi pembuatan berikut:
   1. Pilih M0 untuk jenis database yang ingin dibuat.
   2. Silahkan pilih Providernya. Untuk rekomendasi, silahkan pilih Google Cloud sebagai providernya.
   3. Pada opsi Region, silahkan pilih “Jakarta (asia-southeast2).
   4. Beri nama untuk database kalian. Gunakanlah nama yang mudah dikenal dan di mengerti.
   5. Tag digunakan sebagai pengkategorian pada isi database. Silahkan kosongkan saja jika belum terlalu memahami fungsi nya.
3. Jika semua sudah siap, langsung Create databasenya.
4. Pembuatan Database selesai, dan sekarang kalian tinggal membuat User sebagai akun yang akan mengakses database kalian. Isikanlah Username dan Password nya lalu pilih Create User. User siap dipakai.

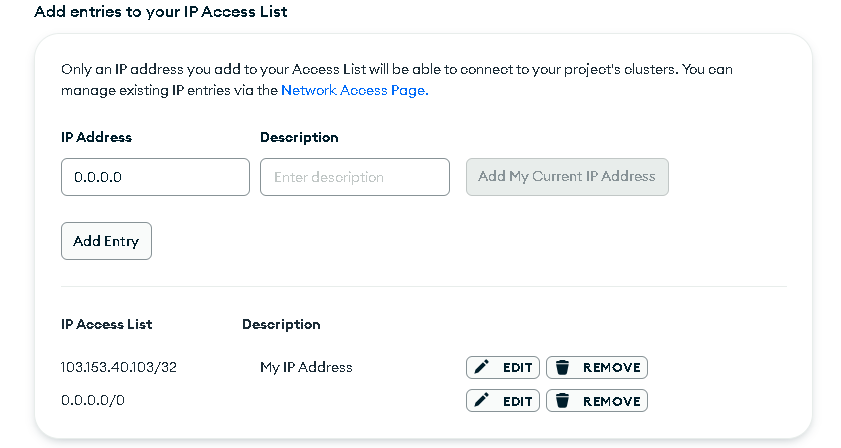


Gambar 39 Tampilan Pembuatan User Database

### Menyambungkan Database Dengan MongoDB Compass

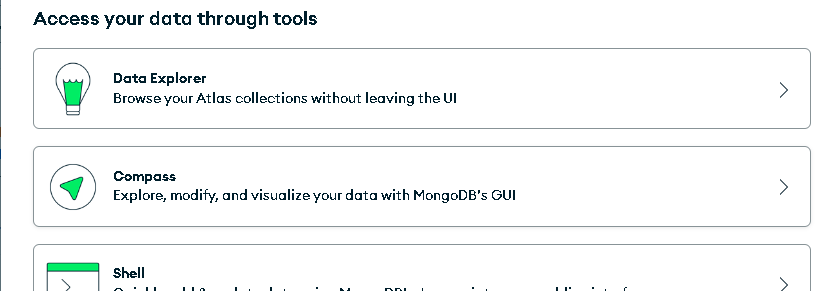
Ikuti tahap berikut untuk menyambungkan database dalam Atlas dengan MongoDB Compass.

1. Pada halaman Quickstart, lihat pada bagian “Add Entries to your IP Access List”
2. Pada IP address, silahkan isikan dengan 0.0.0.0, ini bertujuan agar semua IP yang dimiliki client dengan MongoDB compass dapat tersambung. Lalu pilih “Add my current IP Address”.



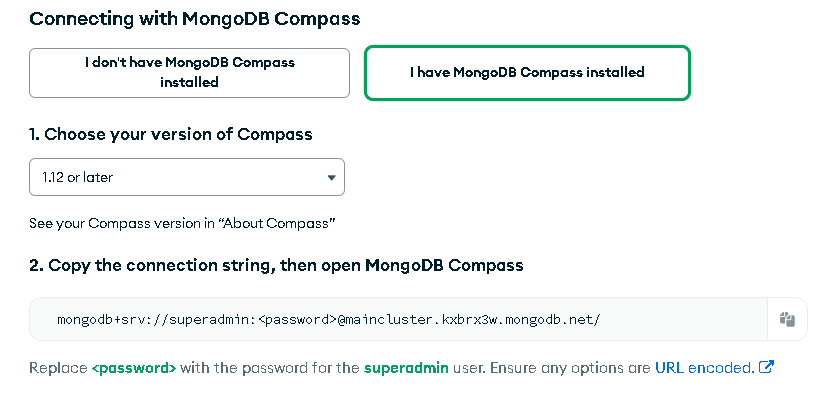
Gambar 40 Menambah Entries IP

1. Masuk ke menu Database, pada halaman tersebut terdapat database yang kita buat sebelumnya.
2. Silahkan pilih Connect.
3. Pada opsi berikut, silahkan pilih Compass.



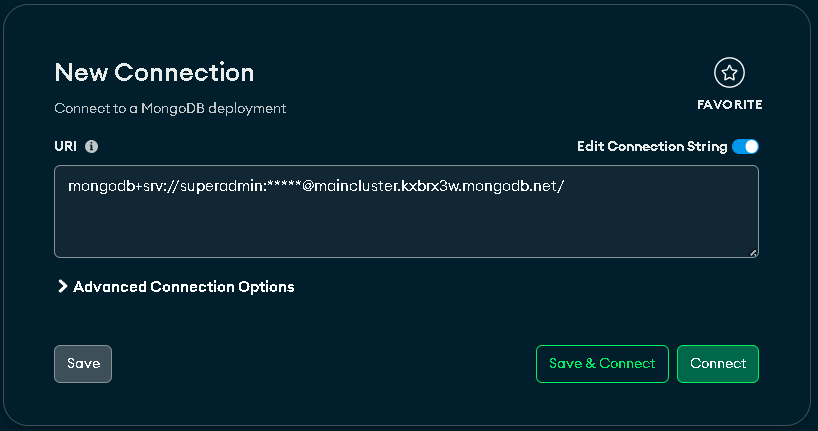
Gambar 41 Pilihan Untuk Menyambungkan MongoDB

1. Kemudian pilih “I have MongoDB Compass Installed”. Copy Alamat koneksi yang disediakan.



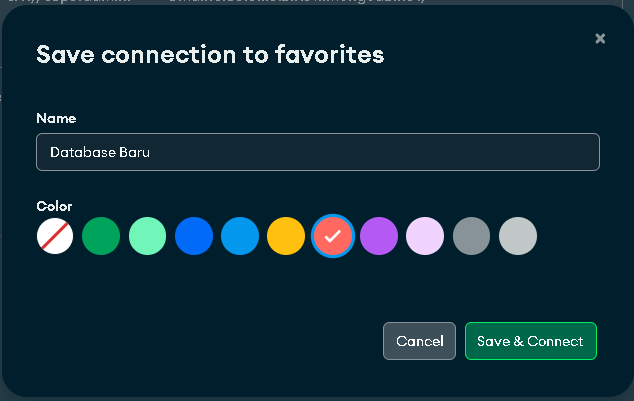
Gambar 42 Connecting with MongoDB Compass

1. Silahkan buka MongoDB Compass nya. Pada tampilan New Connection, paste Alamat yang di copy tadi, ganti <password> dengan password user kalian. Kemudian pilih Save & Connect.



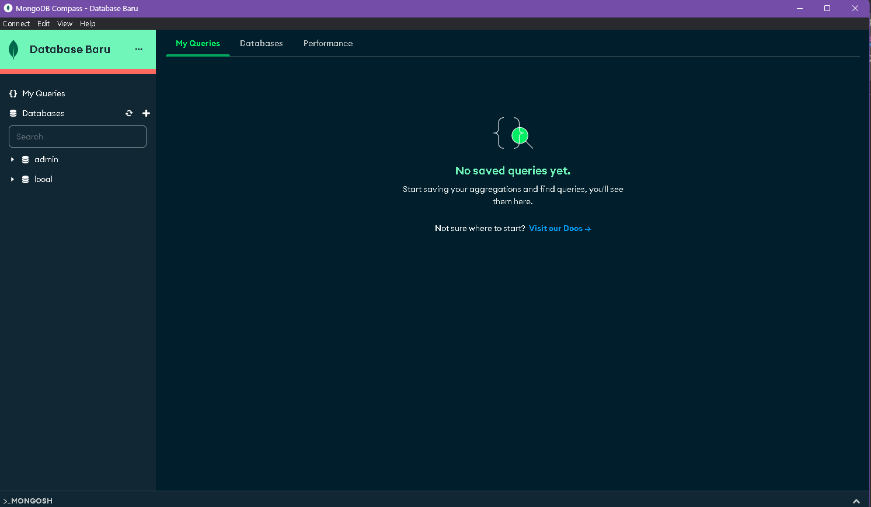
Gambar 43 Tampilan New Connection

1. Isikan nama dan warna untuk database yang dikoneksikan, lalu pilih Save & Connect.



Gambar 44 Tampilan Opsi Menambahkan Koneksi Database

1. Tunggu beberapa saat, dan jika tampilan terlihat seperti berikut, database berhasil tersambung. Sekarang, kalian dapat membuat databasenya menggunakan Compass.



Gambar 45 Tampilan Database

1. Untuk membuat database baru di MongoDB Compass, silahkan arahkan menuju symbol “+” di sebelah nama database kalian. Dan masukkan nama Database yang ingin kalian gunakan dalam aplikasi website nya.
2. Silahkan buat 2 Database untuk Article dan Users.

### Menyambungkan MongoDB dengan Backend

Berikut ini adalah cara menyambungkan MongoDB dengan Backend yang sudah dibuat sebelumnya.

1. Pada Repository untuk Backend, buatlah satu file go dengan nama mongo.go.
2. Isikan dengan beberapa function berikut:
3. Package dan import

package berkatbepkg

import (

    "os"

    "github.com/aiteung/atdb"

    "github.com/whatsauth/watoken"

    "go.mongodb.org/mongo-driver/bson"

    "go.mongodb.org/mongo-driver/mongo"

)

1. SetConnection

func SetConnection(MONGOCONNSTRINGENV, dbname string) \*mongo.Database {

    var DBmongoinfo = atdb.DBInfo{

        DBString: os.Getenv(MONGOCONNSTRINGENV),

        DBName:   dbname,

    }

    return atdb.MongoConnect(DBmongoinfo)

}

1. GetNameAndPassword

func GetNameAndPassowrd(mongoconn \*mongo.Database, collection string) []User {

    user := atdb.GetAllDoc[[]User](mongoconn, collection)

    return user

}

1. SearchByAuthor

func SearchByAuthor(mongoconn \*mongo.Database, collection string, searchBy Article) Article {

    filter := bson.M{"author": searchBy.Author}

    return atdb.GetOneDoc[Article](mongoconn, collection, filter)

}

1. SearchByCategory

func SearchByCategory(mongoconn \*mongo.Database, collection string, searchBy Article) Article {

    filter := bson.M{"category": searchBy.Category}

    return atdb.GetOneDoc[Article](mongoconn, collection, filter)

}

1. SearchByTitle

func SearchByTitle(mongoconn \*mongo.Database, collection string, searchBy Article) Article {

    filter := bson.M{"title": searchBy.Title}

    return atdb.GetOneDoc[Article](mongoconn, collection, filter)

}

1. SearchByTags

func SearchByTags(mongoconn \*mongo.Database,collection string, searchBy Article) Article {

    filter := bson.M{"tags": searchBy.Tags}

    return atdb.GetOneDoc[Article](mongoconn, collection, filter)

}

1. GetAllUser

func GetAllUser(mongoconn \*mongo.Database, collection string) []User {

    user := atdb.GetAllDoc[[]User](mongoconn, collection)

    return user

}

1. IsPasswordValid

func IsPasswordValid(mongoconn \*mongo.Database, collection string, userdata User) bool {

    filter := bson.M{"username": userdata.Username}

    res := atdb.GetOneDoc[User](mongoconn, collection, filter)

    return CheckPasswordHash(userdata.Password, res.Password)

}

1. PostArticle

func PostArticle(mongoconn \*mongo.Database, collection string, articleData Article) interface{} {

    return atdb.InsertOneDoc(mongoconn, collection, articleData)

}

1. GetArticle

func GetArticle(mongoconn \*mongo.Database, collection string) []Article {

    tampilartikel := atdb.GetAllDoc[[]Article](mongoconn, collection)

    return tampilartikel

}

1. LoadArticle

func LoadArticle(mongoconn \*mongo.Database, collection string, articleData Article) Article {

    // Load by title if article selected

    if articleData.Title != "" {

        filter := bson.M{"title": articleData.Title}

        return atdb.GetOneDoc[Article](mongoconn,collection, filter)

    }

    return atdb.GetOneDoc[Article](mongoconn, collection, nil)

}

1. GetOneArticle

func GetOneArticle(mongoconn \*mongo.Database, collection string, articleData Article) Article {

    filter := bson.M{"title": articleData.Title}

    return atdb.GetOneDoc[Article](mongoconn, collection, filter)

}

1. GetByLastDate

func GetByLastDate(mongoconn \*mongo.Database, collection string, articleData Article) Article {

    filter := bson.M{"date": articleData.Date}

    return atdb.GetOneDoc[Article](mongoconn, collection, filter) }

1. GetByAuthor

func GetByAuthor(mongoconn \*mongo.Database, collection string, articleData Article) Article {

    filter := bson.M{"author": articleData.Author}

    return atdb.GetOneDoc[Article](mongoconn, collection, filter)

}

1. UpdateArticle

func UpdateArticle(mongoconn \*mongo.Database, collection string, articleData Article) interface{} {

    filter := bson.M{"title": articleData.Title}

    return atdb.ReplaceOneDoc(mongoconn, collection, filter, articleData)

}

1. DeleteArticle

func DeleteArticle(mongoconn \*mongo.Database, collection string, articleData Article) interface{} {

    filter := bson.M{"title": articleData.Title}

    return atdb.DeleteOneDoc(mongoconn, collection, filter)

}

1. CreateNewUserRole

func CreateNewUserRole(mongoconn \*mongo.Database, collection string, userdata User) interface{} {

    // Hash the password before storing it

    hashedPassword, err := HashPassword(userdata.Password)

    if err != nil {

        return err

    }

    userdata.Password = hashedPassword

    // Insert the user data into the database

    return atdb.InsertOneDoc(mongoconn, collection, userdata) }

1. CreateUserAndAddedToken

func CreateUserAndAddedToken(PASETOPRIVATEKEYENV string, mongoconn \*mongo.Database, collection string, userdata User) interface{} {

    // Hash the password before storing it

    hashedPassword, err := HashPassword(userdata.Password)

    if err != nil {

        return err

    }

    userdata.Password = hashedPassword

    // Insert the user data into the database

    atdb.InsertOneDoc(mongoconn, collection, userdata)

    // Generate Token

    // Create a token for the user

    tokenstring, err := watoken.Encode(userdata.Username, os.Getenv(PASETOPRIVATEKEYENV))

    if err != nil {

        return err

    }

    userdata.Token = tokenstring

    // Update the user data in the database

    return atdb.ReplaceOneDoc(mongoconn, collection, bson.M{"username": userdata.Username}, userdata) }

1. Save, jalankan tidy, dan buat tag versi baru untuk backend dan publish backend ke Library Golang.

## KELENGKAPAN SISTEM

Setelah dilakukan tahap tutorial di atas, setiap repository seharusnya terdapat beberapa item berikut:

### Frontend

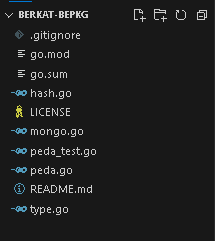
Berikut ini adalah kelengkapan pada Frontend:



Gambar 46 Kelengkapan Frontend

### Backend

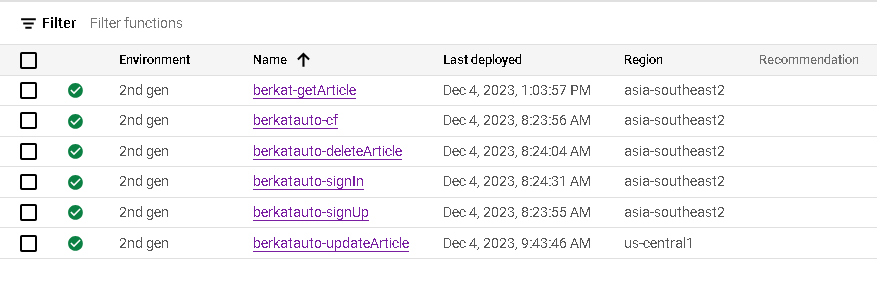
Berikut adalah kelengkapan untuk Backend:



Gambar 47 Kelengkapan Backend

### API

Berikut adalah Kelengkapan API yang digunakan pada Tutorial:



Gambar 48 Kelengkapan API Yang Di Gunakan Untuk Frontend

### MongoDB

Berikut adalah kelengkapan Database MongoDB pada tutorial ini:



Gambar 49

Untuk lebih lanjutnya, kalian dapat melihat pada akun github kami sekaligus menjadi Source Code untuk Tutorial ini.

[github.com/berkatauto](https://github.com/berkatauto)

# DAFTAR PUSTAKA

# GLOSARIUM

A

**API:**

B

**Backend:**

C

D

**Database**

E

**API** :

F

**Frontend:**

G

**Golang:**

H

**API** :

I

**API** :

J

**JSON:**

K

**API** :

L

**Library:**

M

N

**NonSQL :**

O

**Otomotif:**

**Open Source:**

P

**API** :

Q

**API** :

R

**API** :

S

**API** :

T

**API** :

U

**URL:**

V

**API** :

W

**Website:**

X

**API** :

Y

**API** :

Z

**API** :

# INDEKS

# TENTANG PENULIS