

MagicAppBuilder

PANDUAN PENGGUNA



18 JUNI 2025

Kamshory

Jakarta

# KATA PENGANTAR

Puji dan syukur penulis panjatkan ke hadirat Allah SWT, karena atas rahmat dan karunia-Nya, penulis dapat menyelesaikan dokumen ini dengan baik. Dokumen ini disusun sebagai salah satu bentuk penyampaian informasi yang diharapkan dapat memberikan manfaat bagi pembaca.

Dalam dunia teknologi yang terus berkembang, konsep pemrograman tanpa kode atau *low-code programming* semakin populer. *Low-code programming* memungkinkan pengembang untuk membangun aplikasi dengan sangat sedikit menulis kode program secara manual. Pendekatan ini dirancang untuk mempercepat proses pengembangan dengan cara mendefinisikan elemen-elemen inti aplikasi seperti database, formulir, dan berbagai operasi yang akan dijalankan oleh modul tanpa perlu menulis skrip pemrograman yang kompleks. Oleh karena itu, dokumen ini disusun untuk menguraikan lebih lanjut konsep serta implementasi dari *low-code programming*.

Dalam penyusunan dokumen ini, penulis berusaha menyajikan informasi dengan sejelas dan sebaik mungkin agar mudah dipahami serta dapat digunakan sesuai dengan kebutuhan. Penulis juga menyadari bahwa dalam proses penyusunan ini tidak terlepas dari bantuan, dukungan, serta bimbingan dari berbagai pihak. Oleh karena itu, penulis ingin mengucapkan terima kasih kepada semua pihak yang telah memberikan kontribusi baik secara langsung maupun tidak langsung.

Semoga dokumen ini dapat memberikan manfaat dan wawasan yang berguna bagi para pembaca. Penulis juga menyadari bahwa dokumen ini masih memiliki kekurangan, oleh karena itu, kritik dan saran yang membangun sangat diharapkan demi perbaikan di masa mendatang.

Jakarta, Mei 2025

Penulis

# DAFTAR ISI

[KATA PENGANTAR 1](#_Toc201914299)

[DAFTAR ISI 2](#_Toc201914300)

[BAB 1 – PENDAHULUAN 4](#_Toc201914301)

[1.1 Definisi *Low-code Programming* 4](#_Toc201914302)

[1.2 Keunggulan *Low-code Programming* 4](#_Toc201914303)

[1.3 Keterbatasan *Low-code Programming* 5](#_Toc201914304)

[BAB 2 – IMPLEMENTASI 6](#_Toc201914305)

[2.1 Implementasi Pada Aplikasi Berbasis Web 6](#_Toc201914306)

[2.2 MagicAppBuilder - Platform *Low-code Programming* 6](#_Toc201914307)

[BAB 3 – MAGICAPPBUILDER 8](#_Toc201914308)

[3.1 Fitur MagicAppBuilder 8](#_Toc201914309)

[3.2 Subsitem MagicAppBuilder 8](#_Toc201914310)

[3.2.1 Database Explorer 9](#_Toc201914311)

[3.2.2 Entity Editor 9](#_Toc201914312)

[3.2.3 Administration 10](#_Toc201914313)

[3.3 File Manager 10](#_Toc201914314)

[3.3.1 Fitur Utama 11](#_Toc201914315)

[3.3.2 Menu Konteks Direktori 11](#_Toc201914316)

[3.3.3 Menu Konteks Direktori Root 12](#_Toc201914317)

[3.3.4 Menu Konteks File 13](#_Toc201914318)

[3.3.5 Cara Menggunakan File Manager 13](#_Toc201914319)

[BAB 4 – PANDUAN PENGGUNAAN 15](#_Toc201914320)

[4.1 Versi MagicAppBuilder 15](#_Toc201914321)

[4.2 Antarmuka Pengguna 15](#_Toc201914322)

[4.3 Langkah-Langkah Penggunaan 19](#_Toc201914323)

[4.1.1 Langkah 1: Instalasi Server 19](#_Toc201914324)

[4.1.2 Langkah 2: Mengunduh dan Memasang MagicAppBuilder di Server 22](#_Toc201914325)

[4.1.3 Langkah 3: Membuat Workspace 32](#_Toc201914326)

[4.1.4 Langkah 4: Membuat Aplikasi 34](#_Toc201914327)

[4.1.5 Langkah 5: Konfigurasi Aplikasi 35](#_Toc201914328)

[4.1.6 Langkah 6: Menyiapkan Struktur Database 38](#_Toc201914329)

[4.1.7 Langkah 7: Membuat Entitas dengan Entity Editor 39](#_Toc201914330)

[4.1.8 Langkah 8: Membuat Menu Aplikasi 41](#_Toc201914331)

[4.1.9 Langkah 9: Membuat Modul 41](#_Toc201914332)

[4.1.10 Langkah 10: Penjelasan Tab Generate Module 42](#_Toc201914333)

[4.1.11 Langkah 11: Mengirimkan Formulir untuk Membuat Modul dan Entitas 56](#_Toc201914334)

[4.1.12 Langkah 12: Memperbarui Struktur Database 56](#_Toc201914335)

[4.1.13 Langkah 13: Membuat Lokalisasi 56](#_Toc201914336)

[4.1.14 Langkah 14: Opsi Aplikasi 57](#_Toc201914337)

[4.1.15 Langkah 15: Menambahkan Favicon ke Aplikasi 59](#_Toc201914338)

[4.1.16 Langkah 16: Melakukan Kustomisasi Aplikasi 60](#_Toc201914339)

[4.4 Langkah 17: Deploymen Aplikasi 60](#_Toc201914340)

[4.5 Siklus Hidup Pengembangan Perangkat Lunak 61](#_Toc201914341)

[4.1.1 Pengembangan Fitur Aplikasi 61](#_Toc201914342)

[4.1.2 Optimasi Database 62](#_Toc201914343)

[4.1.3 Scaling dan Sharding Database 62](#_Toc201914344)

[4.1.4 Perbaikan Masalah (Debugging) 63](#_Toc201914345)

[4.1.5 Pembaruan Sistem (Maintenance) 63](#_Toc201914346)

[BAB 5 – KESIMPULAN 64](#_Toc201914347)

# BAB 1 – PENDAHULUAN

## Definisi *Low-code Programming*

*Low-code programming* adalah metode pengembangan perangkat lunak yang memungkinkan pengembang untuk membangun dan mengelola aplikasi menggunakan antarmuka visual, tanpa harus menulis banyak kode secara manual. Dalam pendekatan ini, seorang programer hanya perlu mendefinisikan komponen utama aplikasi, seperti:

1. Database: Struktur dan relasi antara data yang akan digunakan dalam aplikasi.
2. Form: Antarmuka yang digunakan pengguna untuk memasukkan atau melihat data.
3. Operasi Modul: Fungsi-fungsi utama yang dapat dilakukan dalam sistem, seperti:

Dengan *low-code programming*, pengembang dapat membangun aplikasi dengan lebih cepat karena sebagian besar logika bisnis sudah tersedia dalam bentuk modul yang dapat dikonfigurasi tanpa perlu menulis kode secara manual.

## Keunggulan *Low-code Programming*

*Low-code programming* menawarkan solusi yang cepat, efisien, dan mudah digunakan dalam pengembangan aplikasi. Dengan pendekatan ini, pengembang dapat fokus pada definisi database, form, serta berbagai operasi modul tanpa perlu menulis banyak kode.

Dengan semakin berkembangnya teknologi, *low-code programming* berpotensi menjadi standar baru dalam pengembangan perangkat lunak, membuka peluang bagi lebih banyak orang untuk menciptakan aplikasi tanpa harus menjadi ahli pemrograman.

Beberapa keunggulan dari *low-code programming* antara lain sebagai berikut:

* 1. Meningkatkan Kecepatan Pengembangan

Pendekatan ini mempercepat siklus pengembangan perangkat lunak karena pengembang tidak perlu menulis kode dari nol. Sebagian besar fungsi aplikasi dapat dibuat dengan mengkonfigurasi modul yang sudah tersedia.

* 1. Mengurangi Kesalahan Kode

Karena pemrograman dilakukan secara visual dan minim kode, risiko terjadinya bug atau kesalahan sintaksis dalam kode berkurang secara signifikan.

* 1. Memudahkan Pemeliharaan dan Skalabilitas

*Low-code programming* memungkinkan aplikasi lebih mudah untuk diperbarui dan diperbaiki, karena komponen-komponen utama sudah terstruktur dengan baik dan dapat dikonfigurasi tanpa perlu menulis ulang banyak kode.

* 1. Aksesibilitas bagi Non-Programer

Dengan adanya pendekatan ini, orang-orang yang bukan dari latar belakang pemrograman juga dapat membangun aplikasi. Hal ini membuka peluang bagi bisnis untuk mengembangkan solusi mereka sendiri tanpa harus bergantung sepenuhnya pada tim pengembang.

## Keterbatasan *Low-code Programming*

Meskipun memiliki banyak keunggulan, *low-code programming* juga memiliki beberapa keterbatasan. Keterbatasan ini disebabkan karena bisnis memiliki proses yang spesifik dibandingkan dengan bisnis lainnya. Selain itu, pemrosesan data yang kompleks mungkin dibutuhkan dan tidak dapat diprediksi sebelumnya.

Oleh sebab itu, *low-code programming* tetap memberikan keleluasaan bagi programer untuk menambahkan kode program secara manual. Penambahan kode program dapat mengatasi semua kebutuhan bisnis. Dengan demikian, tidak ada yang tidak dapat dilakukan oleh aplikasi.

# BAB 2 – IMPLEMENTASI

## Implementasi Pada Aplikasi Berbasis Web

Aplikasi berbasis web merupakan implementasi yang paling mudah untuk *low-code programming* karena aplikasi berbasis web pada umumnya adalah aplikasi scripting *non compilation* yang berarti kode program langsung dapat dijalankan oleh system tanpa melalui proses kompilasi terlebih dahulu. Contoh dari salah satu scripting adalah PHP-Hypertext Preprocessor. PHP adalah salah satu Bahasa permrograman yang popular yang banyak digunakan dan masih terus dikembangkan. Versi terbaru dari PHP adalah 8.3 yang dirilis di akhir September tahun 2024.

Selain kode program, desain struktur database merupakan salah satu pekerjaan yang sangat penting. Karena program akan dibuat secara otomatis oleh aplikasi, maka struktur database harus konsisten dan memiliki pola tertentu yang merupakan persyaratan teknis dari aplikasi pembuat kode program. Beberapa kolom tabel harus mengikuti penamaan tententu dan bersifat kaku. Struktur database yang dibuat sembarangan dan tidak mengikuti pola yang diinginkan akan menyebabkan aplikasi gagal dalam mendefinisikan fungsi dari masing-masing kolom dan akan menimbulkan masalah dalam relasi antar entitas.

## MagicAppBuilder - Platform *Low-code Programming*

Tulisan ini akan membatasi pembahasan aplikasi pada MagicAppBuilder, yaitu aplikasi yang didesain untuk membuat kode program secara otomatis sesuai dengan database dan konfigurasi yang dibuat oleh pengguna.

MagicAppBuilder dibuat dengan bahasa PHP. MagicAppBuilder mulai dibangun tahun 2024 namun cikal bakalnya sudah dibuat sejak tahun 2013. Program yang dibuat oleh MagicAppBuilder juga menggunakan bahasa PHP dengan dukungan dari framework MagicObject dan MagicApp.

Sebelum mulai program mulai dibuat, database harus dipersiapkan terlebih dahulu. Pembuatan program dilakukan dari MagicAppBuilder. Setelah kerangka terbentuk, konfigurasi program harus ditentukan terlebih dahulu termasuk memilih *Database Management System (DBMS)* termasuk mengatur hubungan dengan database tersebut. MagicAppBuilder mendukung DBMS sebagai berikut:

* + 1. MySQL
    2. MariaDB
    3. PostgreSQL
    4. SQLite

Setelah hubungan terbentuk, pengguna memilih sebuah tabel dari database yang akan dibuatkan modulnya. MagicAppBuilder akan menampilkan kolom yang ada pada tabel tersebut. Pengguna membuat konfigurasi modul sesuai dengan proses yang diharapkan ada pada modul tersebut. Selanjutnya, MagicAppBuilder akan membuat modul dan beberapa entitas yang terkait. Entitas adalah model yang digunakan oleh program untuk berhubungan dengan database. Entitas akan memiliki kolom sesuai dengan tabel yang digunakan. Entitas mungkin memiliki objek yang merupakan instan dari entitas lain.

# BAB 3 – MAGICAPPBUILDER

## Fitur MagicAppBuilder

MagicAppBuilder memiliki beberapa fitur yang mendukung pembuatan program. Fitur-fitur pada MagicAppBuilder adalah sebagai berikut:

1. Membuat workspase
2. Membuat aplikasi baru
3. Mengubah pengaturan aplikasi
4. Membuat modul aplikasi
5. Membuat entitas sesuai dengan tabel yang terkait
6. Mengubah modul dengan cara mengentik kode program secara manual di file modul
7. Mengubah entitas dengan cara mengentik kode program secara manual di file entitas
8. Menghapus modul yang tidak digunakan
9. Menghapus entitas yang tidak digunakan
10. Menampilkan diagram hubungan entitas atau *entity relationship diagram* (ERD)
11. Mengatur tampilan dan perbesaran ERD
12. Mengunduh ERD dalam format SVG
13. Mengunduh ERD dalam format PNG
14. Mengunduh desktipsi entitas dalam format Markdownn
15. Membuat query alter tabel pada database dari diferensiasi antara database yang sedang digunakan dengan entitas baru pasca pembuatan modul aplikasi atau perubahan pada entitas
16. Membuat query untuk database baru sesuai dengan entitas pasca pembuatan modul aplikasi
17. Mengeksekusi query alter tabel pada database dari diferensiasi antara database yang sedang digunakan dengan entitas baru pasca pembuatan modul aplikasi dan query lainnya
18. Menterjemahkan modul ke berbagai bahasa
19. Menterjemahkan entitas ke berbagai bahasa
20. Mengubah opsi aplikasi untuk lingkungan pengembangan dan produksi
21. File manager
22. Database explorer

## Subsitem MagicAppBuilder

MagicAppBuilder terdiri dari beberapa subsistem yang membentuk sebuah palform yang utuh. Subsistem tersebut antara lain adalah sebagai berikut:

### Database Explorer

Database explorer atau penjelajah database adalah subsistem dari MagicAppBuilder. Penjelajah database ini digunakan untuk keperluan pembuatan aplikasi. Penjelajah database ini memili fitur yang sangat terbatas yaitu sebagai berikut:

1. Memilih Database di sebuah server
2. Menampilkan table di database terpilih
3. Menampilkan struktir table
4. Menampilkan isi table dengan pembagian halaman
5. Menambah baris data
6. Mengubah baris data
7. Menghapus baris data
8. Mengeksekusi query database
9. Mengimpor data
10. Mengkonversi query dari database management system (DBMS) lain menjadi query yang sesuai dengan database yang sedang digunakan
11. Mengekspor struktur tabel dan database
12. Mengekspor data tabel dan database

### Entity Editor

Entity editor atau editor entitas adalah subsistem dari Database Explorer. Editor entitas ini digunakan untuk membuat desain database relasional dengan membuat tabel dan kolom. Entitas-entitas ini akan digambarkan dalam bentuk hubungan antar entitas. Adapun fitur dari editor entitas adalah sebagai berikut:

1. Membuat entitas baru
2. Menambah kolom entitas
3. Mengubah kolom entitas
4. Menghapus kolom entitas
5. Membuat template kolom
6. Mengatur preferensi tipe data dan ukuran dari kolom dan kunci utama
7. Menambahkan beberapa kolom sekaligus dari template
8. Mengimpor entitas dari dari file JSON
9. Mengekspor entitas ke dari file JSON
10. Mengimpor entitas dari dari file SQL
11. Mengekspor entitas ke dari file SQL
12. Membuat diagram dari entitas
13. Mengatur tampilan diagram
14. Mengunduh diagram dalam format SVG
15. Mengunduh diagram dalam format PNG
16. Mengkonversi SQL sesuai dengan database yang sedang digunakan untuk dapat dieksekusi oleh editor database

### Administration

Administration atau Administrasi adalah halaman terpisah untuk mengelola pengguna, workspace dan aplikasi. Meskipun demikian, halaman ini menggunakan database yang sama dengan MagicAppBuilder. Beberapa fitur yang terdapat pada halaman Administrasi adalah sebagai berikut:

1. Master
   1. Application
   2. Application Group
   3. Workspace
   4. Administrator
2. Role
   1. Administrator Workspace
   2. Application Group Member
3. Reference
   1. Administrator Level
4. Message
   1. Message
   2. Notification
5. Cache
   1. Error Cache
6. Database

Database di sini merupakan penjelajah database namun hanya dapat mengakses database yang digunakan oleh MagicAppBuilder. Pengguna diberi keleluasaan untuk mengelola database dari halaman ini. Meskipun demikian, kehati-hatian perlu diperhatikan agar pengguna tidak salah dalam melakukan operasi sehingga menyebabkan suatu masalah di MagicAppBuilder. Pengguna tetap harus masuk ke aplikasi menggunakan username dan password pengguna untuk dapat mengakses halaman ini.

## File Manager

File Manager ini memungkinkan pengguna untuk mengelola file dan direktori dengan berbagai fitur yang mudah digunakan. Pengguna dapat melihat pratinjau gambar, membuka dan mengedit file teks, serta melakukan berbagai operasi pada file dan direktori seperti mengunggah, menghapus, dan mendownload. File Manager pada MagicAppBuilder tidak dilengkapi dengan fitur move file dan copy file karena setiap file memiliki fungsinya masing-masing. Dengan demikian, kesalahan yang diakibatkan oleh penyalinan dan pemindahan file dapat dihindari.

Berikut adalah petunjuk lengkap untuk memanfaatkan berbagai fitur yang ada di dalam File Manager.

### Fitur Utama

**1. Pratinjau Gambar**

Pengguna dapat melihat pratinjau gambar secara langsung di dalam File Manager. Cukup klik pada gambar, dan pratinjau gambar akan muncul, memungkinkan pengguna untuk melihatnya tanpa membuka aplikasi lain.

**2. Buka File Teks**

File teks seperti .txt atau file berbasis teks lainnya dapat dibuka langsung di File Manager. Pengguna hanya perlu mengklik file teks, dan isinya akan muncul di jendela editor di dalam File Manager.

**3. Edit File Teks**

Setelah membuka file teks, pengguna dapat mengeditnya secara langsung. Pengguna cukup mengubah isi file dan menyimpannya, dan perubahan tersebut akan diterapkan pada file asli di direktori.

Menu Konteks File Manager

Menu konteks (klik kanan) memungkinkan pengguna untuk mengakses berbagai fungsi terkait file atau direktori. Menu ini terbagi menjadi beberapa kategori berdasarkan konteks yang berbeda: **Direktori**, **Direktori Root**, dan **File**.

### Menu Konteks Direktori

Menu konteks ini muncul ketika pengguna mengklik kanan pada direktori.

1. **Buat File Baru**
   * Pilih opsi ini untuk membuat file baru di dalam direktori yang sedang dipilih. Pengguna akan diminta untuk memasukkan nama file baru.
2. **Buat Direktori Baru**
   * Pilih opsi ini untuk membuat direktori baru di dalam direktori yang sedang dipilih. Pengguna dapat memberi nama pada direktori baru yang akan dibuat.
3. **Unggah File**
   * Pilih opsi ini untuk mengunggah file dari perangkat pengguna ke dalam direktori yang sedang dipilih. Pengguna dapat memilih satu atau beberapa file untuk diunggah.
4. **Perluas Direktori**
   * Pilih opsi ini untuk menampilkan atau menyembunyikan subdirektori di dalam direktori yang sedang dipilih. Direktori yang sudah memiliki subdirektori akan diperluas dan menampilkan kontennya.
5. **Muat ulang Direktori**
   * Pilih opsi ini untuk me-reload atau memuat ulang konten dari direktori yang sedang dipilih, memastikan konten terbaru ditampilkan.
6. **Ganti Nama Direktori**
   * Pilih opsi ini untuk mengganti nama direktori yang sedang dipilih. Pengguna akan diminta untuk memasukkan nama baru untuk direktori tersebut.
7. **Unduh Direktori**
   * Pilih opsi ini untuk mengunduh seluruh direktori beserta isinya ke perangkat pengguna.
8. **Hapus Direktori**
   * Pilih opsi ini untuk menghapus direktori yang sedang dipilih beserta seluruh isinya.

### Menu Konteks Direktori Root

Menu konteks ini muncul ketika pengguna mengklik kanan pada direktori root (direktori utama).

1. **Buat File Baru**
   * Pilih opsi ini untuk membuat file baru di dalam direktori root. Pengguna akan diminta untuk memasukkan nama file baru.
2. **Buat Direktori Baru**
   * Pilih opsi ini untuk membuat direktori baru di dalam direktori root. Pengguna akan diminta untuk memberi nama pada direktori baru tersebut.
3. **Unggah File**
   * Pilih opsi ini untuk mengunggah file ke direktori root. Pengguna dapat memilih beberapa file sekaligus.
4. **Setel Ulang Konten**
   * Pilih opsi ini untuk menghapus semua konten yang ada di direktori root dan mengembalikannya ke keadaan semula. Semua file dan subdirektori akan dihapus.
5. **Unduh Semua**
   * Pilih opsi ini untuk mengunduh seluruh konten direktori root ke perangkat pengguna.

### Menu Konteks File

Menu konteks ini muncul ketika pengguna mengklik kanan pada file.

1. **Buka File**
   * Pilih opsi ini untuk membuka file yang dipilih. Jika file tersebut adalah file teks, pengguna akan dapat melihat dan mengedit isinya di dalam File Manager. Untuk file lain seperti gambar atau PDF, file akan dibuka di dalam pratinjau.
2. **Ganti Nama File**
   * Pilih opsi ini untuk mengganti nama file yang dipilih. Pengguna akan diminta untuk memasukkan nama baru untuk file tersebut.
3. **Unduh File**
   * Pilih opsi ini untuk mengunduh file ke perangkat pengguna. Pengguna akan mendapatkan file dalam format asli seperti yang ada di server.
4. **Hapus File**
   * Pilih opsi ini untuk menghapus file yang dipilih. Pengguna akan diminta konfirmasi sebelum file dihapus secara permanen.

### Cara Menggunakan File Manager

**1. Navigasi Direktori**

* Klik pada direktori untuk membuka dan melihat isinya.
* Gunakan menu konteks untuk membuat file baru, membuat direktori baru, mengunggah file, atau melakukan operasi lainnya.

**2. Mengunggah File**

* Pilih direktori tempat pengguna ingin mengunggah file.
* Klik kanan pada direktori dan pilih "Unggah File".
* Pilih file yang ingin pengguna unggah dari perangkat pengguna.

**3. Buat Direktori Baru**

* Klik kanan pada direktori tempat pengguna ingin membuat subdirektori baru.
* Pilih opsi "Buat Direktori Baru" dan beri nama direktori baru pengguna.

**4. Ganti Nama File atau Direktori**

* Klik kanan pada file atau direktori yang ingin pengguna ganti namanya.
* Pilih opsi "Ganti Nama" dan masukkan nama baru.

**5. Unduh File atau Direktori**

* Klik kanan pada file atau direktori yang ingin pengguna unduh.
* Pilih opsi "Unduh" untuk file atau "Unduh Direktori" untuk direktori.

**6. Menghapus File atau Direktori**

* Klik kanan pada file atau direktori yang ingin pengguna hapus.
* Pilih opsi "Hapus" dan konfirmasi penghapusan.

# BAB 4 – PANDUAN PENGGUNAAN

## Versi MagicAppBuilder

Dokumen panduan ini didasarkan pada fitur-fitur yang tersedia pada **MagicAppBuilder versi 1.3.1**. Fitur yang dirilis setelah versi tersebut mungkin belum disertakan dalam pembahasan ini. Namun, pemahaman Anda terhadap versi 1.3.1 akan memudahkan adaptasi dengan fitur-fitur baru, mengingat MagicAppBuilder dikembangkan dengan pendekatan yang intuitif.

## Antarmuka Pengguna

Antarmuka pengguna adalah penghubung antara aplikasi dengan pengguna. Antarmuka ini digunakan oleh pengguna untuk berinteraksi dengan aplikasi. Aplikasi ini terdiri dari beberapa tab di yaitu sebagai berikut:

1. **Administration**  
   Antarmuka web terpisah untuk mengelola pengguna, akses pengguna, ruang kerja, aplikasi, dan pengaturan administratif lainnya.
2. **Workspace**  
   Tab ini memungkinkan pengguna untuk membuat dan melihat ruang kerja. Pengguna juga dapat mengatur ruang kerja yang aktif.
3. **Apps**  
   Tab ini memungkinkan pengguna untuk membuat dan melihat aplikasi. Pengguna juga dapat mengatur aplikasi yang aktif.
4. **Select Table**  
   Di tab ini, pengguna dapat memilih tabel dan menentukan:
   * Nama modul
   * Nama entitas
   * Nama menu
   * Opsi konfigurasi untuk pembuatan modul
   * Apakah akan memuat konfigurasi yang telah disimpan sebelumnya untuk modul
5. **Generate Module**  
   Tab ini digunakan untuk mengonfigurasi modul dengan memilih:
   * Kolom yang akan dimasukkan dalam modul
   * Elemen UI untuk Buat, Perbarui, Lihat Detail, Lihat Daftar, dan Ekspor
   * Hubungan antara kolom dan entitas lain atau tabel database
   * Filter data dan opsi pengurutan
   * Fitur tambahan seperti:
     + Aktifkan/Nonaktifkan
     + Urutan manual
     + Ekspor ke CSV
     + Ekspor ke Excel
     + Alur kerja persetujuan
     + Pilihan untuk membuat kode program untuk bagian belakang saja
     + Sampah (*soft delete*)
     + Perenderan daftar berbasis AJAX
6. **ERD (*Entity Relationship Diagram*)**  
   Di tab ini, pengguna dapat membuat diagram ERD untuk satu atau lebih entitas yang dipilih. Pengguna juga dapat menentukan tingkat kedalaman hubungan entitas yang akan ditampilkan.
7. **Query**  
   Tab ini menampilkan kueri database untuk satu atau lebih entitas yang dipilih. Fungsi utamanya adalah untuk menghasilkan:
   * Kueri *CREATE TABLE* dan *ALTER TABLE* setelah entitas dibuat
   * Kueri untuk pembuatan database berdasarkan entitas yang telah didefinisikan dalam aplikasi

Sistem manajemen basis data (*DBMS*) yang didukung:

* MySQL
* MariaDB
* PostgreSQL
* SQLite

1. **Translate Entity**  
   Tab ini memungkinkan pengguna membuat file lokalisasi untuk entitas, sehingga mendukung banyak bahasa dalam aplikasi.
2. **Translate Module**  
   Tab ini memungkinkan pengguna membuat file lokalisasi untuk modul, sehingga mendukung banyak bahasa dalam aplikasi.
3. **Translate Apps**

Tab ini memungkinkan pengguna membuat lokalisasi untuk mendukung banyak bahasa dalam aplikasi. Ada 3 lokalisasi pada tab ini yaitu:

* **Menu Group**  
  Menu Group digunakan untuk menterjemahkan teks group menu aplikasi. Terjemahan ini disimpan di database alih-alih file dan digunakan untuk membuat menu bagi masing-masing level pengguna sesuai dengan bahasa yang digunakan. Menu yang telah dibuat kemudian disimpan ke database dan dapat langsung diambil dan di-*render* untuk ditampilkan di layar pengguna.
* **Menu**  
  Sama dengan Group Menu namun berlaku pada menu.
* **Validation**

Pada bagian Validation, terjemahkan disimpan sebagai file dan digunakan untuk membuat pesan validasi. Meskipun demikian, file dibaca sebelum validasi dilakukan. Dengan demikian, meskipun semua input memenuhi aturan, namun pesan tetap diproses meskipun tidak pernah ditampilkan di layar pengguna. Jika file validasi tidak ditemukan, proses validasi akan menggunakan template bawaan dari MagicObject.

1. **Edit Entity**  
   Tab ini memungkinkan pengguna untuk membuat entitas baru, mengedit kode entitas secara manual dan menghapus file entitas. Pembuatan entitas baru dilakukan secara otomatis dengan memilih tabel dan menulis nama entitas yang akan dibuat.

MagicAppBuilder akan memeriksa kode yang dibuat oleh pengguna. Apabila terdapat error, MagicAppBuilder akan memberitahukan posisi errornya. Posisi yang ditunjukkan mungkin tidak tepat pada penyebab kesalahannya.

1. **Edit Module**  
   Tab ini memungkinkan pengguna untuk mengedit kode modul secara manual. Pengguna juga dapat menghapus file modul.

MagicAppBuilder tidak akan memeriksa kode yang dibuat oleh pengguna. Untuk itu pengguna harus berhati-hati agar tidak terjadi error.

1. **Edit Validator**

Tab ini memungkinkan pengguna mengelola kelas validator. Pengguna bisa mengubah aturan validasi dengan menambah, mengubah, atau menghapus anotasi validasi beserta atributnya. Untuk menyimpan perubahan, pengguna dapat menggunakan tombol **Save Validator**.

MagicAppBuilder akan secara otomatis memeriksa kode yang pengguna buat. Jika terdeteksi kesalahan, MagicAppBuilder akan memberikan informasi posisi error. Perlu diketahui bahwa posisi yang ditunjukkan mungkin tidak selalu menjadi penyebab langsung dari kesalahan tersebut.

Jika pengguna ingin menyimpan sebuah validator dengan nama lain, pilih tombol **Save Validator As**. Pengguna akan diminta untuk memasukkan nama baru. Nama file akan sesuai dengan nama dasar kelas. Setelah tersimpan dengan nama baru, pengguna dapat mengubahnya sesuai dengan kebutuhan.

Pengguna juga dapat menguji validator langsung dari halaman ini. Cukup pilih kelas validator yang akan diuji lalu pilih tombol **Test Validator**, dan MagicAppBuilder akan menampilkan formulir yang berisi properti yang divalidasi. Untuk memulai pengujian, klik tombol **OK**. MagicAppBuilder kemudian akan menampilkan hasilnya. Fitur ini memungkinkan pengguna menguji kelas validator tanpa perlu mengintegrasikannya dengan aplikasi, sehingga meminimalkan data sampah dari proses pengujian.

Untuk menghapus sebuah file validator, pengguna dapat menggunakan tombol **Delete Validator**. Pengguna harus memilih validator yang akan dihapus sebelum memilih tombol tersebut.

**PERHATIAN!**

Berhati-hatilah saat akan menghapus sebuah validator karena validator tersebut mungkin sedang digunakan oleh sebuah modul atau validator lain. Jika aplikasi kehilangan file validator, maka akan terjadi **Fatal Error**. Pengguna lanjut dapat menangani **Fatal Error** ini di PHP 7 dengan kode sebagai berikut:

catch(\Error $e)

{

// Tangani eksepsi di sini

}

Kode tersebut harus dittulis sebelum

catch(Exception $e)

{

}

MagicAppBuilder tidak menyediakan ini secara default karena berpotensi akan menimbulkan masalah di PHP versi 5.

Pengguna dapat mengubah aturan validasi dengan *Graphical User Interface* (GUI) dengan memilih tombol **Update Validator**. Update validator hanya akan memvalidasi kolom-kolom dari tabel yang bersangkutan dan tidak mendukung validasi bersarang dengan menggunakan validator lain. Pengguna dapat menambahkan, mengubah, mengurangi, atau bahkan menghapus validasi pada setiap properti entitas. Setelah melakukan perubahan, pilih tombol **Update** untuk menyimpan perubahan.

Untuk membuat validator baru, pengguna dapat memilih tombol **Create Validator**. Pengguna akan diminta untuk memilih tabel sebagai referensi dan memasukkan nama kelas validator. Perhatikan bahwa membuat validator baru akan menimpa file yang sudah ada jika namanya sama. Penting untuk mengorganisir kelas validator untuk mencegah terjadinya error dan prilaku aplikasi yang tidak dikehendaki.

1. **File Manager**

Tab ini memungkinkan pengguna untuk melihat dan mengelola file-file aplikasi termasuk membuat kode program. Meskipun demikian, sangat disarankan agar pengguna menggunakan *Integrated Development Enviroment* (IDE) yang sudah mendukung file PHP, HTML, CSS, JavaScript, Yaml dan INI untuk membuat kode program yang panjang.

1. **Logout**  
   Tautan untuk keluar dari sesi administrator.

## Langkah-Langkah Penggunaan

Panduan ini akan membantu pengguna melalui setiap langkah, dari instalasi hingga pembuatan modul, untuk memastikan pengalaman yang lancar dengan platform ini. Langkah 1 dan 2 adalah tahap persiapan yang hanya perlu dilakukan sekali.

Berikut adalah terjemahan langkah-langkah instalasi server untuk MagicAppBuilder ke dalam bahasa Indonesia:

### Langkah 1: Instalasi Server

Sebelum dapat menggunakan MagicAppBuilder, pengguna harus menginstal server terlebih dahulu. Server ini harus mencakup komponen-komponen berikut:

**Komponen yang Dibutuhkan**

Komponen yang dibutuhkan untuk menjalankan MagicAppBuilder adalah sebagai berikut:

* **Web Server:** Apache
* **Database:** MySQL atau MariaDB
* **PHP:** Untuk skrip sisi server

**Opsi Instalasi**

Terdapat beberapa pilihan untuk menginstal komponen-komponen ini di komputer pengguna:

* **WAMP** (Windows, Apache, MySQL, PHP)
* **XAMPP** (Cross-platform Apache, MySQL, PHP)
* **USBWebServer** (Versi portabel untuk Windows)

Pilih salah satu yang paling sesuai dengan sistem operasi dan kebutuhan pengguna. Setiap alat ini menyediakan proses instalasi yang sederhana dengan komponen yang telah dikonfigurasi sebelumnya, sehingga pengguna tidak perlu mengatur setiap komponen secara manual.

**Setelah Instalasi**

Setelah menginstal MagicAppBuilder, pastikan semua komponen lingkungan berjalan dengan baik:

1. **Jalankan Apache (Web Server)**
   * Pastikan server web Apache berjalan. Pengguna dapat memulainya melalui panel kontrol XAMPP (jika menggunakan XAMPP) atau melalui alat manajemen server web yang relevan.
2. **Jalankan MySQL atau MariaDB (Database Server)**
   * Jika pengguna menggunakan MySQL atau MariaDB sebagai basis data, pastikan server basis data berjalan. Pengguna dapat memulainya dari panel kontrol XAMPP atau antarmuka manajemen basis data yang digunakan.
3. **Verifikasi PHP Berfungsi**
   * Untuk memastikan PHP berfungsi dengan baik, akses halaman PHP default (biasanya *http://localhost*) di browser. Jika halaman PHP yang diharapkan muncul, maka PHP telah terinstal dan berjalan dengan baik.

**Mengaktifkan Driver SQLite di Windows**

Jika pengguna menggunakan sistem operasi Linux, driver SQlite sudah termasuk dalam paket instalasi dan secara default sudah aktif. Jika pengguna menggunakan sistem operasi Windows, pengguna harus memastikan bahwa driver SQLite untuk PHP diaktifkan dalam konfigurasi PHP. Jika driver SQlite tidak diaktifkan, MagicAppBuilder tidaka akan dapat digunakan.

Langkah-langkah Mengaktifkan Driver SQLite di Windows adalah sebagai berikut:

1. **Cari File php.ini**
   * File **php.ini** adalah file konfigurasi utama untuk PHP. Lokasi file ini tergantung pada instalasi PHP pengguna, tetapi jika menggunakan XAMPP, biasanya dapat ditemukan di:
   * C:\xampp\php\php.ini
2. **Buka File php.ini**
   * Buka file **php.ini** dengan editor teks seperti Notepad atau Visual Studio Code (VSCode).
3. **Temukan Ekstensi SQLite**
   * Cari baris berikut dalam file **php.ini** (gunakan fungsi *Find* dengan menekan **Ctrl + F** di editor teks):
   * extension=sqlite3
   * extension=pdo\_sqlite
4. **Hapus Tanda Koma (;)**
   * Biasanya, baris-baris ini akan dikomentari secara default dengan tanda titik koma (;) di awal baris. Untuk mengaktifkan ekstensi SQLite, hapus tanda titik koma tersebut sehingga menjadi:
   * extension=sqlite3
   * extension=pdo\_sqlite
5. **Simpan dan Tutup File php.ini**
   * Setelah menghapus tanda titik koma, simpan perubahan dan tutup editor teks pengguna.
6. **Restart Apache**
   * Agar perubahan konfigurasi diterapkan, restart server Apache. Jika menggunakan XAMPP, pengguna dapat menghentikan Apache dengan menekan tombol **Stop**, lalu menjalankannya kembali dengan menekan tombol **Start**. Atau, gunakan alat manajemen server web lainnya yang digunakan.
7. **Verifikasi SQLite Sudah Aktif**

Untuk memastikan bahwa ekstensi SQLite telah diaktifkan, buat skrip PHP sederhana untuk menampilkan ekstensi yang terinstal:

* + Buat file bernama **phpinfo.php** di direktori root server web (misalnya, di **C:\xampp\htdocs** jika menggunakan XAMPP).
  + Tambahkan kode berikut ke dalam file tersebut:   
    <?php phpinfo();?>
  + Akses file ini di browser dengan membuka <http://localhost/phpinfo.php>.
  + Cari bagian SQLite dalam tampilan hasil. Jika ekstensi SQLite telah diaktifkan, pengguna akan melihat entri untuk SQLite3 dan PDO\_SQLite.
  + Setelah dipastikan bahwa driver SQLite sudah aktif, jangan lupa untuk menghapus file **phpinfo.php** yang dibuat karena file ini bisa menjadi celah keamanan jika diakses oleh orang yang tidak bertanggung jawab.

Setelah langkah-langkah ini selesai, driver SQLite akan aktif untuk PHP di server Windows. Jika MagicAppBuilder telah dikonfigurasi untuk menggunakan SQLite, maka kini aplikasi dapat terhubung ke basis data SQLite tanpa masalah.

### Langkah 2: Mengunduh dan Memasang MagicAppBuilder di Server

Untuk memasang MagicAppBuilder di server, ikuti langkah-langkah berikut:

* 1. **Kunjungi Repositori GitHub MagicAppBuilder**

Buka repositori resmi MagicAppBuilder di:  
https://github.com/Planetbiru/MagicAppBuilder

Sebuah gambar berisi teks, cuplikan layar, nomor, software

Konten yang dihasilkan AI mungkin salah.

* 1. **Unduh Rilis Terbaru**

Navigasikan ke halaman rilis di:  
https://github.com/Planetbiru/MagicAppBuilder/releases

Sebuah gambar berisi teks, elektronik, cuplikan layar, software

Konten yang dihasilkan AI mungkin salah.

* 1. **Ekstrak File ke Root Dokumen Server**

Setelah mengunduh rilis, ekstrak file ZIP ke direktori Root Dokumen server web. Direktori ini biasanya:

* **Linux**: /var/www/html
* **Windows (XAMPP)**: C:\xampp\htdocs

Pastikan file diekstrak ke dalam direktori bernama **MagicAppBuilder** agar aplikasi dapat diakses dengan benar melalui jalur URL.

* 1. **Verifikasi Operasi Server Web**

Pastikan server web (misalnya Apache atau Nginx) berjalan dengan baik. Pengguna bisa memeriksa status server atau memastikan situs web lain yang dihosting di server yang sama berfungsi dengan benar.

* 1. **Akses MagicAppBuilder dari Browser**

Buka browser web (disarankan Mozilla Firefox versi terbaru). Masukkan URL berikut di bilah alamat:  
http://localhost/MagicAppBuilder

Jika server di-host di mesin lain, ganti **"localhost"** dengan alamat IP atau nama domain yang sesuai.

Sebuah gambar berisi teks, cuplikan layar, software, komputer

Konten yang dihasilkan AI mungkin salah.

* 1. **Masuk ke Aplikasi**

Saat mengakses URL, halaman login MagicAppBuilder akan muncul. Gunakan kredensial default berikut:

* **Username**: administrator
* **Password**: administrator

Masukkan kredensial dengan benar (tanpa tanda kutip) dan klik tombol **Login**.

Sebuah gambar berisi teks, cuplikan layar, nomor, software

Konten yang dihasilkan AI mungkin salah.

* 1. **Setelah Login**

Setelah berhasil masuk, pengguna akan diarahkan ke **dashboard MagicAppBuilder**, di mana pengguna dapat mulai mengonfigurasi dan mengelola aplikasi.

* 1. **Mengganti Password**

Sangat disarankan pengguna mengganti password pengguna setelah berhasil login. Jika pengguna masuk sebagai superuser, pengguna dapat masuk ke halaman Administrasi. Jika pengguna masuk bukan sebagai superuser, pengguna dapat masuk ke halaman Profile.

Halaman Administrasi digunakan untuk mengatur administrasi pengguna, workspace dan aplikasi. Selain itu, halaman Administrasi juga dapat digunakan untuk mengganti password pengguna lain.

Sebuah gambar berisi teks, cuplikan layar, software, nomor

Konten yang dihasilkan AI mungkin salah.

Pengguna juga dapat mengganti bahasa yang pengguna gunakan dari halaman Administrasi. Lokalisasi ini hanya berlaku di halaman Administrasi sehingga tidak akan mengubah bahasa pada MagicAppBuilder.

Sebuah gambar berisi teks, cuplikan layar, software, nomor

Konten yang dihasilkan AI mungkin salah.

Untuk mengganti password akun sendiri, pilih ikon user di kanan atas. Sebuah dropdown akan muncul. Pilih **Profile**. Administrasi akan menampikan informasi profil pengguna. Klik tombol **Update**. Administrasi akan menampilkan formulir yang berisi isian sebagai berikut:

* Name
* Username
* Password
* Gender
* Birth Day
* Email
* Phone

Silakan isi dengan data yang sesuai dan pilih tombol **Save**.

Setelah pengguna berhasil mengubah password, pengguna dapat keluar dari halaman Administrasi dengan memilih menu MagicAppBuilder.

* 1. **Mengatur Ulang Password**

Dalam kasus pengguna lupa password akun, ada 4 cara yang dapat dilakukan untuk dapat mengakses kembali MagicAppBuilder. Ketiga cara tersebut adalah sebagai berikut:

* Menghapus database dan mengakses kembali MagicAppBuilder dari awal
* Mengubah konfigurasi MagicAppBuilder agar mengarah ke database yang baru
* Mengatur ulang password dengan menggunakan MagicAppBuilder
* Mengatur ulang password menggunakan aplikasi pihak ketiga yang dapat mengubah data di database SQLite

**Cara 1: Menghapus Database**

Cara ini adalah cara paling mudah untuk dilakukan. Meskipun mudah, cara ini memiliki kekurangan. Kekurangan dari cara ini adalah pengguna akan kehilangan data-data seperti pengguna, workspace, dan mengaturan akses pengguna terhadap workspace dan aplikasi. Meskipun demikian, data-data konfigurasi aplikasi yang telah dibuat tidak akan hilang jika pengguna kembali membuat workspace di lokasi yang sama. MagicAppBuilder akan memindai lokasi di mana pengguna membuat sebuah workspace baru. Pengguna juga dapat melakukan pemindaian lokasi dengan cara memilih tombol **Scan** pada workspace.

Untuk dapat menghapus file database MagicAppBuilder, pengguna harus menemukan lokasinya terlebih dahulu. Kemudian hapus atau ganti nama file tersebut. MagicAppBuilder akan kehilangan akses ke database dan akan membuat database baru.

**Cara 2: Mengubah Konfigurasi**

Cara ini lebih aman digunakan dibandingkan dengna cara pertama. Meskipun MagicAppBuilder akan membuat database baru, namun pengguna tetap memiliki database lama sehingga pengguna memiliki kesempatan di lain waktu untuk mengatur ulang password sembari dapat terus mengerjakan pembuatan aplikasi.

Untuk mengubah konfigurasi MagicAppBuilder, pengguna harus menemukan file konfigurasi MagicAppBuilder terlebih dahulu. Secara default, lokasinya berada di dalam subdirektori **inc.cfg** di dalam direktori MagicAppBuilder. Ubah lokasi file database lalu simpan file tersebut. Kemudian akses kembali MagicAppBuilder. MagicAppBuilder akan kehilangan akses ke database dan akan membuat database baru.

**Cara 3: Mengatur Ulang Password**

Cara ini sebenarnya tidak terlalu sulit namun bagi pengguna awam mungkin menjadi tantangan tersendiri. MagicAppBuilder dibuat sebagai sebuah sistem mandiri sehingga pengguna tidak akan dapat mengirimkan link reset password ke email atau menggunakan *one time password* (OTP). Satu-satunya cara adalah pengguna mengakses langsung ke dalam sistem dengan cara yang tidak dapat dilakukan oleh orang lain dari perangkat di luar server yang digunakan.

Untuk melakukan reset password dari MagicAppBuilder, klik tombol **Reset Password** di formulir login. Kemudian pengguna harus masuk ke direktori **inc.cfg** di dalam direktori MagicAppBuilder dan membuat sebuah file dengan nama **reset-password.yml**. Tulis di dalam file reset-password.yml kode Yaml sebagai berikut:

passwordToReset:

-

username: "username1"

password: "new-password-1"

Ganti **username1** dengan username pengguna dan **new-password-1** dengan password baru. Jika pengguna ingin melakukan pengaturan ulang lebih dari satu akun, ulangi mulai dari tanda “-“.

Contoh:

passwordToReset:

-

username: "username1"

password: "new-password-1"

-

username: "username2"

password: "new-password-2"

Setetelah pengguna selesai menulis kode Yaml tersebut, kembali ke browser lalu klik reset password. Pastikan bahwa file **reset-password.yml** terghapus otomatis oleh MagicAppBuilder. Jika tidak, pengguna harus menghapusnya secara manual demi keamanan.

Setelah password berhasil diatur ulang, kembali ke formulir login dengan memilih tombol **Login Form** lalu masuk dengan username dan password yang baru.

**Cara 4: Menggunakan Aplikasi Pihak Ketiga**

Cara ini hanya dapat dilakukan oleh pengguna tingkat lanjut. Pengguna dapat menggunakan aplikasi SQLite Viewer atau sejenisnya. SQLite merupakan sistem database embedded berbasis file. Operasi database langsung ditangani oleh aplikasi tanpa memerlukan aplikasi tambahan. Dengan semikian, pengguna dapat menggunakan SQLite Viewer untuk mengubah password.

Password berada di tabel **admin** dan disimpan di kolom **password**. Pastikan bahwa pengguna hanya mengubah password pada baris yang benar yaitu baris akun pengguna. Setelah operasi dijalankan, pengguna dapat mengakses kembali MagicaAppBuilder dengan cara biasa.

**Mengonfigurasi SQLite sebagai Database Default**

Secara default, MagicAppBuilder menggunakan **SQLite**, yang ringan dan mudah dikonfigurasi. Saat pengguna membuat aplikasi baru, MagicAppBuilder akan secara otomatis membuat file database dalam direktori aplikasi tanpa memerlukan server database terpisah.

**Beralih ke MySQL atau PostgreSQL**

Jika pengguna menginginkan solusi database yang lebih skalabel, pengguna bisa menggunakan **MySQL** atau **PostgreSQL** dengan mengedit file konfigurasi core.yml.

Berikut ini adalah langkah-langkah untuk mengubah konfigurasi MagicAppBuilder.

1. **Temukan File core.yml**

File ini berada di dalam direktori **inc.cfg** dalam folder instalasi MagicAppBuilder. Buka file tersebut dengan editor teks.

1. **Edit Konfigurasi Database**

Di dalam core.yml, cari bagian yang terkait dengan database dan ubah pengaturan driver serta detail koneksi.

**Contoh untuk MySQL:**

dataLimit: 20

database:

driver: mysql

host: 'localhost'

port: 3306

username: 'root'

password: 'YourPasswordHere'

databaseName: 'your\_database\_name'

databaseSchema: 'public'

timeZone: 'Asia/Jakarta'

**Contoh untuk PostgreSQL:**

dataLimit: 20

database:

driver: postgresql

host: 'localhost'

port: 5432

username: 'postgres'

password: 'YourPasswordHere'

databaseName: 'your\_database\_name'

databaseSchema: 'public'

timeZone: 'Asia/Jakarta'

**Konfigurasi Default untuk SQLite:**

dataLimit: 20

database:

driver: sqlite

databaseFilePath: 'D:\database\db.sqlite'

# sesuaikan dengan

1. **Simpan Perubahan**

Setelah mengedit file core.yml, simpan dan tutup editor teks.

1. **Verifikasi Koneksi Database**

* Restart server web.
* Periksa apakah MagicAppBuilder berhasil terhubung ke database dengan menjalankan aplikasi dan memastikan tidak ada error koneksi.

1. **Membuat Tabel Database**

Setelah koneksi database berhasil, pengguna bisa mulai membuat tabel dan mengelola data menggunakan **Database Explorer** dan **Entity Editor** di MagicAppBuilder.

**Pertimbangan Tambahan**

* **Keamanan:** Segera ubah kredensial default administrator setelah login pertama untuk meningkatkan keamanan aplikasi.
* **Kompatibilitas Browser:** Disarankan menggunakan **Mozilla Firefox terbaru** untuk pengalaman terbaik.
* **Kinerja SQLite:** SQLite cocok untuk aplikasi kecil hingga menengah. Jika aplikasi berkembang besar dan kompleks, pertimbangkan migrasi ke **MySQL** atau **PostgreSQL**.

Dengan mengikuti langkah-langkah ini, MagicAppBuilder akan siap digunakan dengan konfigurasi database yang dipilih. Pastikan server dikonfigurasi dengan benar dan semua dependensi yang diperlukan telah diinstal.

### Langkah 3: Membuat Workspace

Setelah MagicAppBuilder berhasil diinstal, langkah berikutnya adalah **membuat Workspace**. Workspace adalah direktori yang berisi beberapa proyek. Setiap workspace terisolasi, memungkinkan pengelolaan akses admin yang lebih baik.

**Cara Membuat Workspace**

Cara membuat sebuah workspace adalah sebagai berikut:

1. Buka MagicAppBuilder.
2. Navigasikan ke tab **Workspace**.
3. Tentukan direktori workspace baru.
4. Klik tombol **Save**.

Sebuah gambar berisi teks, cuplikan layar, software, komputer

Konten yang dihasilkan AI mungkin salah.

Sebuah gambar berisi teks, cuplikan layar, software, Ikon komputer

Konten yang dihasilkan AI mungkin salah.

Sebuah gambar berisi teks, cuplikan layar, software, Ikon komputer

Konten yang dihasilkan AI mungkin salah.

### Langkah 4: Membuat Aplikasi

Setelah workspace dibuat, pengguna bisa melanjutkan ke langkah berikutnya, yaitu membuat aplikasi.

Untuk membuat aplikasi baru, navigasi ke tab **Apps**. MagicAppBuilder akan menampilkan halaman baru yang berisi dua tombol dan satu filter. Klik tombol **Create New**. MagicAppBuilder akan menampilkan sebuah dialog dengan formulir. Isi formulir dengan informasi berikut:

* **Application Name**: Pilih nama untuk aplikasi.
* **Application ID**: ID unik untuk aplikasi, digunakan secara internal oleh MagicAppBuilder.
* **Architecture**: Pilih antara **Monolith** (struktur tunggal) atau **Microservices** (layanan terdistribusi).
* **Description**: Ringkasan fungsionalitas aplikasi.
* **Installation Method:** Metode instalasi (Online atau Offline)
* **MagicApp Version**: Versi MagicApp yang akan digunakan (disarankan versi terbaru). Jika menggunakan metode instalasi offline, MagicAppBuilder akan menyertakan versi MagicApp yang sama dengan versi yang digunakan oleh MagicAppBuilder ke aplikasi. Versi ini dapat diperbaru di masa mendatang.
* **Workspace**: Pilih workspace yang telah dibuat sebelumnya.
* **Application Directory**: Tentukan folder tempat file aplikasi akan disimpan.
* **Base Namespace**: Namespace dasar untuk kode sumber aplikasi, misalnya AppNamespace.
* **Author**: Masukkan nama penulis atau perusahaan.
* **Path**: Direktori tempat modul aplikasi akan dibuat.

Setelah semua kolom diisi, klik tombol **Save**. MagicAppBuilder akan menampilkan aplikasi yang telah dibuat dalam bentuk **kartu aplikasi**.

Dengan mengikuti langkah-langkah ini, pengguna dapat mulai membangun dan mengelola aplikasi menggunakan MagicAppBuilder!

### Langkah 5: Konfigurasi Aplikasi

Selanjutnya, pengguna perlu mengonfigurasi aplikasi pengguna melalui **Application Setting**. Klik tombol **Setting** pada kartu aplikasi. MagicAppBuilder akan menampilkan sebuah dialog dengan formulir yang dibagi menjadi beberapa bagian yaitu sebagai berikut:

1. **Application**   
   Pengguna dapat mengubah nama aplikasi, arsitektur aplikasi, deskripsi aplikasi, direktori aplikasi, path aplikasi, pilihan gaya menu dan tema yang digunakan.
2. **Database**   
   Secara default, MagicAppBuilder menggunakan **SQLite** dengan file database yang tersimpan dalam direktori aplikasi. Jika ingin menggunakan database lain seperti **MySQL** atau **PostgreSQL**, pengguna dapat mengubah pengaturannya.
3. **Session and Cookie**   
   Atur parameter sesi seperti **nama sesi**, **masa aktif sesi**, **Session Save Handler**, dan **Session Save Path**. Selain itu juga terdapat pengaturan cookie seperti **cookie path, cookie domain, cookie secure, cookie HTTP only** dan **cookie same site**.
4. **Reserved Columns**   
   Terrdapat bagian penting yang disebut **Reserved Columns** (Kolom yang Direservasi) yang harus dikonfigurasi sejak awal. **Kolom ini tidak boleg diubah** setelah database, entitas, dan modul mulai dibuat.

Kolom yang direservasi meliputi:

1. name
2. active
3. draft
4. waiting\_for
5. admin\_create
6. admin\_edit
7. admin\_ask\_edit
8. admin\_delete
9. admin\_restore
10. time\_create
11. time\_edit
12. time\_ask\_edit
13. time\_delete
14. time\_restore
15. ip\_create
16. ip\_edit
17. ip\_ask\_edit
18. ip\_delete
19. ip\_restore
20. sort\_order
21. approval\_id
22. approval\_note
23. approval\_status
24. restored

**Pemetaan Kolom yang Direservasi**

Kolom yang direservasi ini dapat dipetakan ke nama lain sesuai dengan bahasa asli yang digunakan dalam aplikasi serta terminologi yang akan digunakan pada setiap entitas. Setiap entitas harus secara konsisten menggunakan nama kolom yang sudah dipetakan.

**Contoh Penggunaan Kolom Reserved dalam Entitas**

1. **Entitas Album**
   * Menggunakan kolom **sort\_order** untuk mengurutkan album. **Pengurutan data harus menggunakan kolom ini** dan bukan kolom lain.
   * Menggunakan kolom **active** untuk mengaktifkan dan menonaktifkan album. **Harus menggunakan kolom ini** untuk mengelola status aktif.
2. **Entitas Artis**
   * **Tidak memerlukan** kolom **sort\_order** karena data artis tidak perlu diurutkan secara default oleh pengguna.
   * **Tetap menggunakan** kolom **active** untuk mengaktifkan dan menonaktifkan artis.

**Pemetaan Nama Kolom ke Bahasa Indonesia**

Jika aplikasi dibangun dalam bahasa selain Inggris, akan terasa aneh menggunakan nama kolom seperti active, admin\_create, ip\_create, dan sebagainya kecuali kolom-kolom tersebut tidak dapat diakses oleh pengguna biasa. Oleh karena itu, pengembang bebas memilih nama lain, tetapi **harus** membuat pemetaan kolom.

Contoh dari **sortOrder** adalah data referensi, seperti genre lagu, yang perlu diurutkan berdasarkan jumlah genre yang diproduksi oleh sebuah studio. Contoh lainnya adalah mengurutkan jenis pengguna aplikasi berdasarkan tingkat otoritas mereka. Jenis pengguna dengan otoritas lebih tinggi dapat ditempatkan di bagian atas, sehingga saat menetapkan peran, pengguna dapat dengan mudah mengidentifikasi jenis dengan otoritas tertinggi dan terendah.

Setelah semua konfigurasi diisi, klik **Save**. Pengguna selalu dapat kembali dan mengubah pengaturan ini, termasuk kolom yang telah dicadangkan selama belum mulai membuat database dan modul.

Contoh pemetaan kolom ke Bahasa Indonesia:

| **Nama Kolom Asli** | **Bahasa Indonesia** | **Deskripsi** |
| --- | --- | --- |
| name | nama | Mewakili satu baris dalam sebuah entitas. |
| active | aktif | Untuk mengaktifkan atau menonaktifkan data. |
| draft | draft | Menandai data baru yang menunggu persetujuan. |
| waiting\_for | waiting\_for | Menentukan persetujuan yang diperlukan. |
| admin\_create | admin\_buat | ID pengguna yang membuat data. |
| admin\_edit | admin\_ubah | ID pengguna yang terakhir mengedit data. |
| admin\_ask\_edit | admin\_minta\_ubah | ID pengguna yang meminta pengeditan. |
| time\_create | waktu\_buat | Waktu saat data dibuat. |
| time\_edit | waktu\_ubah | Waktu saat data terakhir diubah. |
| time\_ask\_edit | waktu\_minta\_ubah | Waktu saat permintaan edit dibuat. |
| ip\_create | ip\_buat | Alamat IP tempat data dibuat. |
| ip\_edit | ip\_ubah | Alamat IP tempat data terakhir diubah. |
| ip\_ask\_edit | ip\_minta\_ubah | Alamat IP tempat permintaan edit dibuat. |
| sort\_order | sort\_order | Digunakan untuk mengurutkan data. |
| approval\_id | approval\_id | ID data dalam tabel persetujuan. |
| approval\_note | approval\_note | Catatan untuk persetujuan. |
| approval\_status | approval\_status | Status persetujuan. |

### Langkah 6: Menyiapkan Struktur Database

Setelah pengguna mengonfigurasi aplikasi, langkah berikutnya adalah menyiapkan struktur database. **MagicAppBuilder** menyediakan dua alat utama untuk ini:

1. Database Explorer
2. Entity Editor

Pengguna juga dapat menggunakan aplikasi pihak ketiga untuk membuat struktur database jika lebih disukai.

**Menggunakan Database Explorer dan Entity Editor**

Untuk memulai, klik tombol **Database** pada kartu aplikasi. **MagicAppBuilder** akan menampilkan dialog **Database Explorer**.

Proses pembuatan struktur database akan dijelaskan dalam bagian terpisah. Namun, dengan menggunakan **Database Explorer** dan **Entity Editor**, pengguna dapat dengan mudah mengelola skema database dan entitas.

### Langkah 7: Membuat Entitas dengan Entity Editor

Untuk membuat dan mengelola entitas menggunakan **Entity Editor** dalam **MagicAppBuilder**, ikuti langkah-langkah berikut:

* 1. Buka Entity Editor
* Masuk ke **Database Explorer**.
* Di bagian bawah, temukan tombol **Entity Editor** dan klik untuk membuka antarmuka **Entity Editor**.
  1. Antarmuka Entity Editor

**Entity Editor** memiliki dua bagian utama:

* **Main Bar**: Terletak di bagian atas, berisi berbagai tombol untuk mengelola entitas.
* **Sidebar**: Menampilkan struktur entitas yang dipilih, termasuk kolom, tipe data, dan detail lainnya.
  1. Aksi yang Tersedia di Main Bar dan Sidebar

Di bagian bawah **Main Bar** dan **Sidebar**, terdapat tombol-tombol berikut:

* **Add New Entity**: Untuk membuat entitas baru.
* **Upload Entity**: Untuk mengimpor entitas dari file JSON.
* **Download Entity**: Untuk mengekspor entitas ke file JSON.
* **Upload SQL**: Untuk mengimpor entitas dari file SQL.
* **Download SQL**: Untuk mengekspor entitas ke file SQL.
* **Download SVG**: Untuk mengunduh diagram entitas dalam format SVG.
* **Download PNG**: Untuk mengunduh diagram entitas dalam format PNG.
* **Sort Entity**: Untuk mengurutkan entitas secara alfabetis.
  1. Membuat Entitas Baru
* Klik tombol **Add New Entity**.
* Isi nama entitas baru, lalu tekan **Enter** pada keyboard.
  1. Menambahkan Kolom ke Entitas
* Setelah nama entitas ditentukan, **Entity Editor** akan menampilkan formulir untuk menambahkan kolom.
* Klik tombol **Add Column** untuk menambahkan kolom baru.
* Isi nama kolom, lalu tekan **Enter** untuk mengonfirmasi.
* Ulangi proses ini untuk setiap kolom yang diperlukan.
  1. Konvensi Penamaan Primary Key

Primary key dari entitas harus mengikuti konvensi penamaan tertentu. Nama primary key harus sesuai dengan nama entitas, dengan akhiran **\_id**.  
Contoh: Jika nama entitas adalah **user**, maka primary key harus bernama **user\_id**.

* 1. Kolom yang Dicadangkan dan Template
* Sebelumnya, pengguna telah mengonfigurasi **kolom yang dicadangkan**. Kolom-kolom ini dapat digunakan untuk membuat template.
* Klik tombol **Edit Template** untuk mengedit template dengan kolom yang telah ditentukan sebelumnya (misalnya, **name, active**, dll.).

Di dalam **Template Editor**, tersedia tombol-tombol berikut:

* **Add Column**: Untuk menambahkan kolom ke template.
* **Save Template**: Untuk menyimpan template untuk penggunaan mendatang.
* **Cancel**: Untuk membatalkan pengeditan template.

Template berguna untuk menambahkan beberapa kolom yang selalu sama, mengurangi kesalahan, dan memastikan konsistensi dalam struktur database.

* 1. Mengatur Preferensi

Klik tombol **Preferences** untuk mengonfigurasi pengaturan kolom.

Di dalam dialog **Preferences**, pengguna dapat mengatur:

* **Primary Key Type**: Tentukan tipe data primary key (misalnya, **INT, UUID**).
* **Primary Key Length**: Tentukan panjang data primary key.
* **Column Type**: Pilih tipe data untuk kolom selain primary key (misalnya, **VARCHAR, INT, DATE**).
* **Column Length**: Tentukan panjang data untuk kolom selain primary key.
* Setelah selesai, klik **OK** untuk menerapkan perubahan atau **Cancel** untuk membatalkan.

Setelah pengguna selesai menambahkan kolom dan mengatur preferensi, jangan lupa untuk menyimpan entitas dengan mengklik tombol **Save Entity**.  
Lakukan hal yang sama untuk entitas lainnya. Template dan preferensi tidak perlu diubah lagi. Pengguna selalu bisa kembali untuk mengedit entitas dan kolomnya jika diperlukan.

Dengan langkah-langkah ini, pengguna dapat membuat dan mengelola entitas secara efektif menggunakan **Entity Editor**, sehingga mempercepat proses pembuatan skema database aplikasi.

* 1. Menerapkan Entitas ke Database

Setelah pengguna membuat semua entitas, langkah berikutnya adalah menerapkannya ke database.

* **Pilih Semua Entitas**: Centang semua entitas di sidebar.
* **Impor Entitas**: Klik tombol **Import** di bagian bawah dialog. Ini akan membawa pengguna kembali ke **Database Explorer**. Di dalam **Database Explorer**, pengguna akan melihat perintah **"CREATE TABLE"** di editor query.
* **Jalankan Query**: Klik tombol **Execute** untuk menjalankan query. **MagicAppBuilder** akan membuat tabel di database sesuai dengan desain yang telah pengguna buat. Pastikan bahwa semua entitas berhasil diimpor ke dalam database.

Setelah proses pembuatan database selesai, pengguna dapat keluar dari **Database Explorer** dan mulai membuat modul aplikasi.

### Langkah 8: Membuat Menu Aplikasi

1. Buka Kartu Aplikasi  
   Kembali ke kartu aplikasi dan klik tombol Menu.
2. Tentukan Menu:

* Pilih menu aplikasi yang ingin pengguna buat atau modifikasi.
* Pengguna bisa membuat menu baru, menambahkan item ke menu yang sudah ada, atau mengedit menu yang telah dibuat sebelumnya. Langkah ini memungkinkan pengguna untuk mengatur dan mengelola struktur navigasi aplikasi.

### Langkah 9: Membuat Modul

1. Pilih Tabel

* Di tab **Select Table**, klik tombol **Reload Table** untuk memuat tabel yang telah pengguna buat sebelumnya.
* Pilih salah satu tabel yang akan digunakan untuk membuat modul.

1. Opsi Validator

* Sesuaikan nama kelas untuk validasi pembuatan data baru.
* Sesuaikan nama kelas untuk validasi perubahan data.
* Centang **Update Validation Definition** jika definisi yang telah ada untuk kelas tersebut akan ditimpa sesuai dengan aturan yang baru.

1. Opsi Modul

* Di bagian **Module**, pilih nama file, kode modul, nama modul, dan target tempat modul akan dibuat.
* Jika path target belum ada, klik tombol **Manage** dan buat path baru.
* Atur ikon untuk modul di tampilan menu. Ikon ini adalah kelas dari *Font Awesome*. Sesuaikan dengan versi yang digunakan.
* Centang opsi **Update Entity** jika pengguna ingin menimpa file entitas yang telah dibuat sebelumnya.
* Opsi **Load Saved Module** memungkinkan pengguna untuk memuat konfigurasi modul yang telah dibuat sebelumnya, sehingga pengguna tidak perlu mengisi ulang detail secara manual.
* Jika pengguna telah membuat menu, pilih tempat modul akan ditempatkan dalam menu tersebut.

1. Muat Kolom

* Klik tombol **Load Column**. Ini akan membawa pengguna ke tab **Generate Module**, di mana pengguna dapat mengatur opsi tambahan.

Dengan mengikuti langkah-langkah ini, pengguna dapat membuat dan mengelola modul dalam **MagicAppBuilder** dengan lebih efisien.

### Langkah 10: Penjelasan Tab Generate Module

Tab **Generate Module** adalah tab utama dalam pembuatan setiap modul dan entias. Tab ini berisi tabel dengan beberapa kolom. Penjelasan untuk masing-masing kolom adalah sebagai berikut:

1. Kolom dalam Tabel

* **Field**: Sesuai dengan nama kolom dari tabel yang dipilih.
* **Caption**: Versi kapitalisasi judul dari nama kolom dalam tabel yang dipilih.
* **I**: Centang untuk mengaktifkan fungsionalitas Insert.
* **U**: Centang untuk mengaktifkan fungsionalitas Update.
* **D**: Centang untuk mengaktifkan fungsionalitas Detail.
* **L**: Centang untuk mengaktifkan fungsionalitas List.
* **E**: Centang untuk mengaktifkan fungsionalitas Export.
* **K**: Centang untuk menandai kolom sebagai Primary Key.
* **R**: Centang untuk menandai kolom sebagai Required (Wajib).

1. Bagian Input

Di bagian Input, tersedia opsi berikut:

* **TE**: Menandakan input sebagai bidang teks (misalnya, teks, angka, tanggal, datetime-local, waktu, telepon, email, URL, warna, dll.).
* **TA**: Menandakan input sebagai textarea.
* **CB**: Menandakan input sebagai checkbox.
* **SL**: Menandakan input sebagai bidang pilihan (dropdown).
* **MI**: Menandakan input memiliki banyak nilai.

1. Bagian Filter

Di bagian Filter, tersedia opsi berikut:

* **TE**: Menandakan input sebagai bidang teks (misalnya, teks, angka, tanggal, datetime-local, waktu, telepon, email, URL, dll.).
* **SL**: Menandakan input sebagai bidang pilihan (dropdown).
* **MI**: Menandakan input memiliki banyak nilai.

1. Tipe Data

Isian ini akan menentukan tipe data dari bidang teks input.

1. Format Data

Isian ini akan menentukan format data pada bagian **Detail** dan **List**.

1. Tipe Filter

Filter yang akan diterapkan oleh aplikasi saat menerima input dari pengguna.

1. Validasi

Aturan yang akan digunakan digunakan untuk memvalidasi input dari pengguna. Sebuah input dapat divalidasi dengan satu aturan atau lebih. Aturan ini dapat diterapkan pada:

* + Pembuatan data baru saja
  + Perubahan data saja
  + Baik pembuatan data baru maupun perubahan data

#### **Penjelasan Kolom SL dalam Input dan Filter**

Kolom SL dalam bagian Input dan Filter digunakan untuk menghubungkan kolom dari suatu entitas atau tabel dengan data dari sumber lain. Data ini bisa berasal dari database lain atau nilai yang ditentukan pengguna. Biasanya digunakan untuk menghubungkan kolom dengan data dari tabel lain.

**Contoh**

Tabel lagu memiliki kolom berikut:

* song\_id
* name
* artist\_id
* recording\_date

Tabel artis memiliki kolom berikut:

* artist\_id
* name
* phone

Saat menampilkan data lagu, aplikasi akan menampilkan nama artis daripada kode artis. Untuk melakukannya, aplikasi akan melakukan join antara tabel lagu dan artis.

#### **Sumber untuk SL dalam Input dan Filter**

Ketika pengguna memilih SL, tombol **Source** akan muncul yang harus dikonfigurasi oleh pengguna.

**Konfigurasi Sumber**

1. **Entity**: Jika pengguna memilih **Entity**, MagicAppBuilder akan menampilkan formulir dengan bidang berikut:

**Bagian Entity**

Bagian Entity adalah bagian dasar yang wajib diis. Di bagian ini terdapat beberapa isian sebagai berikut:

* + **Entity Name**: Nama entitas yang akan dibuat (secara default, ini adalah versi Pascal-case dari nama tabel). Disarankan menambahkan akhiran "Min".
  + **Table Name**: Nama tabel sumber.
  + **Primary Key**: Nama primary key dalam tabel sumber.
  + **Value Column**: Nama kolom yang digunakan sebagai label untuk opsi dropdown filter di tampilan List dan dropdown di tampilan Create dan Update.
  + **Reference Object Name**: Nama properti yang digunakan untuk join dalam entitas.
  + **Reference Property Name**: Nama properti dari entitas referensi yang akan muncul di tampilan List dan Detail.

**Bagian Option Node**

Bagian ini digunakan untuk memformat tampilan dropdown. Pengguna dapat mengatur tampilan pada dropdown dengan cara lain alih-alih tampilan standard. Jika pengguna ingin mengambil data dari entitas referensi, pastikan pengguna tidak menggunakan entitas yang minimal.

Sebagai contoh:

Pengguna ingin menampilkan dropdown album dengan menampilkan nama album, nama produser, dan jumlah lagu dalam album. Pengguna dapat menggunakan format sebagai berikut:

“%s - %s (%d)”, name, producer.name, numberOfSong

**Bagian Specification**

Bagian Specification digunakan untuk mengatur filter pada dropdown. Aplikasi hanya akan menampilkan data sesuai dengan spesifikasi yang ditentukan.

Secara default, spesifikasi diset dengan active = true dan draft = false. Nama property ini sesuai dengan pemetaan kolom yang direservasi. Pengguna dapat menambahkan spesifikasi dengan cara membuat baris baru. Jika pengguna ingin menggunakan properti dari entitas referensi, gunakan tanda titik sebagai pemisah antara nama properti entitas dropdown dengan nama properti entitas referensi. Hubungan entitas ini ditentukan di dalam deklarasi entitas alih-alih kunci asing di database.

A screenshot of a computer

AI-generated content may be incorrect.

**Bagian Sortable**

Bagian Sortable digunakan untuk mengurutkan data yang ditampilkan di dropdown. Secara default, sortable diset dengan sortOrder = ASC dan name = ASC. Nama property ini sesuai dengan pemetaan kolom yang direservasi. Pengguna dapat menambahkan sortable dengan cara membuat baris baru. Jika diperlukan untuk menggunakan properti dari entitas referensi, gunakan tanda titik. Hubungan entitas ini ditentukan di dalam deklarasi entitas alih-alih kunci asing di database.

A screenshot of a computer

AI-generated content may be incorrect.

**Bagian Grouping**

Bagian Grouping digunakan untuk mengelompokkan pilihan pada dropdown. Setelah dikelompokkan, urutan tampilan mungkin akan berubah karena harus mengikuti kelompoknya.

Pengelompokan dapat dilakukan dengan dua cara yaitu dengan menggunakan Entity sebagai

Sumber dan Map sebagai sumber.

1. Pengeleompokan dengan Entity  
   Pengeleompokan dengan entity atau entitas dilakukan dengan cara menggunakan entitas sebagai sumber.

Untuk mengisi formular pengelompokan dropdown, lakukan dengan cara berikut:

**Value** diisi dengan nama properti dari entitas referensi yang digunakan sebagai referensi dari entitas dropdown.

**Label** diisi dengan nama properti dari entitas referensi yang akan ditampilkan sebagai label dari kelompok dropdown.

**Source** diisi dengan nama property dari entitas dropdown yang merupakan objek dari entitas referensi.

1. Pengelompokan dengan Map

Pengelompokan dengan Map atau peta yang dibuat.

Untuk mengisi formular pengelompokan dropdown, lakukan dengan cara berikut:

**Value** diisi dengan nama properti dari entitas dropdown yang akan digunakan sebagai parameter pengelompokan.

**Label** dapat dikosongkan.

**Reference** diisi dengan pasangan value dan label. Jika nilai property dari entitas dropdown yang dimasukkan ke Value dalam pengelompokan sama dengan nilai pada Value di referensi, maka nilai pada Label akan digunakan sebagai label kelompok. Dropdown akan dikelompokkan sesuai dengan nilai property yang diisikan ke Value.

**Bagian Additional Output**

Bagian Additional Ouput digunakan untuk membuat atribul tambahan di dropdown. Atribut ini dapat digunakan oleh JavaScript sebagai informasi dari dropdown yang dipilih. Dengan atribut tambahan ini, aplikasi tidak perlu meminta data ke server saat pengguna mengubah pilihan dropdown.

Aplikasi akan menambahkan atribut pada tag <option> pada dropdown dengan *kebab case* yang diawali dengan “data-”.

Sebagai contoh

Jika pada additional data dimasukkan property “numberOfSong”, maka tag <option> akan menjadi sebagai berikut:

<option value=”1234” data-number-of-song=”10”>Album Name</option>

**Column** diisi dengan nama kolom yang akan dijadikan atribut tambahan. Misalnya saat memilih dropdown album dan aplikasi menghendaki menampilkan jumlah lagu pada album tersebut, pengguna dapat memasukkan property numberOfSong pada Column. Pengguna juga dapat memasukkan properti dari entitas referensi. Jika pengguna ingin menggunakan properti dari entitas referensi, gunakan tanda titik sebagai pemisah antara nama properti entitas dropdown dengan nama properti entitas referensi. Hubungan entitas ini ditentukan di dalam deklarasi entitas alih-alih kunci asing di database.

1. **Map**: Jika pengguna memilih **Map**, MagicAppBuilder akan menampilkan formulir dengan bidang berikut:
   * **Value**: Nilai dari opsi.
   * **Label**: Label untuk opsi.
   * **Group**: Label untuk grup opsi.

Pengguna juga dapat menambahkan informasi tambahan di Additional Data. Aplikasi akan menambahkan atribut pada tag <option> pada dropdown dengan *kebab case* yang diawali dengan “data-”.

Berikut adalah penjelasan tentang formulir ini:

* Nilai dari kolom Value akan dimasukkan ke atribut value.
* Nilai dari kolom Label akan menjadi teks di dalam tag <option>.
* Nilai dari kolom Group akan menjadi nilai pada atribut label pada tag <optgroup> yang akan melingkupi tag <option>.
* Nilai pada Additional Data akan dimasukkan sebagai atribut dari tag <option> diawali dengan “data-” dan ditulis dengan *kebab case*. Pengguna dapat menambahkan Additional Data berapapun.

1. **Yes/No, True/False, 1/0**: Untuk opsi ini, pengguna tidak perlu mengisi apa pun.

#### **Penjelasan Tentang Data Type**

Data Type akan menentukan kontrol input, tipe data input dan tampilan output.

| **Nilai** | **Kode HTML** | **Deskripsi Singkat** | **Output** |
| --- | --- | --- | --- |
| text | <input type='text'> | Teks | Teks |
| email | <input type='email'> | Alamat email | Teks |
| url | <input type='url'> | URL | Teks |
| tel | <input type='tel'> | Nomor telepon | Teks |
| password | <input type='password'> | Kata sandi | Tidak ada |
| int | <input type='number'> | Bilangan bulat | Teks\* |
| float | <input type='number' step='any'> | Bilangan desimal | Teks\* |
| date | <input type='date'> | Tanggal | Teks\* |
| time | <input type='time'> | Waktu | Teks\* |
| datetime-local | <input type='datetime-local'> | Tanggal dan waktu | Teks\* |
| month | <input type='month'> | Bulan | Teks |
| week | <input type='week'> | Minggu | Teks |
| color | <input type='color'> | Pemilih warna | Teks |
| file | <input type='file'> | Unggah file | Link |
| image | <input type='file'> | Unggah gambar | Gambar |
| audio | <input type='file'> | Unggah audio | Audio |
| video | <input type='file'> | Unggah video | Video |

\*) Output dapat diformat sesuai dengan format yang diberikan pada Format Data.

#### **Penjelasan Tentang Format Data Output**

Tombol **Output Data Format** akan muncul jika pengguna memilih salah satu atau lebih dari pilihat **Detail** atau **List** dan elemen input berupa **Text** dan **Select**. Jika tombol ini muncul, pengguna dapat mengklik dan MagicAppBuilder akan menampilkan dialog seperti gambar berikut:

Sebuah gambar berisi teks, elektronik, cuplikan layar, tampilan

Konten yang dihasilkan AI mungkin salah.

Dialog terdiri dari 3 tab yaitu sebagai berikut:

* 1. Tab String Format
  2. Tab Date Format
  3. Tab Number Format

Setiap halaman pada ketiga tab dilengkapi dengan tombol Save dan Load untuk menyimpan format dan menggunakannya kembali di masa mendatang. Format ini akan disimpan di profil pengguna sesuai dengan aplikasi yang sedang digunakan. Dengan demikian, pada aplikasi yang sama, dua orang pengguna dapat menyimpan format yang berbeda.

Tab String Format digunakan untuk menformat data menggunakan pemformat %. Misalnya %s, %d, %x, %o, %f, dan sebagainya. Format data ini sama dengan format yang digunakan oleh PHP dalam fungsi **printf**, **sprintf** dan **vprintf**.

Sebuah gambar berisi teks, elektronik, cuplikan layar, tampilan

Konten yang dihasilkan AI mungkin salah.

Tab Date Format digunakan untuk memformat waktu. Format data ini sama dengan format yang digunakan oleh PHP dalam fungsi **date** dan **DateTime::format()**.

Sebuah gambar berisi teks, cuplikan layar, Font, tampilan

Konten yang dihasilkan AI mungkin salah.

Tab Number Format digunakan untuk memformat bilangan mengambang yaitu **float**. Parameter yang dapat dimasukkan adalah **Decimal**, **Decimal Separator** dan **Thousands Separator**. **Decimal** wajib diisi.

Sebuah gambar berisi teks, cuplikan layar, nomor, Font

Konten yang dihasilkan AI mungkin salah.

#### **Penjelasan Tentang Filter Type**

Filter Type akan menentukan bagaimana input akan difilter sebelum diproses lebih lanjut. Penting untuk memahami bagaimana filter bekerja.

| **Nilai** | **Deskripsi Singkat** |
| --- | --- |
| DEFAULT | Default |
| BOOL | Boolean |
| NUMBER\_INT | Bilangan integer |
| NUMBER\_UINT | Bilangan integer tak bertanda |
| NUMBER\_OCTAL | Bilangan oktal |
| NUMBER\_HEXADECIMAL | Bilangan heksadesimal |
| NUMBER\_FLOAT | Bilangan floating-point |
| STRING | String |
| STRING\_INLINE | String (inline) |
| NO\_DOUBLE\_SPACE | Tanpa spasi ganda |
| STRIPPED | Menghapus spasi awal/akhir |
| SPECIAL\_CHARS | Karakter spesial html |
| ALPHA | Karakter alfabet |
| ALPHANUMERIC | Karakter alfanumerik |
| ALPHANUMERICPUNC | Alfanumerik dan tanda baca |
| STRING\_BASE64 | String base64 |
| EMAIL | Alamat email |
| URL | URL |
| IP | Alamat IP |
| ENCODED | URL-encoded |
| COLOR | Nilai warna |
| MAGIC\_QUOTES | Magic quotes |
| PASSWORD | Kata sandi |

Secara default, MagicAppBuilder akan menggunakan filter SPECIAL\_CHARS untuk tipe data string. Filter akan disesuaikan dengan tipe data pada kolom tabel. Pengguna dapat mengubahnya sesuai dengan kebutuhan.

#### **Penjelasan Tentang Validasi Data Input**

Validasi data input akan diproses oleh server setelah data diterima namun belum dimasukkan ke database. Validasi ini menjamin integritas dan kualitas data bahkan jika pengguna berusaha memasukkan data secara paksa dengan menggunakan aplikasi non peramban maupun modifikasi prilaku web baik dengan *add-on* peramban maupun manipulasi elemen.

Adapun validasi input data yang dapat digunakan adalah sebagai berikut:

**Kehadiran & Nullability**

* **@Required**: Memastikan nilai properti tidak null.
* **@NotEmpty**: Memeriksa apakah sebuah string tidak kosong ("") atau sebuah *array* tidak kosong.
* **@NotBlank**: Memvalidasi bahwa sebuah string tidak kosong dan bukan hanya karakter spasi.

**Rentang Nilai & Ukuran**

* **@Min**: Memastikan nilai properti numerik lebih besar dari atau sama dengan nilai minimum yang ditentukan (value).
* **@Max**: Memastikan nilai properti numerik kurang dari atau sama dengan nilai maksimum yang ditentukan (value).
* **@DecimalMin**: Memvalidasi properti numerik (bisa *float*/string) lebih besar dari atau sama dengan nilai desimal tertentu (value).
* **@DecimalMax**: Memvalidasi properti numerik (bisa *float*/string) kurang dari atau sama dengan nilai desimal tertentu (value).
* **@Range**: Memvalidasi bahwa nilai properti numerik berada dalam rentang inklusif (min dan max).
* **@Size**: Memverifikasi bahwa panjang string atau jumlah elemen *array* berada dalam rentang yang ditentukan (min dan max).
* **@Length**: Mirip dengan @Size, khusus untuk panjang string dalam rentang (min dan max).
* **@MaxLength**: Hampir sama dengan @Length, namun hanya menentukan batas atas. Cocok untuk membatasi panjang karakter pada input dengan tipe kolom seperti VARCHAR (value). Validasi ini secara default ada di kelas entitas. Jika pengguna menggunakan fitur validasi tetapi tidak menentukan aturan validasi secara ekspilit di kolom **Validation**, aplikasi akan memvalidasi entitas menggunakan anotasi validasi yang ada pada entitas itu sendiri alih-alih menggunakan aturan yang ada di kelas referensi.
* **@Digits**: Memeriksa bahwa properti numerik memiliki paling banyak digit *integer* (integer) dan digit pecahan (fraction) tertentu.

**Tanda Numerik**

* **@Positive**: Memastikan nilai numerik positif (> 0).
* **@PositiveOrZero**: Memastikan nilai numerik positif atau nol (>= 0).
* **@Negative**: Memastikan nilai numerik negatif (< 0).
* **@NegativeOrZero**: Memastikan nilai numerik negatif atau nol (<= 0).

**Pola & Format**

* **@Pattern**: Memvalidasi properti string terhadap ekspresi reguler yang ditentukan (regexp).
* **@Email**: Memeriksa apakah properti string adalah alamat email yang terbentuk dengan baik.
* **@Url**: Memastikan sebuah string adalah URL yang valid.
* **@Ip**: Memastikan sebuah string adalah alamat IP yang valid.
* **@DateFormat**: Memastikan sebuah string cocok dengan format tanggal tertentu (format).
* **@Phone**: Memastikan sebuah string adalah nomor telepon yang valid.
* **@NoHtml**: Memeriksa apakah properti string mengandung tag HTML.

**Tanggal & Waktu**

* **@Past**: Memastikan properti DateTimeInterface mewakili tanggal/waktu di masa lalu.
* **@Future**: Memastikan properti DateTimeInterface mewakili tanggal/waktu di masa depan.
* **@PastOrPresent**: Memastikan tanggal/waktu berada di masa lalu atau sekarang.
* **@FutureOrPresent**: Memastikan properti DateTimeInterface mewakili tanggal/waktu di masa depan atau sekarang.
* **@BeforeDate**: Memastikan sebuah tanggal sebelum tanggal yang ditentukan (date).
* **@AfterDate**: Memastikan sebuah tanggal setelah tanggal yang ditentukan (date).

**Boolean**

* **@AssertTrue**: Memastikan nilai properti boolean adalah true.

**Enum & Nilai yang Diizinkan**

* **@Enum**: Memastikan nilai properti string adalah salah satu dari sekumpulan nilai yang telah ditentukan sebelumnya (allowedValues), dengan opsi perbandingan sensitif huruf besar/kecil (caseSensitive).

**Konten & Struktur String**

* **@Alpha**: Memastikan sebuah string hanya mengandung karakter alfabet.
* **@AlphaNumeric**: Memastikan sebuah string hanya mengandung karakter alfanumerik.
* **@StartsWith**: Memastikan sebuah string dimulai dengan awalan tertentu (prefix), dengan opsi sensitivitas huruf besar/kecil (caseSensitive).
* **@EndsWith**: Memastikan sebuah string diakhiri dengan akhiran tertentu (suffix), dengan opsi sensitivitas huruf besar/kecil (caseSensitive).
* **@Contains**: Memastikan sebuah string mengandung *substring* tertentu (substring), dengan opsi sensitivitas huruf besar/kecil (caseSensitive).

**Validasi Bersarang**

* **@Valid**: Secara rekursif memvalidasi *nested MagicObject* dan *MagicDto instances*.

Setiap validasi di atas dapat memiliki atribut pesan (message). Pesan ini akan ditampilkan ke pengguna apabila data yang dimasukkan tidak sesuai dengan aturan yang ditetapkan. Pengguna dapat menentukan pesan yang akan ditampilkan pada atribut ini. Jika ingin memasukkan nilai atribut lain dalam pesan, gunakan *placeholder* sesuai dengan nama atribut yang akan dimasukkan.

Sebagai contoh

@Length(min=3, max=100, message=”Panjang nama harus antara ${min} hingga ${max} karakter.”)

Jika atribut pesan (message) tidaka ada, kosong, atau hanya berisi spasi putih, maka aplikasi akan menampilkan pesan sesuai dengan *template*. *Template* ini dapat diterjemahkan ke berbagai bahasa yang digunakan oleh aplikasi. Template pesan validasi dapat diterjemahkan dari tab **Translate App**. Pilik bagian **Validation** di sisi kiri halaman kemudian pilih bahasa tujuan. Anda dapat menterjemahkan secara manual maupun menggunakan aplikasi penterjemah dari pihak ketiga. Sangat disarankan untuk memeriksa hasil terjemahan untuk menghindari kesalahan.

#### **Penjelasan Tombol di Bawah Tabel**

Di bawah tabel dalam tab Generate Module, terdapat tombol-tombol berikut:

1. **Data Filter**: Digunakan untuk memfilter data yang akan ditampilkan dalam modul, baik di tampilan daftar, detail, edit, hapus, approve, atau reject. Filter ini memastikan bahwa pengguna tidak dapat mengakses data yang tidak memiliki izin.
2. **Data Order**: Digunakan untuk menentukan urutan tampilan data dalam daftar. Pengguna masih dapat mengubah urutan dengan mengklik header kolom.
3. **Fitur Modul**: Bagian ini memungkinkan pengguna mengonfigurasi fitur tambahan untuk modul, seperti:
   * Aktivasi/Deaktivasi data.
   * Urutan manual.
   * Ekspor ke Excel atau CSV.
   * Penggunaan file sementara.
   * Pencatatan aktivitas pengguna.
   * Persetujuan (approval).
   * Trash (data yang dihapus disimpan dalam tabel terpisah).
   * Validasi input data. Validasi ini akan menerapkan aturan yang ditetapkan pada masing-masing kolom data. Meskipun aturan telah ditetapkan, jika fitur ini tidak dipilih, MagicObject tidak akan membuat validasi input.
   * Back end saja, yaitu hanya membuat kode program untuk pemrosesan di sisi server tanpa membuat sumber daya untuk ditampilkan ke pengguna.
   * Dukungan AJAX. Dengan fitur ini, saat pengguna memilih halaman pada daftar data, aplikasi monolit hanya akan memuat data alih-alih memuat semua halaman. Dengan demikian prosesnya akan lebih cepat dan lebih efisien dalam penggunaan sumber daya.
   * Penggunaan subquery untuk meningkatkan performa pada dataset besar

Beberapa pilihan hanya tersedia pada kondisi tertentu.

Pengguna dapat menyimpan konfigurasi fitur modul untuk digunakan pada modul berikutnya. Untuk menyimpan konfigurasi, pilih tombol **Save**. Untuk memuat konfigurasi, pilih tombol **Load**. Untuk mereset konfigurasi, pilih tombol **Clear**.

Konfigurasi akan disimpan untuk aplikasi yang sedang aktif dan hanya berlaku bagi pengguna yang bersangkutan. Pengguna lain dapat menyimpan konfigurasi yang berbeda di aplikasi yang sama.

### Langkah 11: Mengirimkan Formulir untuk Membuat Modul dan Entitas

1. **Generate Modul**: Klik tombol **Generate Module** untuk mengonfirmasi pembuatan modul.
2. **Melihat Modul yang Dibuat**: Modul dapat dilihat dalam tab **Edit Module**.
3. **Melihat Entitas yang Dibuat**: Entitas dapat dilihat dalam tab **Edit Entity**.
4. **Melihat Hubungan Entitas**: Hubungan antar entitas dapat dilihat dalam tab **ERD** (Entity Relationship Diagram).

### Langkah 12: Memperbarui Struktur Database

Setelah membuat modul dan entitas, mungkin perlu memperbarui struktur database. Jika menggunakan fitur seperti aktivasi, deaktivasi, atau urutan, tetapi kolom yang diperlukan belum ada, MagicAppBuilder akan menambahkan beberapa kolom.

1. Buka tab **Query**, centang **Merge queries by table** dan **Select all**.
2. MagicAppBuilder akan menghasilkan query untuk memperbarui struktur database.
3. Jika tidak ada query yang ditampilkan, berarti struktur database sudah sesuai.
4. Jika ada query, jalankan dengan mengklik **Execute Query**.

Untuk membuat struktur database baru dari awal, centang **Create new** agar MagicAppBuilder menghasilkan query untuk membuat seluruh tabel.

### Langkah 13: Membuat Lokalisasi

**Lokalisasi Aplikasi**

Pengguna dapat menerjemahkan modul dan entitas ke dalam bahasa lain sesuai kebutuhan.

* **Terjemahan Modul**: Semua label dan tombol dalam modul dapat diterjemahkan di tab **Translate Module**.
* **Terjemahan Entitas**: Semua label dan tombol dalam entitas dapat diterjemahkan di tab **Translate Entity**.
* **Terjemahan Grup Menu, Menu dan Validasi**: Semua label dari grup menu, menu dan validasi dapat diterjemahkan di tab **Translate App**.

**Antarmuka**

Tampilan dibagi menjadi dua bagian: kiri menunjukkan label asli, kanan menunjukkan label dalam bahasa target. Pengguna dapat mengubah teks yang berada di sebelah kanan. Usahakan terjemahkan secara manual alih-alih menggunakan mesin penterjemah. Terjemahkan sebuah kata secara konsisten selama memiliki konteks yang sama.

### Langkah 14: Opsi Aplikasi

Opsi aplikasi atau Application Option digunakan untuk melakukan konfigurasi akses aplikasi. Secara default, aplikasi akan menampilkan menu dari file /inc.cfg/menu.yml yang dibuat secara otomatis oleh MagicAppBuilder. File ini dapat diubah oleh pengguna. Aplikasi belum memiliki pengguna yang terdaftar di database. Pengguna tanpa melakukan “Log In” ke sistem dapat mengakses semua modul tanpa memerlukan ijin. Aplikasi hanya diapat diakses dari localhost. Application Option memungkinkan pengguna untuk mengatur semua hal di atas.

Saat pengguna memilih tombol “Option” pada kartu aplikasi, Application Option muncul sebagai dialog yang memiliki 3 akordion yaitu sebagai berikut:

1. Application Menu
2. Application User
3. Application Mode

**Application Menu**

Pada akordion ini, pengguna dapat mengimpor semua menu dari file /inc.cfg/menu.yml ke database sebagai module group dan module. Untuk mengimpor menu, pilih tombol **Import Menu**. MagicAppBuilder akan mengimpor semua menu dari file Yaml ke database. Menu dan submenu yang berhasil diimpor akan ditandai dengan tanda centang di sebelah kanan menu.

Jika pengguna memilih **Multi Level Menu** pada pengaturan aplikasi, MagicAppBuilder akan secara otomatis membuat modul yang akan menjadi induk dari modul-modul yang dibuat. Data diambil dari **Module Group** yang terkait. Modul induk ini dibuat untuk memastikan bahwa menu yang dibuat tidak menjadi menu level 1 yang menyebabkan tampilan menjadi tidak rapi. Selain itu, MagicAppBuilder juga akan menentukan hak akses ke modul induk minimal sama dengan hak akses modul yang berada di bawahnya.

Impor menu dapat dilakukan berkali-kali setiap ada pembaruan di file /inc.cfg/menu.yml. Pengguna juga dapat menghapus data yang ada pada module group dan module sebelum kembali melakukan impor menu.

**Application User**

Pada akordion ini, pengguna dapat membuat akun pengguna di database aplikasi. Akun pengguna akan memiliki properti sebagai berikut:

Nama : Super User

Username : superuser

Password : superuser

Untuk membuat akun pengguna, pilih tombol “Create User”. Jika akun tersebut belum ada, MagicAppBuilder akan membuatnya dengan level pengguna “superuser”. Jika level pengguna “superuser” belum ada, MagicAppBuilder akan membuatnya terlebih dahulu sebelum akun pengguna dibuat.

Pengguna dapat melakukan reset password dengan mencentang akun pengguna yang tampil di akordion ini lalu memilih tombol “Reset Password”. Password akan diset sama dengan username dari akun pengguna tersebut.

Selain dapat mereset password, pengguna juga dapat memberi akses “Superuser” pada akun yang dipilih. Perhatikan untuk lebih berhati-hati dalam memberikan akses “Superuser” kepada akun yang dipilih. Akses “Superuser” memungkinkan pengguna untuk mengakses semua fitur pada semua modul tanpa batas.

Untuk memberikan akses “Superuser”, centang akun pengguna lalu pilih tombol “Set Superuser Role”. MagicAppBuilder akan memberikan akses “Superuser” pada semua modul yang ada di database aplikasi.

**Application Mode**

Pada akordion ini, terdapat 3 pilihan yaitu sebagai berikut:

1. **Development Mode**

Development Mode akan menentukan sumber data untuk menu aplikasi. Jika dicentang sevagai “Yes”, aplikasi akan menampilkan menu sesuai dengan data pada file /inc.cfg/menu.yml. Jika tidak dicentang, aplikasi akan menampilkan menu dari database yaitu dari module group dan module.

1. **Bypass Role**

Bypass Role akan menentukan apakah pengguna harus melakukan “Log In” ke aplikasi untuk mengakses semua modul yang ada. Jika Bypass Role dicentang, pengguna dapat mengakses aplikasi tanpa memalui proses “Log In” dan tidak ada aturan untuk membatasi akses pengguna. Jika tidak dicentang, aplikasi akan meminta pengguna untuk melakukan “Log In” ke aplikasi sesuai dengan data akun pengguna. Selanjutnya aplikasi akan memeriksa apakah pengguna tersebut memiliki hak untuk mengakses modul tertentu.

1. **Access Localhost Only**

Access Localhost Only mencegah aplikasi diakses dari perangkat lain. Pilihan ini sangat penting saat pengguna memilih opsi “Bypass Role”. Jika Bypass Role dicentang, Access Localhost Only harus dicentang untuk mencegah akses aplikasi dari perangkat lain. Pengguna mungkin perlu membuka akses dari perangkat lain dalam waktu yang singkat jika diperlukan namun harus segera ditutup kembali jika sudah tidak diperlukan lagi.

Apa bahayanya jika Bypass Role dicentang namun Access Localhost Only tidak dicentang?

Akses ke aplikasi tidak dapat mengubah source code aplikasi namun dapat menghapus data dari aplikasi termasuk modul aplikasi. Hal ini tentu saja akan mengganggu proses pembanggunan aplikasi.

### Langkah 15: Menambahkan Favicon ke Aplikasi

Buka tab **Apps**, klik tombol **Icon** pada aplikasi, lalu unggah gambar persegi dalam format PNG (minimal 512x125 piksel). MagicAppBuilder akan menghasilkan beberapa ikon dengan nama seperti:

* favicon-16x16.png
* apple-icon-57x57.png
* android-icon-192x192.png
* favicon.ico (berisi gambar 16x16, 32x32, dan 48x48 piksel)

Selain membuat file-file di atas, MagicAppBuilder juga akan menghasilkan file **manifest.json** yang berisi informasi terkait ikon aplikasi.

{

"name": "Application Name",

"short\_name": "AppName",

"icons": [

{

"src": "apple-icon-57x57.png",

"sizes": "57x57",

"type": "image\/png"

},

{

"src": "apple-icon-60x60.png",

"sizes": "60x60",

"type": "image\/png"

},

{

"src": "android-icon-192x192.png",

"sizes": "192x192",

"type": "image\/png"

}

],

"start\_url": "\/",

"display": "standalone"

}

### Langkah 16: Melakukan Kustomisasi Aplikasi

**MagicAppBuilder** memberikan kebebasan yang seluas-luasnya kepada pengguna untuk melakukan kustomisasi aplikasi baik logika modul hingga tampilan. Kustomisasi tidak bisa dilakukan sepenuhnya melalui fitur file manager yang ada di MagicAppBuilder. Sangat disarankan untuk menggunakan *Integrated Development Environment* (IDE) dari pihak ketiga untuk melakukan kustomisasi aplikasi.

**MagicAppBuilder** versi gratis dibekali dengan 2 tema standard. Pada kedua tema ini, menu ditampilkan dalam 2 tingkat. Pengguna mungkin membutuhkan menu dengan tingkatan yang tidak terbatas. Dibutuhkan logika baru untuk menampilkan menu secara bertingkat. Pengguna dapat membuatnya di tema baru.

## Langkah 17: Deploymen Aplikasi

Langkah terakhir yang harus dilakukan adalah melakukan deployment aplikasi ke server. Pengguna dapat menggunakan semua server yang mendukung bahasa pemrograman PHP baik versi 5, 7 maupun 8. Untuk PHP versi 5, diwajibkan menggunakan versi 5.6 atau yang lebih tinggi.

Langkah-langkah deployment adalah sebagai berikut:

1. Persiapan Server Web (Apache/Nginx) Pastikan server web Anda, seperti Apache atau Nginx, sudah terinstal dan terkonfigurasi dengan benar untuk melayani aplikasi PHP. Ini termasuk mengaktifkan modul PHP yang diperlukan, mengatur virtual host atau konfigurasi server blok yang sesuai untuk domain atau subdirektori aplikasi Anda, dan memastikan hak akses direktori sudah tepat.
2. Persiapan Lingkungan PHP Instal dan konfigurasikan PHP pada server Anda. Pastikan versi PHP yang terinstal sesuai dengan kebutuhan aplikasi (PHP 5.6+, PHP 7.x, atau PHP 8.x). Verifikasi bahwa semua ekstensi PHP yang diperlukan oleh MagicAppBuilder atau library pihak ketiga sudah diaktifkan (misalnya, pdo\_mysql, mbstring, gd, curl, redis jika digunakan). Periksa juga pengaturan php.ini seperti memory\_limit, max\_execution\_time, dan upload\_max\_filesize agar sesuai dengan kebutuhan aplikasi.
3. Persiapan Server Database Siapkan server database Anda. Sangat tidak disarankan menggunakan SQLite untuk lingkungan produksi karena keterbatasannya dalam skalabilitas dan kinerja untuk aplikasi multi-pengguna. Beberapa pilihan Database Management System (DBMS) yang direkomendasikan adalah sebagai berikut:

* MySQL
* MariaDB
* PostgreSQL

Pilih salah satu yang paling sesuai dengan kebutuhan dan keahlian tim Anda, lalu pastikan DBMS tersebut sudah terinstal dan berjalan dengan baik di server.

1. Pembuatan Database dan Kredensial Setelah server database dipersiapkan, langkah selanjutnya adalah membuat database baru khusus untuk aplikasi Anda. Beri nama database yang relevan dan buatlah pengguna database (user) baru yang memiliki hak akses spesifik hanya untuk database tersebut. Pastikan Anda mengatur kata sandi (password) yang kuat untuk pengguna database ini. Penggunaan kredensial root atau superuser database untuk aplikasi sangat tidak disarankan karena alasan keamanan.
2. Buat struktur database sesuai dengan DBMS yang digunakan.
3. Beberapa data dasar seperti grup modul, modul, level pengguna, administrator utama (superuser) dan pengaturan hak akses modul yang telah dibuat di lingkungan pengembangan dapat diekspor ke lingkungan produksi. MagicAppBuilder menyediakan fitur ekspor pada subsistem Database Explorer.
4. Setelah struktur database di lingkungan produksi selesai dibuat, salin semua kode sumber aplikasi ke lingkungan produksi.
5. Atur konfigurasi aplikasi melalui file /inc.cfg/application.yml.
6. Jika data session akan disimpan di Redis, persiapkan server Redis di lingkungan produksi dan pastikan keamanannya.
7. Pastikan konfigurasi session dan cookie telah memenuhi standar keamanan yang baik.
8. Uji coba aplikasi dengan cara memasukkan data baik data percobaan maupun data nyata.
9. Periksa log server dan pastikan tidak terdapat kesalahan. Pengguna pemula mungkin perlu berkonsultasi untuk memecahkan permasalahan yang mungkin timbul baik dari kesalahan kode, kesalahan struktur database, maupun kesalahan konfigurasi aplikasi.

## Siklus Hidup Pengembangan Perangkat Lunak

### Pengembangan Fitur Aplikasi

Setelah aplikasi berjalan di lingkungan produksi, pengguna mungkin memerlukan penambahan atau perubahan fitur. Perubahan ini sepenuhnya dilakukan di lingkunan pengembangan. Pengguna dapat menambahkan baik tabel maupun kolom tabel dari database yang digunakan di lingkungan pengembangan.

Pengguna memerlukan salinan struktur database dari lingkungan produksi sebagai referensi saat akan membuat skrip perubahan struktur database. Jalankan skrip tersebut saat melakukan depoymen fitur baru. Struktur database bayangan dari lungkungan produksi harus selalu diperbarui setiap akan membuat skrip perubahan data.

Dalam pengembangan aplikasi, hal yang paling dilarang adalah menghapus tabel, mengubah nama tabel, menghapus kolom, mengubah nama kolom, mengubah tipe data kolom yang tidak sesuai, atau mengurangi kapasitas data kolom.

Sebagai contoh: di tabel customer terdapat address dengan tipe VARCHAR(250). Artinya panjang maksimum teks yang dapat dimasukkan adalah 250 karakter. Kolom ini tidak boleh diubah menjadi VARCHAR(200) misalnya. Atau tipe data yang sebelumnya string diubah menjadi tipe data numerik.

### Optimasi Database

MagicAppBuilder hanya menggunakan *primary key* atau kunci utama dalam mengendalikan data. Optimasi database dengan cara menambahkan indeks mungkin diperlukan terutama saat data berkembang menjadi lebih besar. Seorang Database Administrator atau DBA mungkin diperlukan dalam hal ini.

Optimasi database setidaknya memiliki 2 manfaat yaitu:

* + - * 1. Mempercepat Proses

Optimasi database akan mempercepat proses baik menulis maupun membaca, dan menghapus data di database.

* + - * 1. Penghematan Sumber Daya

Optimasi database akan menghemat sumber daya sehingga pemilik aplikasi tidak perlu menyediakan perangkat dengan spesifikasi yang tinggi untuk mendukung jalannya aplikasi. Dengan demikian, biaya yang diperlukan akan menjadi lebih murah.

### Scaling dan Sharding Database

Seiring pertumbuhan data dan jumlah pengguna, optimasi database saja mungkin tidak lagi mencukupi. Pada titik ini, database perlu di-*scale* untuk menangani beban yang lebih besar. Ada dua pendekatan utama:

1. **Pensakalaan Vertikal (Vertical Scaling)**: Ini berarti meningkatkan kapasitas satu server database tunggal, misalnya dengan menambahkan lebih banyak RAM, CPU, atau penyimpanan yang lebih cepat. Meskipun sederhana untuk diterapkan, pendekatan ini memiliki batasan fisik karena ada batas seberapa besar satu server dapat ditingkatkan.
2. **Pensakalaan Horizontal (Horizontal Scaling) / *Sharding***: Ini melibatkan distribusi data di banyak server database. Secara logika, sistem masih berfungsi sebagai satu database terpadu, namun data dan beban kerja tersebar di beberapa mesin fisik.

### Perbaikan Masalah (Debugging)

Ketika masalah muncul pada aplikasi yang sedang berjalan di lingkungan produksi, identifikasi akar masalah merupakan langkah krusial. Perbaikan bug harus dilakukan di lingkungan pengembangan, bukan langsung di produksi. Ini memastikan bahwa perubahan yang dilakukan telah melalui pengujian yang memadai dan tidak menimbulkan masalah baru. Setelah perbaikan selesai dan diuji, skrip perubahan kode akan dijalankan di lingkungan produksi.

### Pembaruan Sistem (Maintenance)

Pembaruan sistem mencakup berbagai aktivitas yang menjaga aplikasi tetap berjalan lancar dan aman. Ini bisa berupa penerapan patch keamanan, update komponen pihak ketiga, atau peningkatan versi sistem operasi server. Pembaruan ini perlu direncanakan dengan cermat dan sering kali memerlukan waktu henti (downtime) yang minim untuk aplikasi. Proses ini juga sebaiknya melalui lingkungan pengembangan dan staging terlebih dahulu sebelum diterapkan ke lingkungan produksi.

# BAB 5 – KESIMPULAN

*Low-code programming* merupakan pendekatan inovatif dalam pengembangan perangkat lunak yang menawarkan solusi cepat, efisien, dan mudah digunakan. Dengan metode ini, pengembang dapat membangun aplikasi tanpa perlu menulis banyak kode secara manual, sehingga memungkinkan proses pengembangan yang lebih cepat dan minim risiko kesalahan. Fokus utama dalam *low-code programming* adalah pada desain database, pembuatan form, serta konfigurasi modul aplikasi, yang semuanya dapat dilakukan melalui antarmuka visual tanpa harus memahami sintaks pemrograman yang kompleks.

**MagicAppBuilder** adalah salah satu aplikasi yang menerapkan konsep *low-code programming*. Aplikasi ini memungkinkan pengguna untuk membuat modul aplikasi berdasarkan struktur database yang telah dikonfigurasi. Dengan fitur-fitur seperti pembuatan dan pengelolaan entitas, pengaturan modul, serta visualisasi hubungan antar entitas, MagicAppBuilder menjadi solusi yang mempermudah proses pengembangan aplikasi tanpa harus menulis kode secara manual. Keunggulan ini menjadikannya pilihan yang ideal bagi bisnis atau individu yang ingin membangun aplikasi dengan cepat dan efisien.

Meskipun memiliki banyak keunggulan, *low-code programming* masih memiliki keterbatasan, terutama dalam hal kustomisasi dan skalabilitas. Beberapa aplikasi dengan logika bisnis yang kompleks mungkin tetap memerlukan penulisan kode tambahan untuk memenuhi kebutuhan spesifik. Namun, **MagicAppBuilder** tetap memberikan fleksibilitas dengan memungkinkan pengembang menyesuaikan modul atau entitas secara manual jika diperlukan.

Dengan perkembangan teknologi yang semakin pesat, *low-code programming*, termasuk MagicAppBuilder, berpotensi menjadi standar baru dalam industri perangkat lunak. Kemudahan penggunaannya membuka peluang bagi individu maupun bisnis untuk menciptakan aplikasi sendiri tanpa harus bergantung sepenuhnya pada tim pengembang profesional. Hal ini tidak hanya mempercepat proses digitalisasi, tetapi juga memungkinkan inovasi lebih luas dengan keterlibatan pengguna dari berbagai latar belakang dalam pengembangan perangkat lunak.

Secara keseluruhan, *low-code programming* dengan dukungan aplikasi seperti **MagicAppBuilder** memberikan alternatif yang menarik dalam pengembangan aplikasi, terutama bagi bisnis yang ingin menghemat waktu dan sumber daya tanpa mengorbankan fungsionalitas. Dengan terus berkembangnya teknologi dan semakin banyaknya alat yang mendukung pendekatan ini, *low-code programming* diprediksi akan semakin populer dan menjadi bagian integral dari ekosistem pengembangan perangkat lunak di masa depan.