



MAGICAPPBUILDER

PANDUAN PENGGUNA



2 FEBRUARI 2026

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, Februari 2026

Penulis

DAFTAR ISI

KATA PENGANTAR.....	1
DAFTAR ISI.....	2
BAB 1 – PENDAHULUAN	4
1.1 Definisi <i>Low-code Programming</i>	4
1.2 Keunggulan <i>Low-code Programming</i>	4
1.3 Keterbatasan <i>Low-code Programming</i>	5
BAB 2 – IMPLEMENTASI	6
2.1 Implementasi Pada Aplikasi Berbasis Web.....	6
2.2 MagicAppBuilder - Platform <i>Low-Code</i>	6
BAB 3 – MAGICAPPBUILDER.....	8
3.1 Fitur MagicAppBuilder.....	8
3.2 Subsitem MagicAppBuilder	8
3.2.1 Database Explorer	9
3.2.2 Entity Editor.....	9
3.2.3 Administration.....	10
3.3 File Manager	20
3.3.1 Fitur Utama	20
3.3.2 Menu Konteks Direktori.....	21
3.3.3 Menu Konteks Direktori Root.....	22
3.3.4 Menu Konteks File	23
3.3.5 Cara Menggunakan File Manager.....	24
BAB 4 – PANDUAN PENGGUNAAN	26
4.1 Versi MagicAppBuilder	26
4.2 Antarmuka Pengguna	26
4.3 Langkah-Langkah Penggunaan	30
4.1.1 Langkah 1: Instalasi Server	30
4.1.2 Langkah 2: Mengunduh dan Memasang MagicAppBuilder di Server	46
4.1.3 Langkah 3: Membuat Workspace	56
4.5.1 Langkah 4: Membuat Aplikasi.....	59
4.5.2 Langkah 5: Konfigurasi Aplikasi	62

4.5.3	Langkah 6: Mempersiapkan Struktur Database	71
4.5.4	Langkah 7: Membuat Entitas dengan Entity Editor	75
4.5.5	Langkah 8: Membuat Menu Aplikasi	86
4.5.6	Langkah 9: Membuat Modul	86
4.5.7	Langkah 10: Penjelasan Tab Generate Module	87
4.5.8	Langkah 11: Mengirimkan Formulir untuk Membuat Modul dan Entitas..	106
4.5.9	Langkah 12: Memperbarui Struktur Database	106
4.5.10	Langkah 13: Membuat Lokalisasi	107
4.5.11	Langkah 14: Opsi Aplikasi	107
4.5.12	Langkah 15: Melakukan Kustomisasi Aplikasi	112
4.5.13	Langkah 16: Deploymen Aplikasi.....	112
4.4	Impor Aplikasi.....	114
4.5	Ekspor Aplikasi	114
4.6	Aplikasi GraphQL	115
4.7	Siklus Hidup Pengembangan Perangkat Lunak	117
4.7.1	Pengembangan Fitur Aplikasi	117
4.1.1	Optimasi Database.....	118
4.1.2	Scaling dan Sharding Database.....	118
4.1.3	Perbaikan Masalah (Debugging)	118
4.1.4	Pembaruan Sistem (Maintenance)	119
4.2	Command Line Interface (CLI)	119
BAB 5 – KESIMPULAN		124

BAB 1 – PENDAHULUAN

1.1 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 programmer hanya perlu mendefinisikan komponen utama aplikasi, seperti:

- a. Database: Struktur dan relasi antara data yang akan digunakan dalam aplikasi.
- b. Form: Antarmuka yang digunakan pengguna untuk memasukkan atau melihat data.
- c. 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.

1.2 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.

2. Mengurangi Kesalahan Kode

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

3. 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.

4. 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.

1.3 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

2.1 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 pemrograman yang populer 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 tertentu 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.

2.2 MagicAppBuilder - Platform *Low-Code*

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 bakal nya 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

3.1 Fitur MagicAppBuilder

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

1. Membuat workspace.
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 deskripsi entitas dalam format Markdown.
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.

3.2 Subsistem MagicAppBuilder

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

3.2.1 Database Explorer

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

1. Memilih Database di sebuah server.
2. Menampilkan table di database terpilih.
3. Menampilkan struktur tabel.
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.
13. Mengekspor data ke format Microsoft Excel.

Database Explorer dapat menjalankan SQL, melakukan INSERT, SELECT dan UPDATE pada tabel. Database Explorer dirancang untuk menjelajahi database MySQL, PostgreSQL dan SQLite. Oleh karenanya tidak tersedia peralatan seperti ALTER TABLE. Jika pengguna terbiasa dengan aplikasi seperti phpMyAdmin, silakan buka phpMyAdmin melalui URL <http://localhost/phpMyAdmin/> setelah semua proses instalasi dilakukan.

3.2.2 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 file JSON.
9. Mengekspor entitas ke file JSON.
10. Mengimpor entitas dari file SQL dan file biner database SQLite.

11. Mengimpor entitas dari file spreadsheet (CSV, DBF, ODS, XLS dan XSLX).
12. Mengimpor entitas dari clipboard setelah pengguna menyalin data dari aplikasi Excel, Word dan web ke clipboard.
13. Mengimpor entitas dari dokumen Markdown yang disalin ke clipboard.
14. Mengekspor entitas ke dari file SQL.
15. Membuat diagram dari entitas.
16. Mengatur tampilan diagram.
17. Mengunduh diagram dalam format SVG.
18. Mengunduh diagram dalam format PNG.
19. Mengekspor diagram hubungan entitas (ERD) ke format HTML.
20. Mengkonversi SQL sesuai dengan database yang sedang digunakan untuk dapat dieksekusi oleh editor database.
21. Mengekspor entitas menjadi aplikasi GraphQL lengkap yang terdiri dari *back end* dan *front end* yang dapat langsung digunakan.

3.2.3 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
 - 1.1. Application
 - 1.2. Application Group
 - 1.3. Workspace
 - 1.4. Administrator
2. Role
 - 2.1. Administrator Workspace
 - 2.2. Application Group Member
3. Reference
 - 3.1. Administrator Level
4. Message
 - 4.1. Message
 - 4.2. Notification
5. Cache
 - 5.1. Error Cache
6. Package
 - 6.1. Starter Package
 - 6.2. Package Database
 - 6.3. Package Code
 - 6.4. Package Theme

7. Tools

7.1. Application Config

7.2. Database Migration

7.3. Database Admin

8. About

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.

Application Config digunakan untuk menyusun konfigurasi aplikasi di lingkungan produksi. Pengguna dapat mendefinisikan placeholder pada file application.yml yang nilainya akan diambil dari environment variable. Selain itu, Application Config menyediakan fitur untuk mengenkripsi konfigurasi yang bersifat rahasia. Dengan demikian, pengguna dapat menyalin file application.yml yang sudah aman ke lingkungan produksi, serta menggunakan script untuk menetapkan nilai environment variable baik dalam bentuk plain text maupun cipher text.

Database Migration digunakan untuk membuat konfigurasi migrasi database menggunakan MagicObject. Migrasi database yang dimaksud di sini adalah migrasi database dengan struktur yang berbeda baik nama tabel, nama kolom, bahkan tipe data. Tentu saja, perbedaan tipe data yang dimaksud di sini adalah tipe data yang masih kompatibel. Misalnya INTEGER menjadi FLOAT, TEXT menjadi VARCHAR, VARCHAR menjadi TEXT, BOOLEAN menjadi TINYINT(1) pada MySQL, atau TINYINT dengan rentang nilai 0 dan 1 menjadi TINYINT(1), ENUM menjadi VARCHAR, dan sebagainya. File konfigurasi memiliki format Yaml yang sangat sensitif terhadap indentasi.

File konfigurasi berisi

1. target (wajib)

Nama tabel dari database target.

2. source (wajib)

Nama tabel dari database sumber.

3. maximumRecord (opsional)

maximumRecord pada sebuah tabel digunakan untuk mengatur ulang jumlah record per query insert pada tabel tersebut. Pengaturan ini akan menimpa pengaturan global.

4. table

table adalah array objek yang berisi nama tabel dari database target dan nama tabel dari database sumber.

5. table[i].map (opsional)

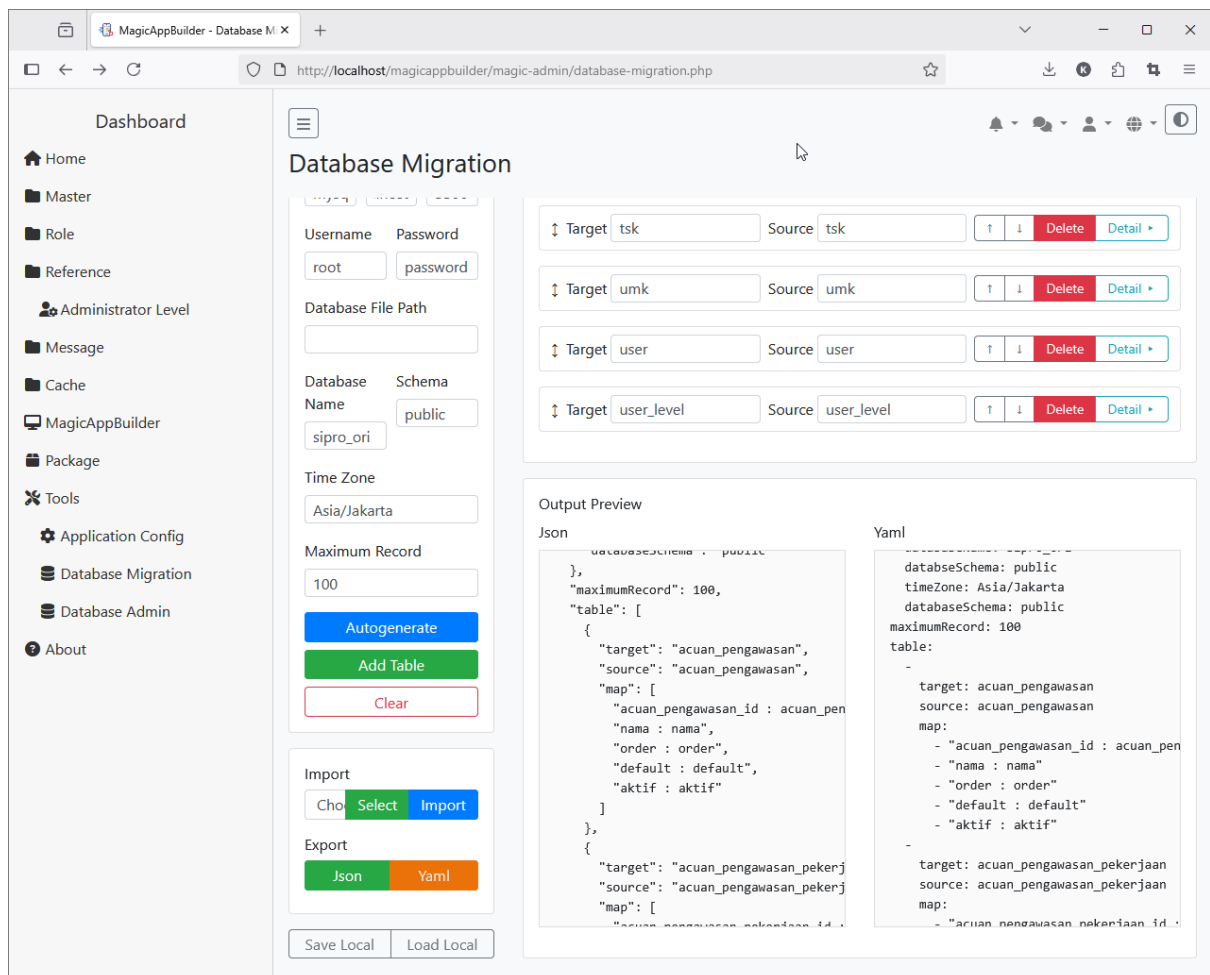
map adalah array teks yang dipisahkan oleh tanda titik dua. Di sisi kiri titik dua adalah nama kolom pada tabel dan database target, sedangkan di sisi kanan adalah nama kolom pada tabel dan database sumber.

5. table[i]preImportScript (opsional)

preImportScript adalah array query yang akan dijalankan sebelum proses impor data dimulai. Biasanya digunakan untuk menghapus data dari sebuah tabel dan mereset semua nilai sequence atau auto increment dari tabel target.

6. table[i]postImportScript (opsional)

postImportScript adalah array query yang akan dijalankan setelah proses impor data selesai. postImportScript dapat digunakan untuk berbagai tujuan seperti memperbaiki beberapa data pada tabel target termasuk mengambil nilai dari tabel lain. Oleh karena itu, postImportScript harus dijalankan setelah semua tabel berhasil diimpor.



Script di bawah ini digunakan untuk memigrasikan data antar database (dari **source** ke **target**) menggunakan **MagicObject**. Gunakan kelas yang sesuai dengan DBMS yang Anda gunakan. Kelas yang tersedia adalah sebagai berikut:

- PicoDatabaseUtilMySQL untuk database MySQL dan MariaDB
- PicoDatabaseUtilPostgreSql untuk database PostgreSQL
- PicoDatabaseUtilSqlite untuk database SQLite

Terdapat dua skenario pemakaian:

1. **Dump query ke file SQL** (untuk disimpan atau dieksekusi manual kemudian).
2. **Langsung eksekusi query ke database target.**

1. Skenario: Dump Query ke File

Pada mode ini, semua query hasil migrasi akan disimpan dalam file **db.sql**, tanpa langsung dieksekusi pada database target.

Cocok digunakan jika Anda ingin:

- Mengecek query hasil migrasi sebelum dijalankan.

- Menyimpan query untuk dokumentasi atau backup.
- Mengeksekusi query di server lain secara manual.

Contoh kode

```
<?php

use MagicObject\SecretObject;
use MagicObject\Util\Database\PicoDatabaseUtilMySQL;

require_once dirname(__DIR__) . "/vendor/autoload.php";

$config = new SecretObject();
$config->loadYamlFile('import.yml', true, true, true);

$fp = fopen(__DIR__ . '/db.sql', 'w');
if ($fp === false) {
    die("Gagal membuka file db.sql untuk ditulis\n");
}

$tool = new PicoDatabaseUtilMySQL();
$tableName = "transaction"; // nama tabel yang akan diimpor
$limit = 200;                // jumlah record per batch
$offset = 0;                 // offset data (untuk resume migrasi)

$sql = $tool->importData(
    $config,
    function($sql, $tableNameSource, $tableNameTarget, $databaseSource,
        $databaseTarget) use ($fp) {
        fwrite($fp, $sql . "\r\n\r\n");
    },
    $tableName,
    $limit,
    $offset
);
fclose($fp);

echo "Dump selesai. Query tersimpan di db.sql\n";
```

Hasil

- File **db.sql** berisi query INSERT sesuai data yang diambil dari database source.
- Tidak ada perubahan di database target.

2. Skenario: Eksekusi Langsung ke Database Target

Pada mode ini, query hasil migrasi langsung dijalankan di database target. Cocok digunakan jika Anda ingin:

- Migrasi data secara otomatis dan langsung.
- Tidak perlu menyimpan query terlebih dahulu.
- Data segera berpindah ke database tujuan.

Contoh kode

```
<?php

use MagicObject\SecretObject;
use MagicObject\Util\Database\PicoDatabaseUtilMySQL;

require_once dirname(__DIR__) . "/vendor/autoload.php";

$config = new SecretObject();
$config->loadYamlFile('import.yml', true, true, true);

$tool = new PicoDatabaseUtilMySQL();
$tableName = "transaction";
$limit = 200;
$offset = 0; // mulai dari awal

$sql = $tool->importData(
    $config,
    function($sql, $tableNameSource, $tableNameTarget, $databaseSource,
        $databaseTarget) {
        try {
            $databaseTarget->executeQuery($sql);
        } catch (Exception $e) {
            echo "Error: " . $e->getMessage() . "\n";
        }
        $databaseTarget->executeQuery($sql);
    },
    $tableName,
    $limit,
    $offset
);

echo "Migrasi selesai. Data berhasil dipindahkan ke database target\n";
```

Hasil

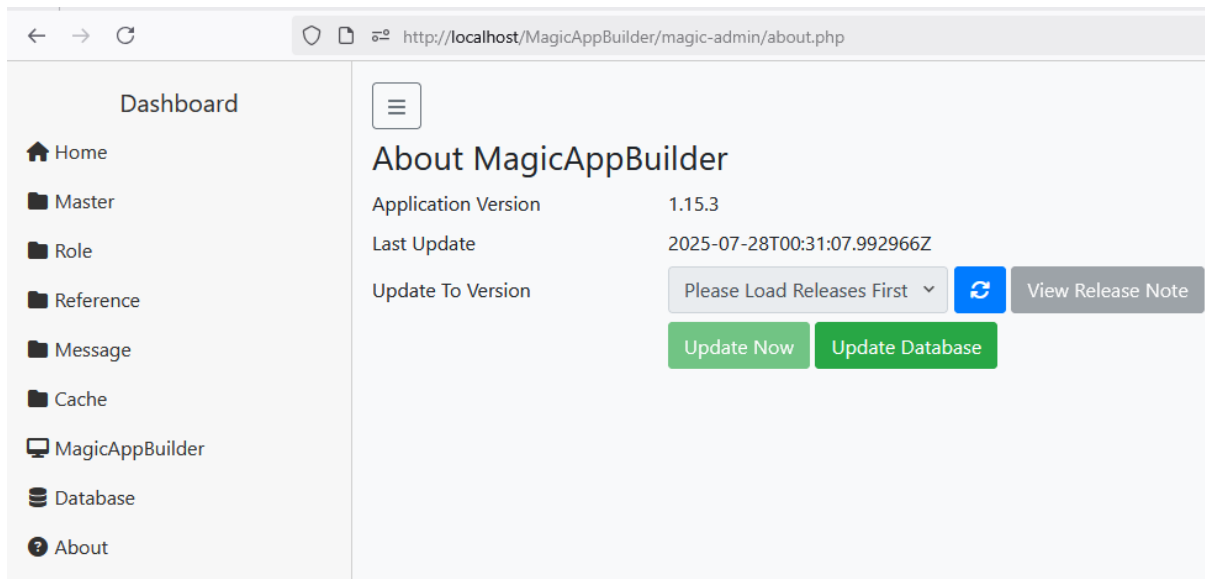
- Data langsung masuk ke **database target** sesuai konfigurasi di import.yml.
- Tidak ada file db.sql yang dihasilkan (kecuali Anda menambahkan logging manual).

Tips

- Gunakan **limit** dan **offset** jika data besar, agar migrasi bisa dilakukan bertahap.
- Mode **dump ke file** lebih aman untuk testing, karena query bisa diperiksa dulu.
- Mode **eksekusi langsung** lebih praktis, tapi pastikan data target siap menerima migrasi.

Hingga versi 1.21.1, MagicAppBuilder tidak menyediakan halaman khusus untuk melakukan migrasi database. Isu keamanan menjadi pertimbangan utama untuk tidak memasukkan fitur ini ke dalam MagicAppBuilder.

Pada menu About, pengguna dapat melihat versi MagicAppBuilder yang sedang digunakan. Pengguna dapat memperbarui versi dari halaman ini. Tidak disarankan menurunkan versi aplikasi. Versi terbaru tidak hanya tentang fitur tetapi juga perbaikan bug. Penggunaan versi MagicAppBuilder yang lebih lama berpotensi menyebabkan terjadinya bug.



Sebelum melakukan pembaruan, klik tombol reload berwarna biru terlebih dahulu. Jika Anda ingin melihat catatan rilis, klik tombol **View Release Note**. Jika terjadi kesalahan, klik tombol **Update Database** agar MagicAppBuilder memperbarui struktur database sesuai dengan entitas yang ada pada kode PHP. Ulangi proses dengan memilih tombol reload.

Untuk melakukan pembaruan, pilih tombol **Update Now**. MagicAppBuilder akan mengunduh rilis terbaru dari MagicAppBuilder. Setelah proses pengunduhan selesai, akan dilanjutkan dengan proses ekstraksi. Setelah proses ekstraksi selesai, MagicAppBuilder akan menghapus file unduhan dan memperbarui struktur database sesuai dengan entitas yang ada pada kode PHP yang terbaru. Jika semua proses berjalan baik, MagicAppBuilder akan menampilkan pesan bahwa proses pembaruan berhasil dilakukan.

Setelah melakukan pembaruan, jangan lupa untuk menghapus semua cache browser. Browser menyimpan cache file-file statis seperti CSS, JS, font, gambar, dan sebagainya.

Cara Menghapus Cache di Berbagai Browser

1. Google Chrome

- Tekan Ctrl + Shift + Delete (Windows) atau Cmd + Shift + Delete (Mac)
- Pilih rentang waktu: **All time (Semua waktu)**

- Centang **Cached images and files**
- Klik **Clear data (Hapus data)**

2. Mozilla Firefox

- Tekan Ctrl + Shift + Delete atau Cmd + Shift + Delete
- Pilih rentang waktu: **Everything (Semua)**
- Centang hanya **Cache**
- Klik **OK**

3. Microsoft Edge

- Tekan Ctrl + Shift + Delete
- Pilih rentang waktu: **All time**
- Centang **Cached images and files**
- Klik **Clear now**

4. Safari (Mac)

- Buka menu Safari > **Preferences** > **Advanced**, lalu centang **Show Develop menu**
- Buka menu **Develop** > pilih **Empty Caches**
- Atau tekan shortcut: Cmd + Option + E

5. Opera

- Tekan Ctrl + Shift + Delete
- Pilih rentang waktu: **All time**
- Centang **Cached images and files**
- Klik **Clear data**

6. Brave

- Tekan Ctrl + Shift + Delete
- Pilih rentang waktu: **All time**
- Centang **Cached images and files**
- Klik **Clear data**

Tips Tambahan

- Setelah menghapus cache, **refresh halaman dengan Ctrl + F5** untuk memastikan tidak ada cache tersisa.
- Jika Anda menggunakan **aplikasi berbasis PWA** atau shortcut ke desktop/mobile, hapus dan pasang ulang untuk memuat versi terbaru.

3.3 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.

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**.

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

3.3.1 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.

File Manager tidak menyediakan editor file biner. Jangan mencoba untuk mengubah isi file benar meskipun isi file tersebut ditampilkan di editor. Editor tidak bisa

menangani *unprintable characters* sehingga saat file biner disimpan, baik diubah atau tidak oleh pengguna, file tersebut akan rusak.

3.3.2 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.
- Pengguna hanya bisa membuat file teks karena File Manager tidak menyediakan opsi membuat atau mengubah file biner.

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.
- Pengguna dapat mengunggah baik file text maupun file biner.
- Nama file pada lokasi target akan dengan nama file sumber yang 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.
- File Manager akan mempacketkan file-file yang ada di dalam direktori tersebut ke dalam sebuah file ZIP.
- Pengguna dapat mengekstrak file ZIP tersebut.

8. Hapus Direktori

- Pilih opsi ini untuk menghapus direktori yang sedang dipilih beserta seluruh isinya.

3.3.3 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.
- Pengguna hanya bisa membuat file teks karena File Manager tidak menyediakan opsi membuat atau mengubah file biner.

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.

3.3.4 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.
- Beberapa jenis file diperlakukan secara khusus saat pengguna membukanya.
 - a. File bertipe teks seperti kode program atau file konfigurasi akan ditampilkan sebagai teks dan dapat diedit. Prolaku editor akan disesuaikan dengan ekstensi file seperti .html, .css, .scss, .js, .yaml, .ini, .txt, dan sebagainya. Pengguna dapat mengedit file ini di editor.
 - b. File bertipe database SQLite dengan ekstensi .sqlite dan .db akan dibuka sebagai database SQLite. Pengguna dapat melihat keseluruhan data yang ada di database namun pengguna tidak dapat mengubahnya.
 - c. File bertipe Word Document dengan ekstensi .docx akan ditampilkan sebagai halaman HTML. Tampilannya akan mirip dengan tampilan dokumen tersebut jika dibuka menggunakan aplikasi Microsoft Word dengan beberapa perbedaan. Pengguna tidak dapat mengubah maupun mencetak isi dokumen ini.
 - d. File bertipe spreadsheet seperti .csv, .xls, .xlsx dan .ods akan ditampilkan sebagai tabel. File dengan ekstensi .xls, .xlsx dan .ods akan ditampilkan dengan multipel sheet. Pengguna dapat melihat semua sheet yang ada pada file tersebut. Sel yang berisi formula akan ditampilkan dengan hasil dari formula tersebut. Pengguna tidak dapat mengubah maupun mencetak isi dokumen ini.
 - e. File bertipe Portable Data Format dengan ekstensi .pdf akan ditampilkan sebagai halaman HTML. Tampilannya akan mirip dengan tampilan dokumen tersebut jika dibuka langsung menggunakan browser. Pengguna tidak dapat mengubah maupun mencetak isi dokumen ini.

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.
- Nama dan ekstensi file yang diunduh akan sama dengan nama dan ekstensi file sumber yang diunduh dengan catatan pengguna belum mengunduh file dengan nama yang sama sebelumnya.
- Browser memiliki mekanisme tersendiri dalam penamaan file jika pengguna pernah mengunduh file ke lokasi yang sama sebelumnya sesuai dengan pengaturan pada browser yang berlaku. Pengguna mungkin diberi wewenang untuk mengubah pengaturan tersebut.

4. Hapus File

- Pilih opsi ini untuk menghapus file yang dipilih. Pengguna akan diminta konfirmasi sebelum file dihapus secara permanen.

3.3.5 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

4.1 Versi MagicAppBuilder

Dokumen panduan ini didasarkan pada fitur-fitur yang tersedia pada **MagicAppBuilder versi 2.1.0**. Fitur yang dirilis setelah versi tersebut mungkin belum disertakan dalam pembahasan ini. Namun, pemahaman Anda terhadap versi 2.1.0 akan memudahkan adaptasi dengan fitur-fitur baru, mengingat MagicAppBuilder dikembangkan dengan pendekatan yang intuitif.

4.2 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

8. **Translate Entity**

Tab ini memungkinkan pengguna membuat file lokalisasi untuk entitas, sehingga mendukung banyak bahasa dalam aplikasi.

9. **Translate Module**

Tab ini memungkinkan pengguna membuat file lokalisasi untuk modul, sehingga mendukung banyak bahasa dalam aplikasi.

10. 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.

11. 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.

12. 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.

13. 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 perilaku aplikasi yang tidak dikehendaki.

14. 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.

15. Logout

Tautan untuk keluar dari sesi administrator.

4.3 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:

4.1.1 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)
- **MagicServer** (Server yang didesain khusus untuk MagicAppBuilder)

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.

Ikuti langkah-langkah instalasi server yang dipilih. Bagi pemula, jangan pernah menginstal lebih dari satu server karena akan menyebabkan konflik penggunaan port. Pengguna tingkat lanjutan diijinkan menginstal lebih dari satu server dan dapat mengonfigurasi port server yang digunakan sesuai dengan kebutuhan.

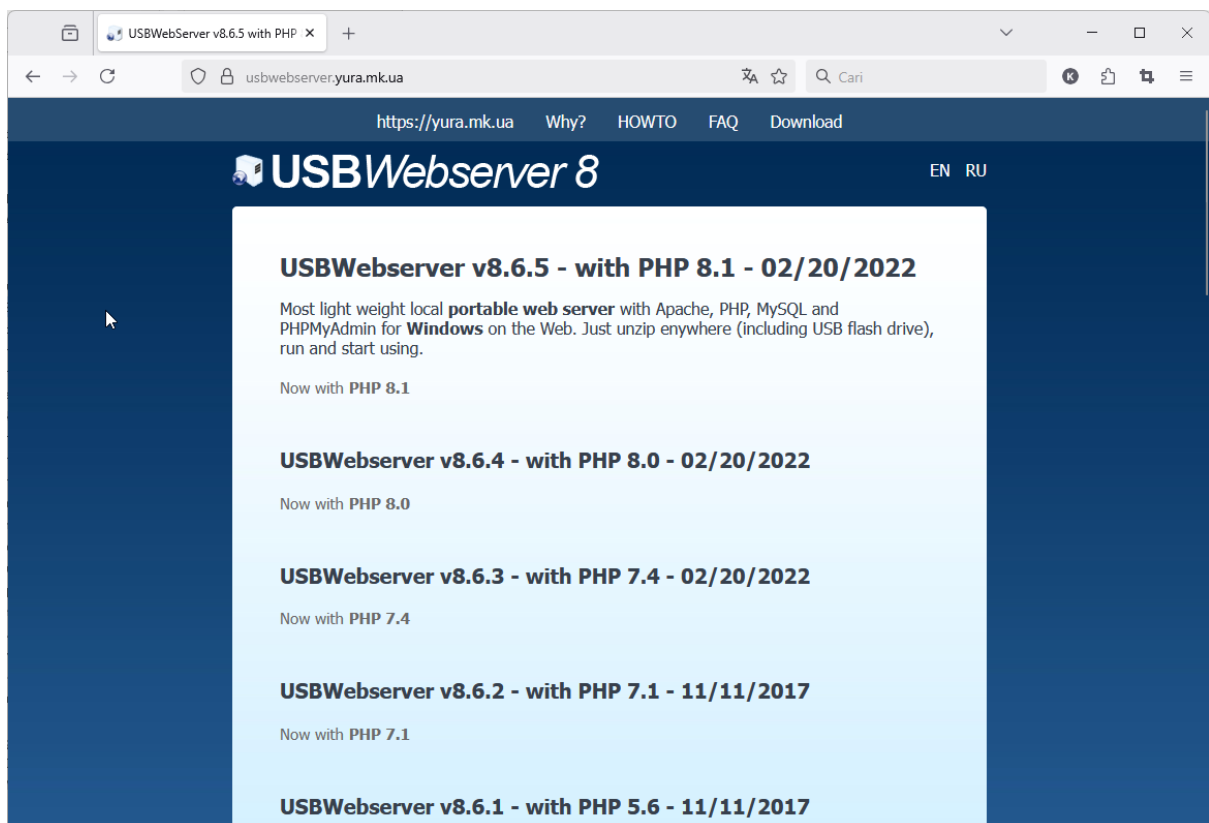
Berikut ini adalah tangkapan layar dari website resmi server di atas.



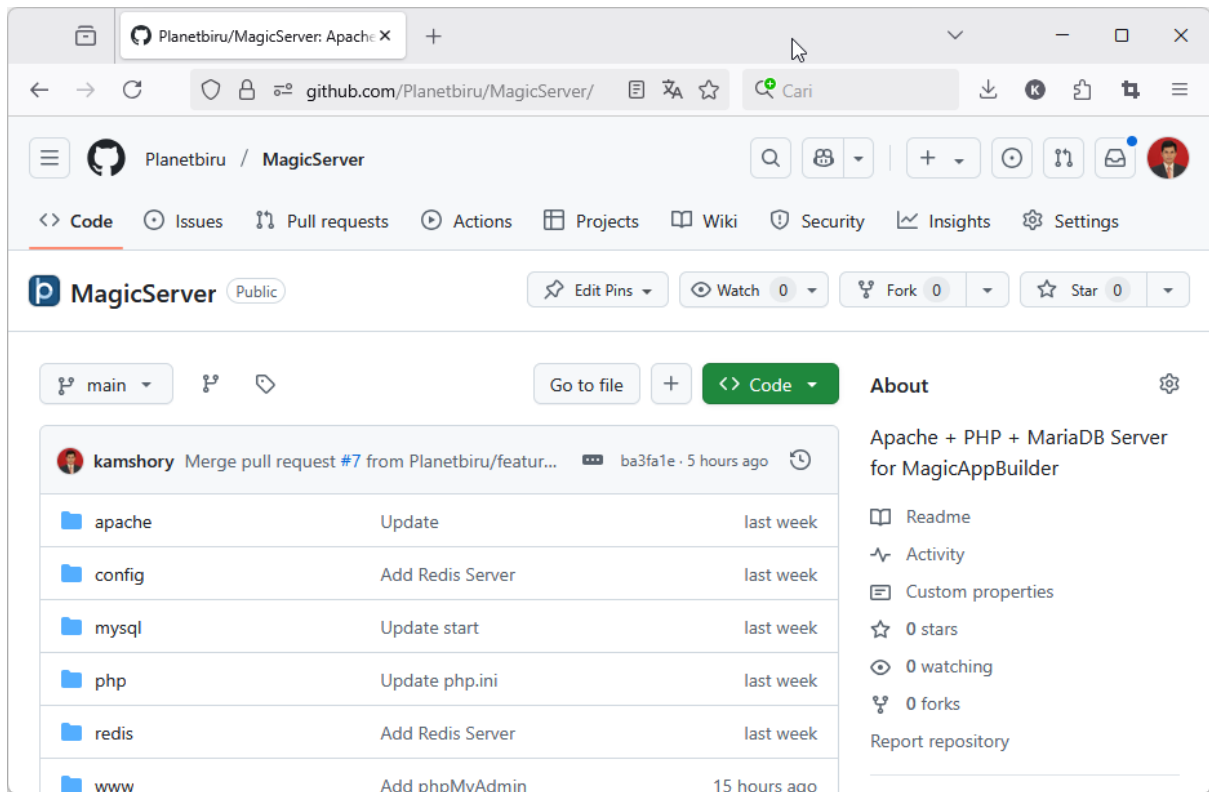
WAMPSEVER



XAMPP



USBWebServer 8



MagicServer

Proses Instalasi MagicServer

Pada Windows 10 atau yang lebih lama, unduh dan install Install Microsoft Visual C++ Redistributable Runtime. URL untuk mengunduhnya adalah

https://aka.ms/vs/17/release/vc_redist.x64.exe

Untuk menginstal MagicServer, kunjungi halaman resmi repositori MagicServer di Github dengan alamat <https://github.com/Planetbiru/MagicServer>

Planetbiru/MagicServer: Apache X

github.com/Planetbiru/ MagicServer

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

MagicServer Public

Edit Pins Watch 0 Fork 0 Star 0

main Go to file Code

About

Apache + PHP + MariaDB Server for MagicAppBuilder

Readme Activity Custom properties 0 stars 0 watching 0 forks Report repository

Releases 4

0.2.1 Latest

Files

apache	Update	2 weeks ago
config	Add redis extension for PHP	3 minutes ago
mysql	Update start	2 weeks ago
php	Add redis extension for PHP	3 minutes ago
redis	Add Redis Server	last week
www	Add phpMyAdmin	3 days ago
.gitignore	Update README.md	3 days ago

Release 0.2.1 · Planetbiru/Magic X

github.com/Planetbiru/MagicServer

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

Releases / 0.2.1

0.2.1 Latest Compare

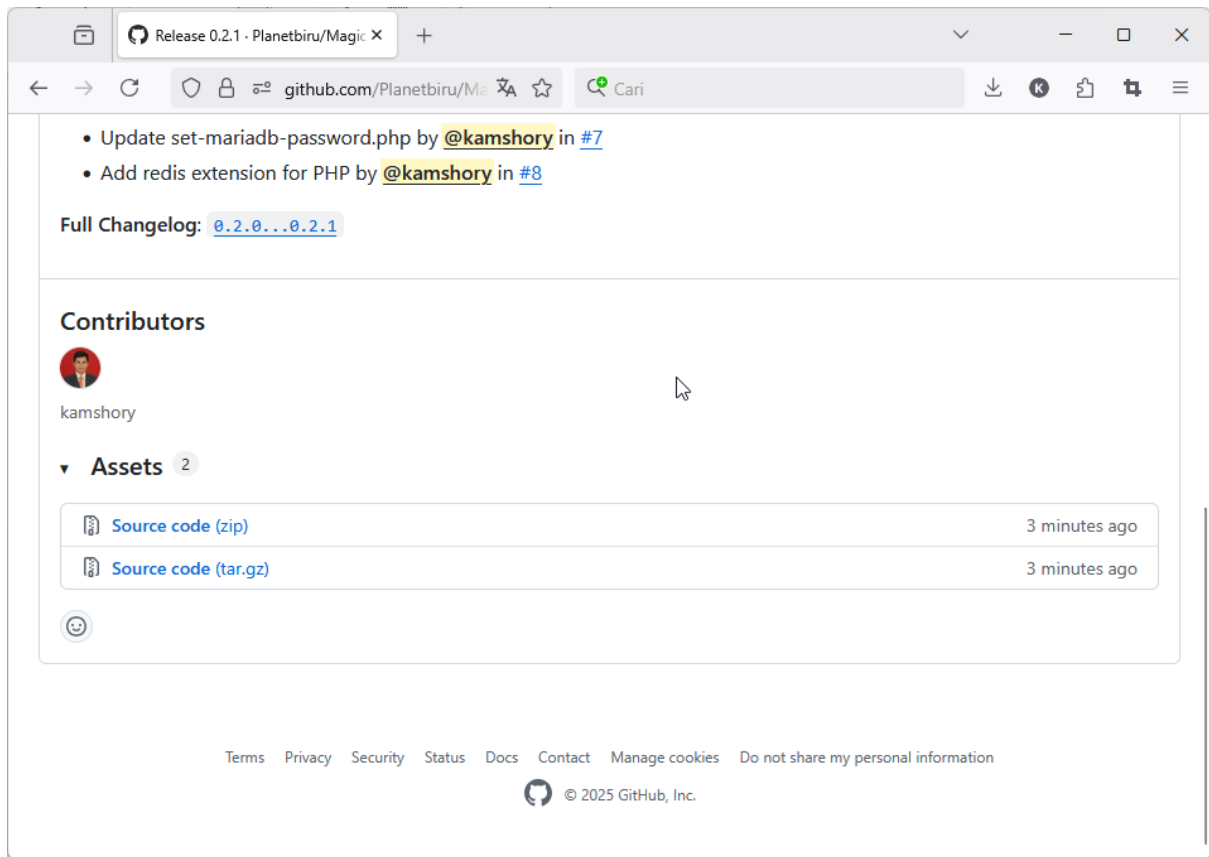
kamshory released this 2 minutes ago 0.2.1 628847e

MagicServer Version 0.2.1

Changes

- Redis PHP Extension Added**
The Redis PHP extension is now included by default, enabling Redis-based features such as session storage and caching to work out of the box.

What's Changed

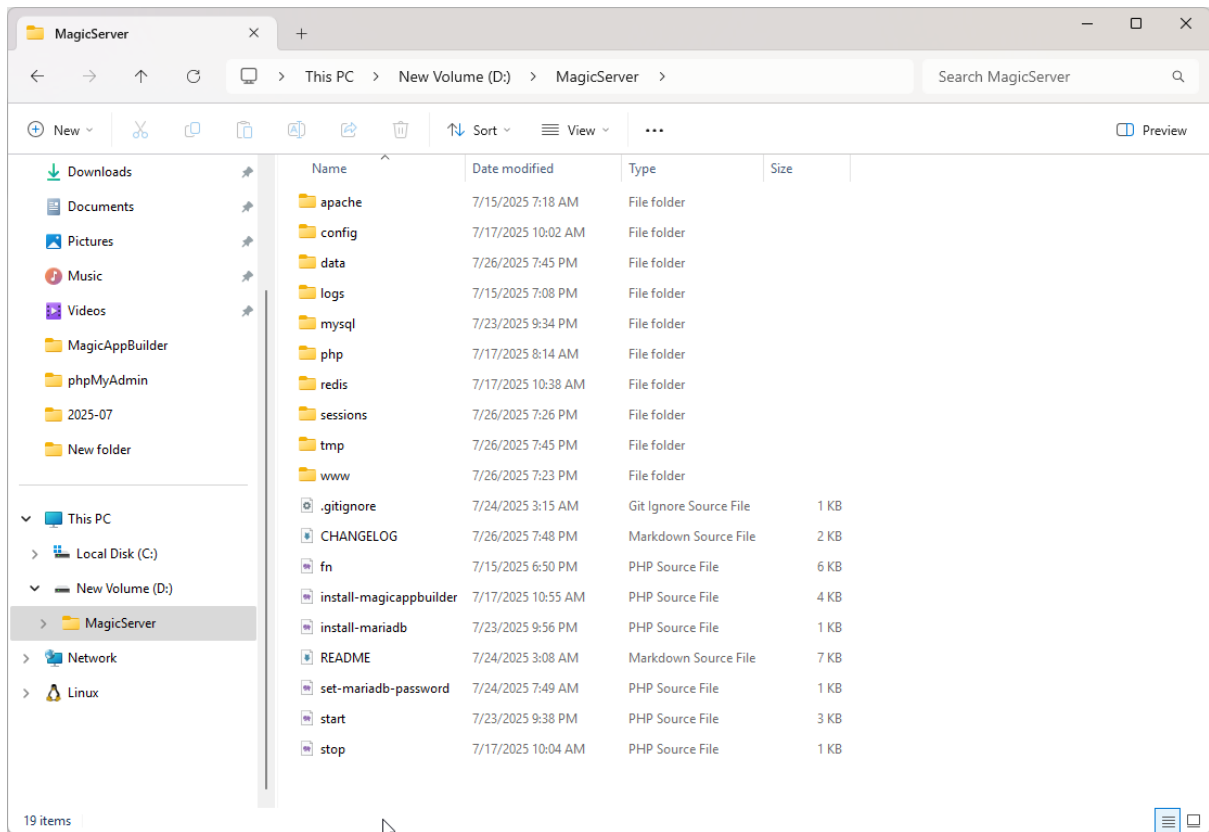


Unduh file Source code.zip dari versi rilis terbaru MagicServer lalu simpan direktori

C:\MagicServer

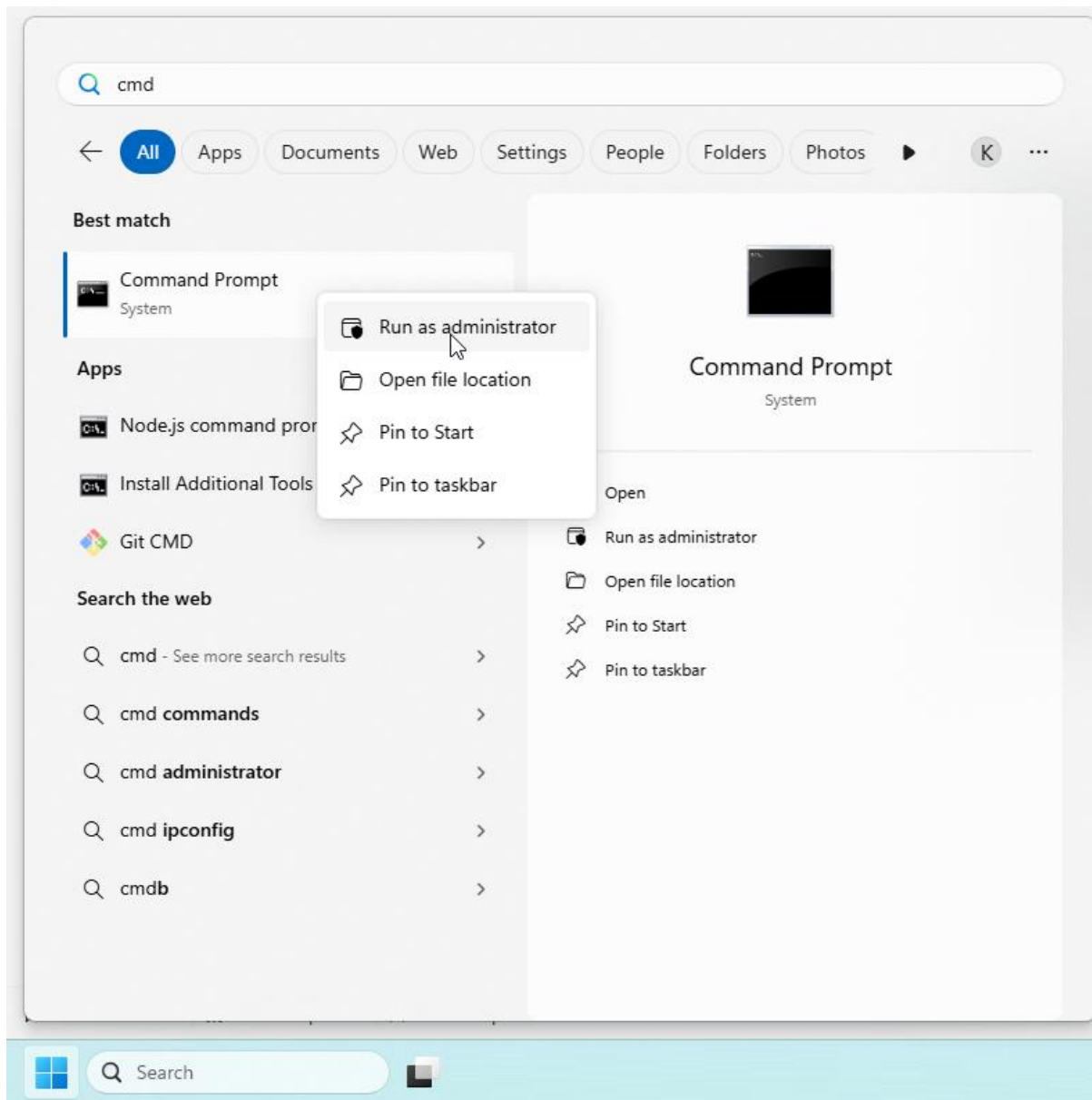
Atau

D:\MagicServer (jika drive D tersedia)



Pastikan bahwa file `install.php`, `start.php` dan `stop.php` berada di direktori `C:\MagicServer` atau `D:\MagicServer` dan bukan berada di dalam direktori di bawahnya.

a. Instalasi Apache, MariaDB, Redis dan PHP



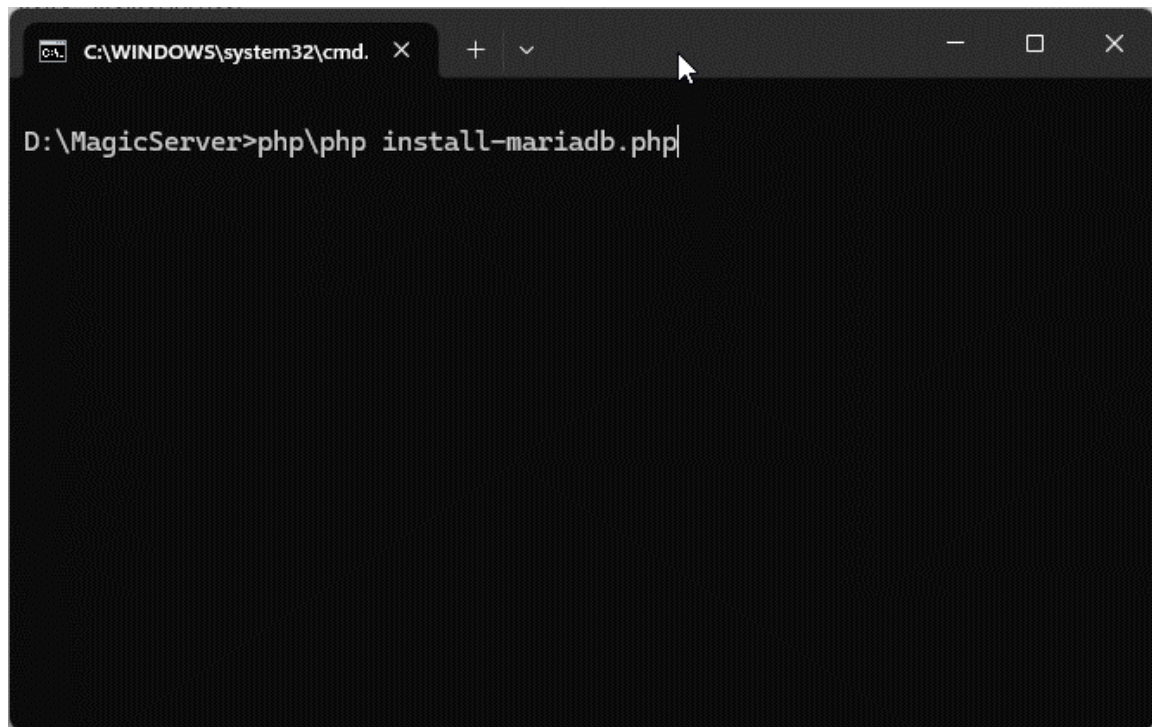
b. **Persiapan Struktur Awal MariaDB**

Pada paket instalasi, tidak ada data awal untuk database MariaDB. Pengguna harus melakukan persiapan agar server MariaDB dapat digunakan.

Pada Command Prompt, ketik perintah sebagai berikut:

```
php\php.exe install-mariadb.php
```

lalu tekan tombol Enter di keyboard.



```
C:\WINDOWS\system32\cmd. X + v - □ X
D:\MagicServer>php\php install-mariadb.php|
```

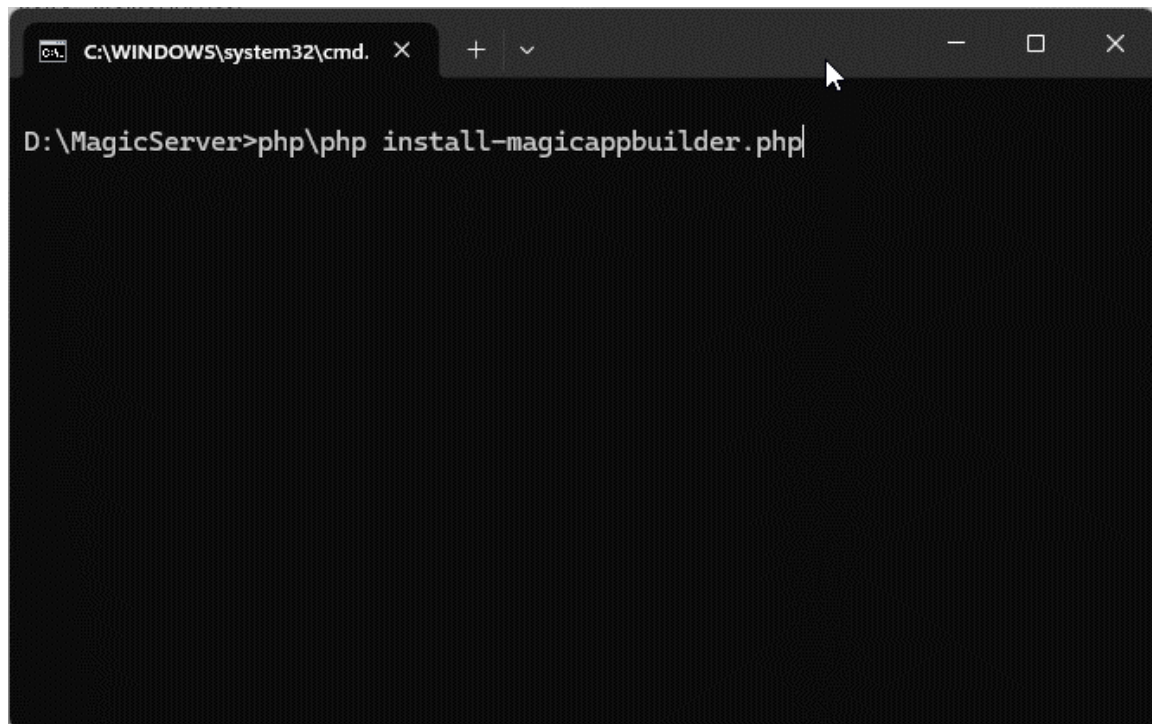
c. Instalasi MagicAppBuilder

Pada Command Prompt, ketik perintah sebagai berikut:

```
php\php.exe install-magicappbuilder.php
```

lalu tekan tombol Enter di keyboard.

Jika muncul pesan error atau warning, ulangi lagi.



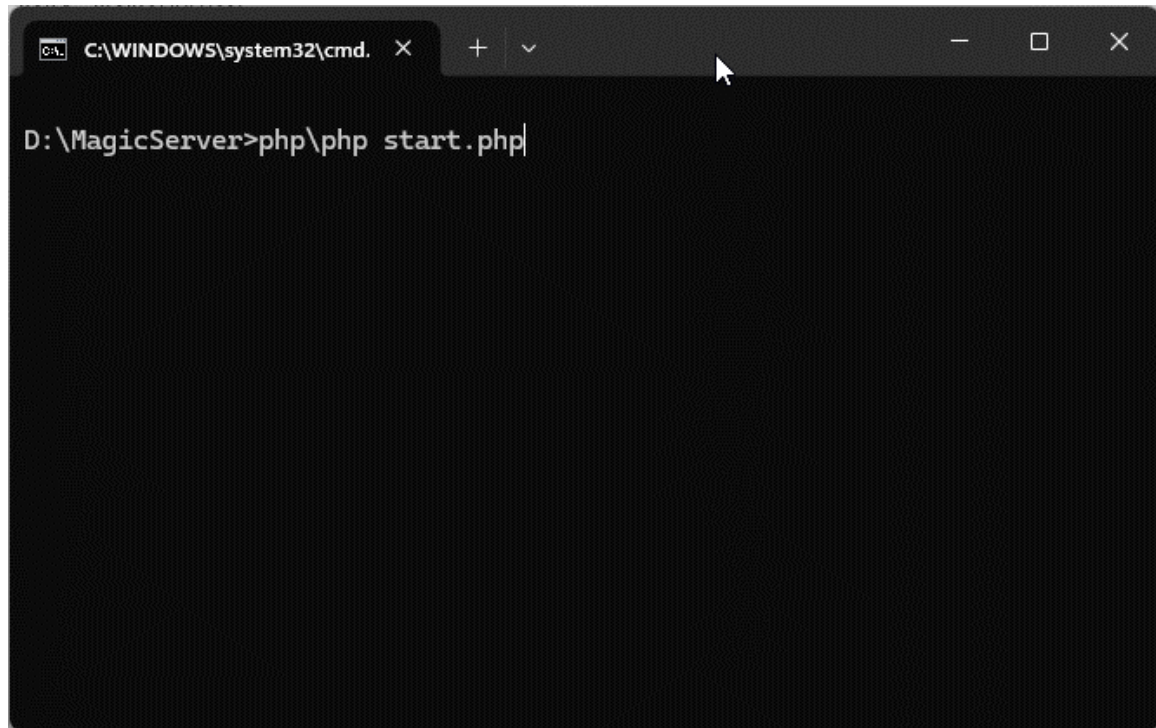
```
C:\WINDOWS\system32\cmd. X + v - □ X
D:\MagicServer>php\php install-magicappbuilder.php|
```

d. Menjalankan server Apache, MariaDB dan Redis

Jalankan server dengan perintah sebagai berikut:

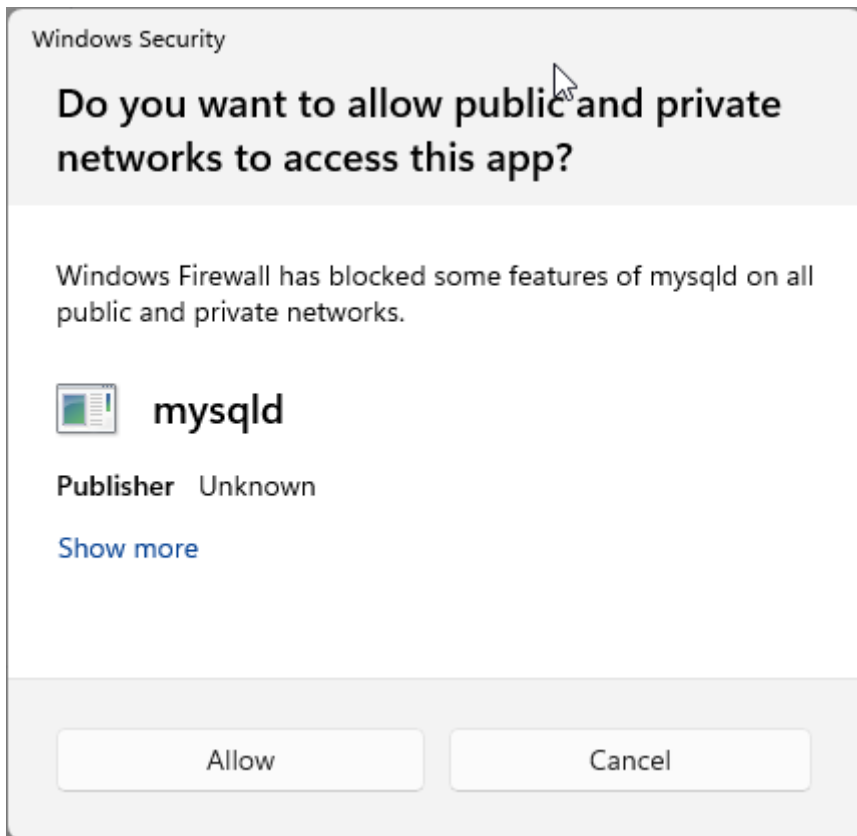
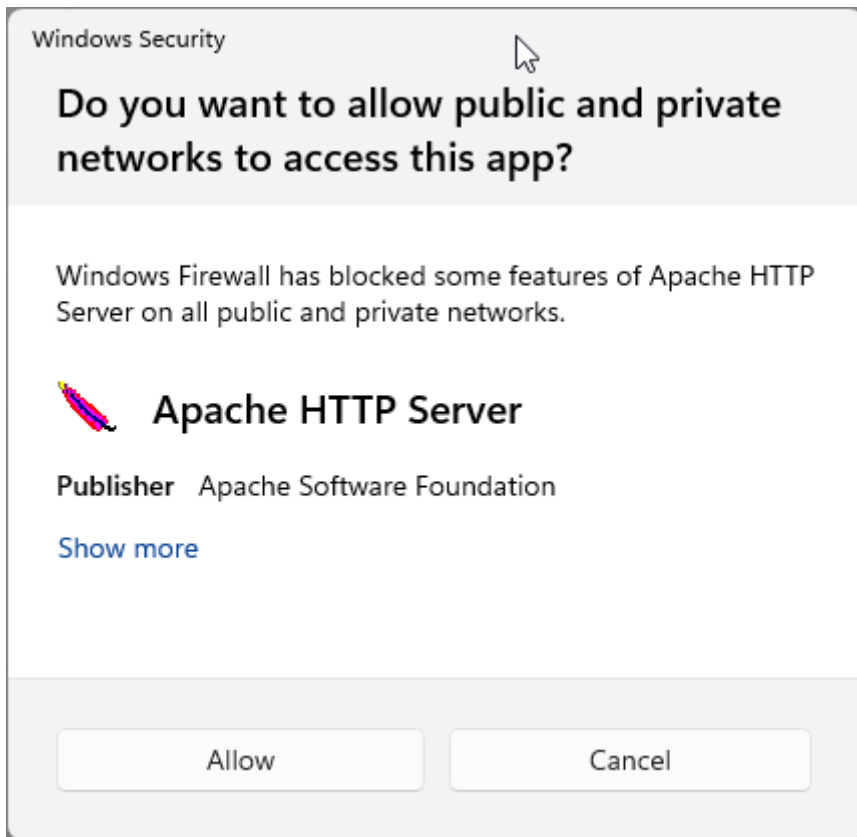
```
php\php.exe start.php
```

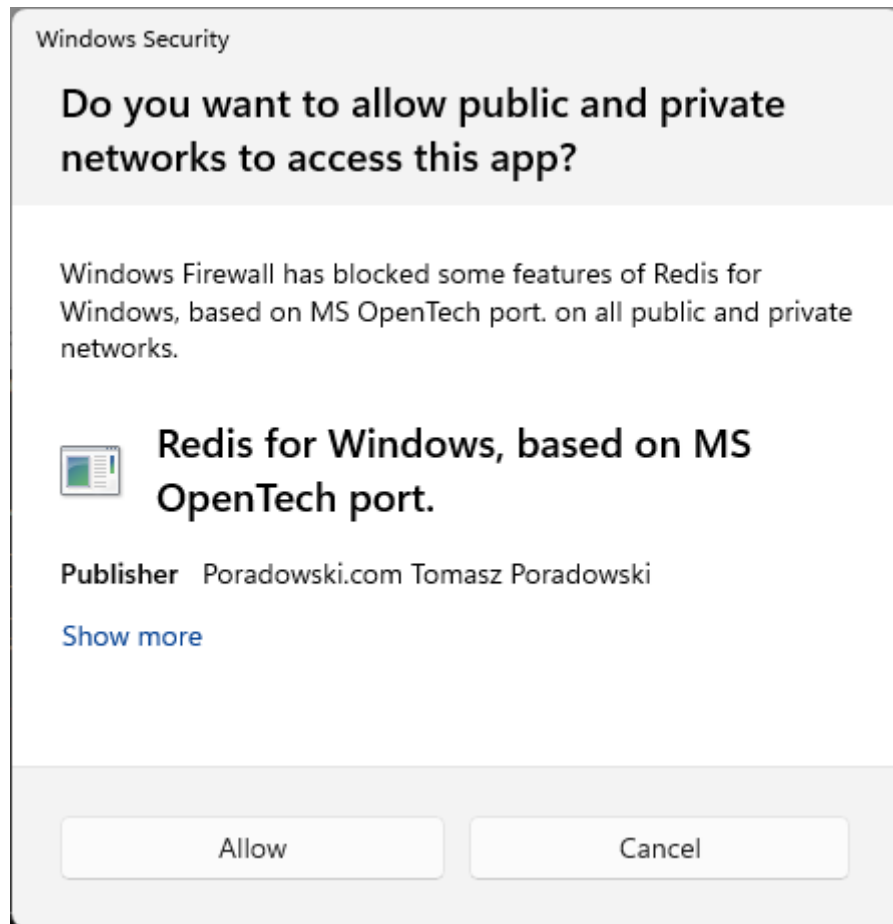
lalu tekan tombol Enter di keyboard.



Jangan menutup Command Prompt di mana Anda menjalankan script start.php karena mungkin Windows akan mengakhiri semua service yang Anda jalankan.

Jika muncul dialog seperti berikut, pilih tombol **Allow**.

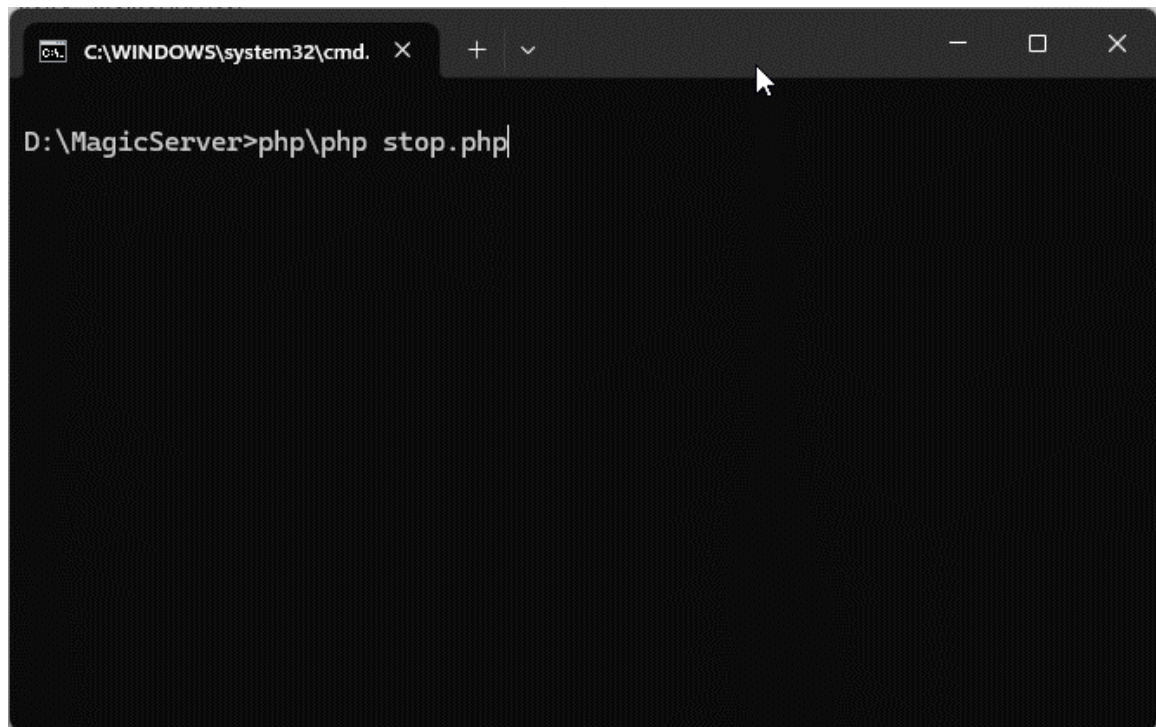




Jika Anda ingin mengakhiri semua service tanpa menutup Command Prompt tersebut, jalankan perintah sebagai berikut:

```
php\php.exe stop.php
```

lalu tekan tombol Enter di keyboard.

A screenshot of a Windows Command Prompt window. The title bar shows the path 'C:\WINDOWS\system32\cmd.' and standard window controls. The command prompt shows the directory 'D:\MagicServer' and the command 'php\php stop.php' entered at the prompt.

Pasca Instalasi

Jika menggunakan MagicServer, pengguna sedikit dimudahkan dalam proses instalasi. Konfigurasi default sudah disesuaikan dengan kebutuhan dari MagicAppBuilder sehingga pengguna tidak perlu melakukan konfigurasi di awal.

Jika menggunakan server selain MagicServer, di sini dicontohkan dengan XAMPP dan diinstal di drive D, ikuti langkah-langkah sebagai berikut:

1. **Unduh dan Ekstrak MagicAppBuilder ke Document Root Server**
2. **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.
3. **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.
4. **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.

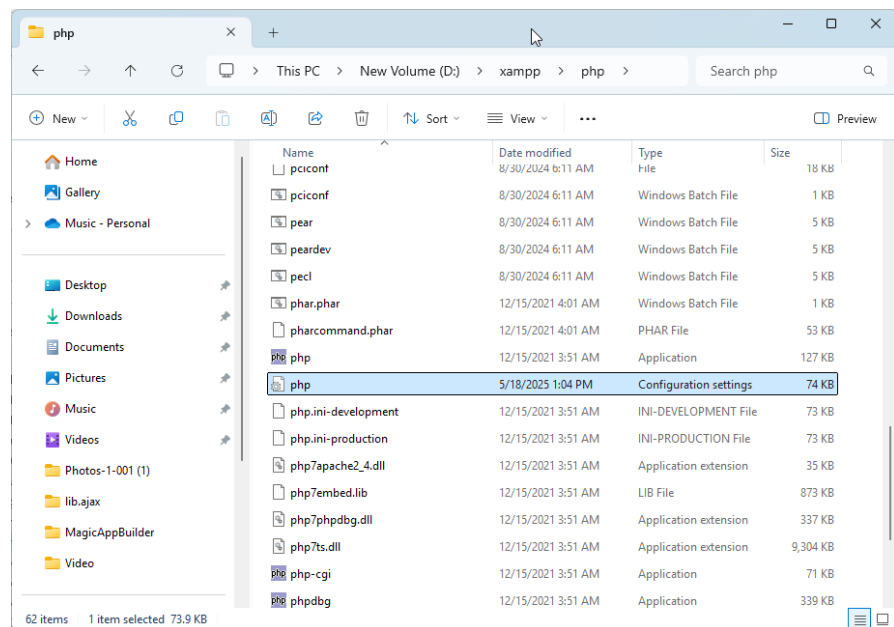
5. Mengaktifkan Driver SQLite di Windows

- 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 tidak akan dapat digunakan.

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

a. Cari File **php.ini**

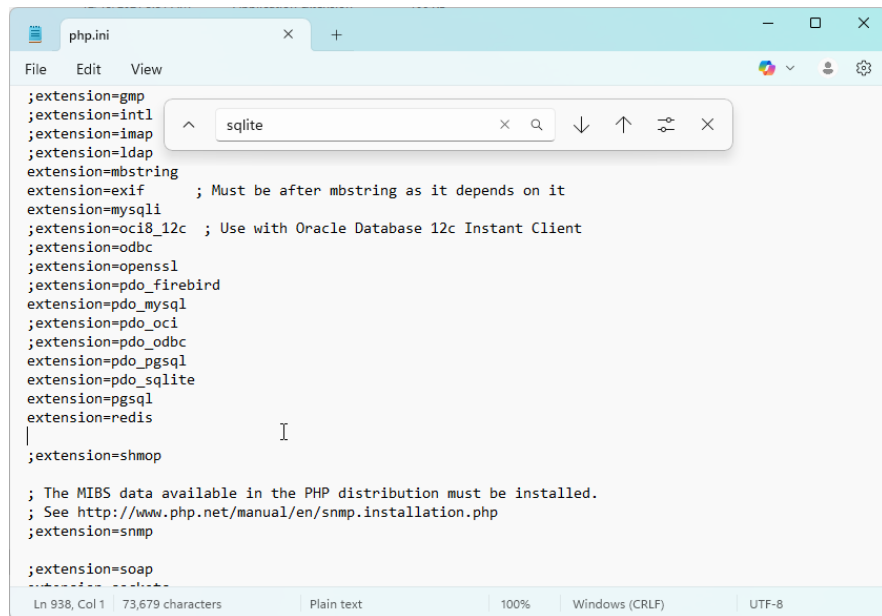
- File **php.ini** adalah file konfigurasi utama untuk PHP. Lokasi file ini tergantung pada instalasi PHP pengguna, tetapi jika menggunakan XAMPP, biasanya dapat ditemukan di D:\xampp\php\php.ini jika pengguna menginstal XAMPP di drive C.



b. Buka File **php.ini**

- Buka file **php.ini** dengan editor teks seperti Notepad atau Visual Studio Code (VSCode).

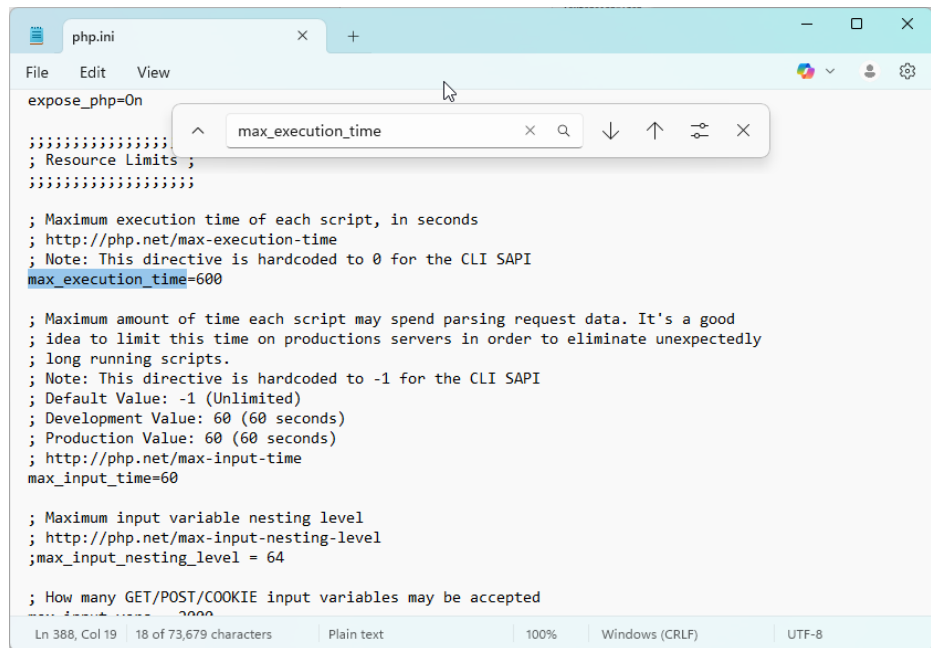
c. Temukan Ekstensi SQLite

A screenshot of a text editor window titled 'php.ini'. The editor has a menu bar with 'File', 'Edit', and 'View'. A search bar is open with the text 'sqlite'. The code content shows various PHP extensions being configured, including 'extension=gmp', 'extension=intl', 'extension=imap', 'extension=ldap', 'extension=mbstring', 'extension=exif', 'extension=mysqli', 'extension=oci8_12c', 'extension=odbc', 'extension=openssl', 'extension=pdo_firebird', 'extension=pdo_mysql', 'extension=pdo_oci', 'extension=pdo_odbc', 'extension=pdo_pgsql', 'extension=pdo_sqlite', 'extension=pgsql', 'extension=redis', 'extension=shmop', 'extension=soap', and 'extension=snmp'. The status bar at the bottom indicates 'Ln 938, Col 1', '73,679 characters', 'Plain text', '100%', 'Windows (CRLF)', and 'UTF-8'.

- Cari baris berikut dalam file **php.ini** (gunakan fungsi *Find* dengan menekan **Ctrl + F** di editor teks):
- `extension=sqlite3`
- `extension=pdo_sqlite`
- Jika tidak ditemukan, tambahkan kedua ekstensi di atas.
- Jika ditemukan, pastikan tidak ada tanda titik koma (;) di depan. Tanda titik koma menyebabkan ekstensi tidak dimuat.

d. **Temukan Pengaturan `max_execution_time`**

- Pengaturan ini digunakan untuk membatasi berapa lama PHP menjalankan script.
- Pengaturan bawaan biasanya adalah 60
- Ubah menjadi 600



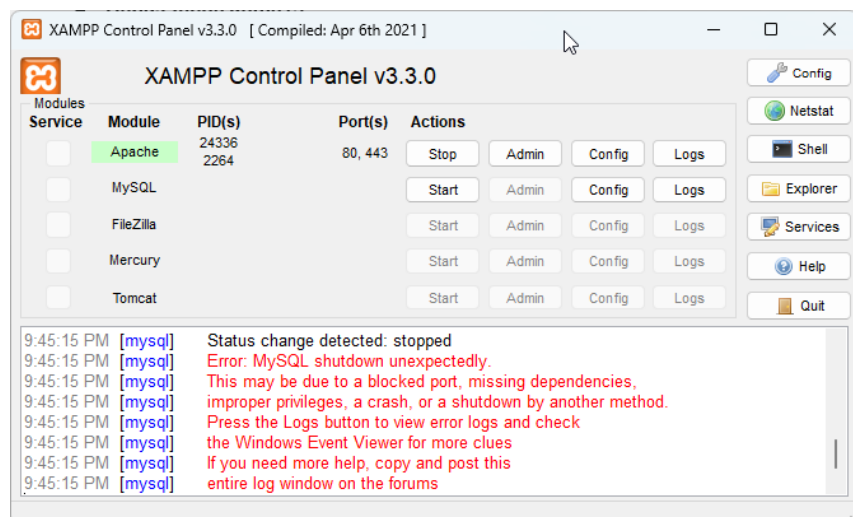
```
php.ini
File Edit View
expose_php=On
; Resource Limits ;
; Maximum execution time of each script, in seconds
; http://php.net/max-execution-time
; Note: This directive is hardcoded to 0 for the CLI SAPI
max_execution_time=600
; Maximum amount of time each script may spend parsing request data. It's a good
; idea to limit this time on productions servers in order to eliminate unexpectedly
; long running scripts.
; Note: This directive is hardcoded to -1 for the CLI SAPI
; Default Value: -1 (Unlimited)
; Development Value: 60 (60 seconds)
; Production Value: 60 (60 seconds)
; http://php.net/max-input-time
max_input_time=60
; Maximum input variable nesting level
; http://php.net/max-input-nesting-level
;max_input_nesting_level = 64
; How many GET/POST/COOKIE input variables may be accepted
max_input_vars = 1000
Ln 388, Col 19 | 18 of 73,679 characters | Plain text | 100% | Windows (CRLF) | UTF-8
```

e. **Simpan dan Tutup File php.ini**

- Setelah menghapus tanda titik koma, simpan perubahan dan tutup editor teks pengguna.

f. **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.



g. **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 **D:\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.

4.1.2 Langkah 2: Mengunduh dan Memasang MagicAppBuilder di Server

Mengunduh dan memasang MagicAppBuilder tidak perlu dilakukan secara manual jika Anda menggunakan MagicServer. MagicServer telah menyediakan file **install.php** untuk mengunduh dan memasang MagicAppBuilder secara otomatis.

Untuk memasang MagicAppBuilder di MagicServer, masuk ke document root dari MagicServer dengan Command Prompt lalu jalankan perintah:

```
php\php.exe install.php
```

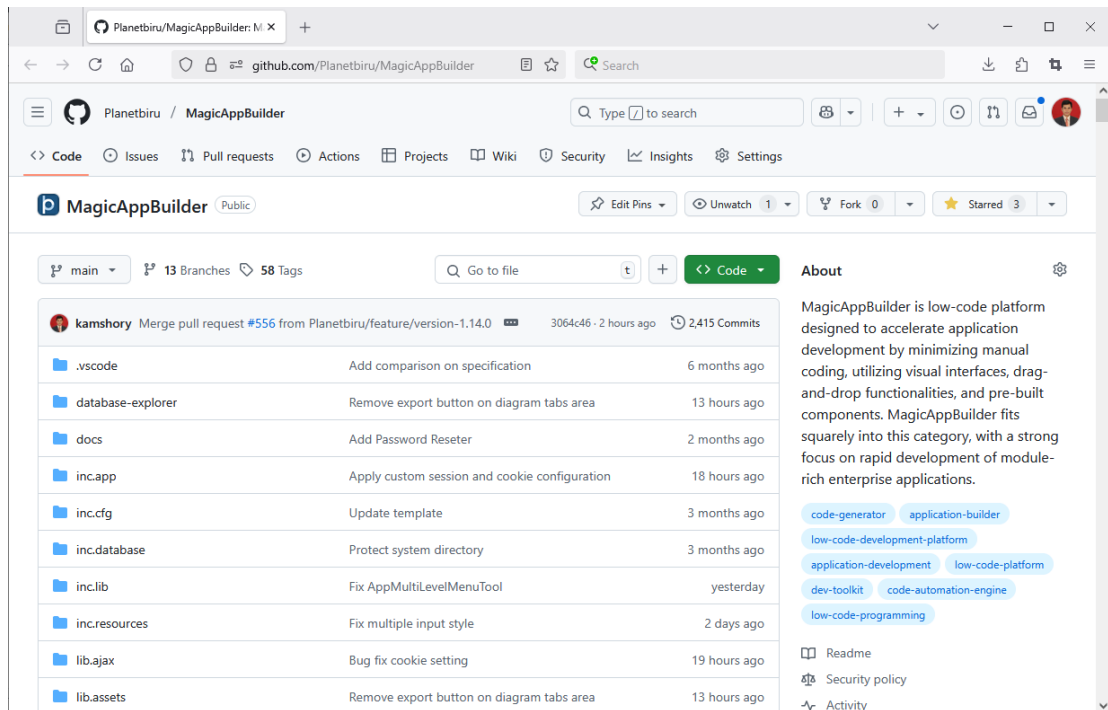
lalu tekan tombol Enter di keyboard. Kode pada file `install.php` akan menngunduh dan mengekstrak MagicAppBuilder versi terbaru. Setelah proses instalasi selesai, silakan buka URL sesuai dengan alamat yang tertera pada Command Prompt.

Jika Anda menggunakan server lain seperti XAMPP, Wamp Server, USBWebServer, dan sebagainya, Anda perlu untuk melakukan pemasangan manual MagicAppBuilder di server Anda. Untuk memasang MagicAppBuilder di server, ikuti langkah-langkah berikut:

1. Kunjungi Repositori GitHub MagicAppBuilder

Buka repositori resmi MagicAppBuilder di:

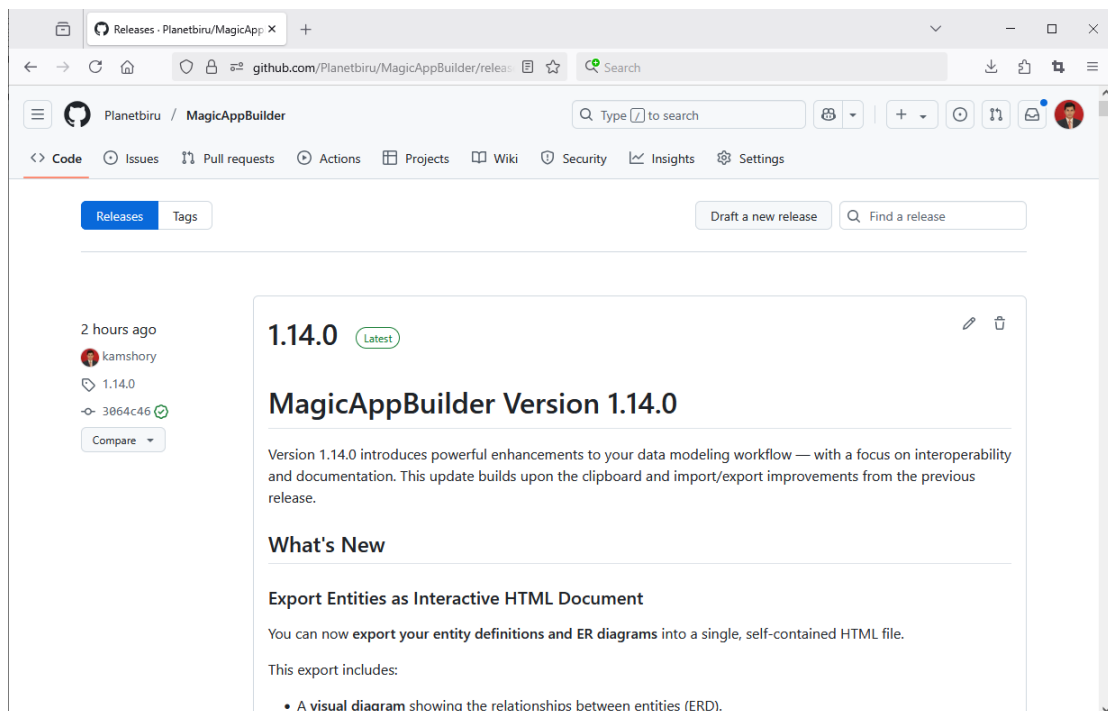
<https://github.com/Planetbiru/MagicAppBuilder>

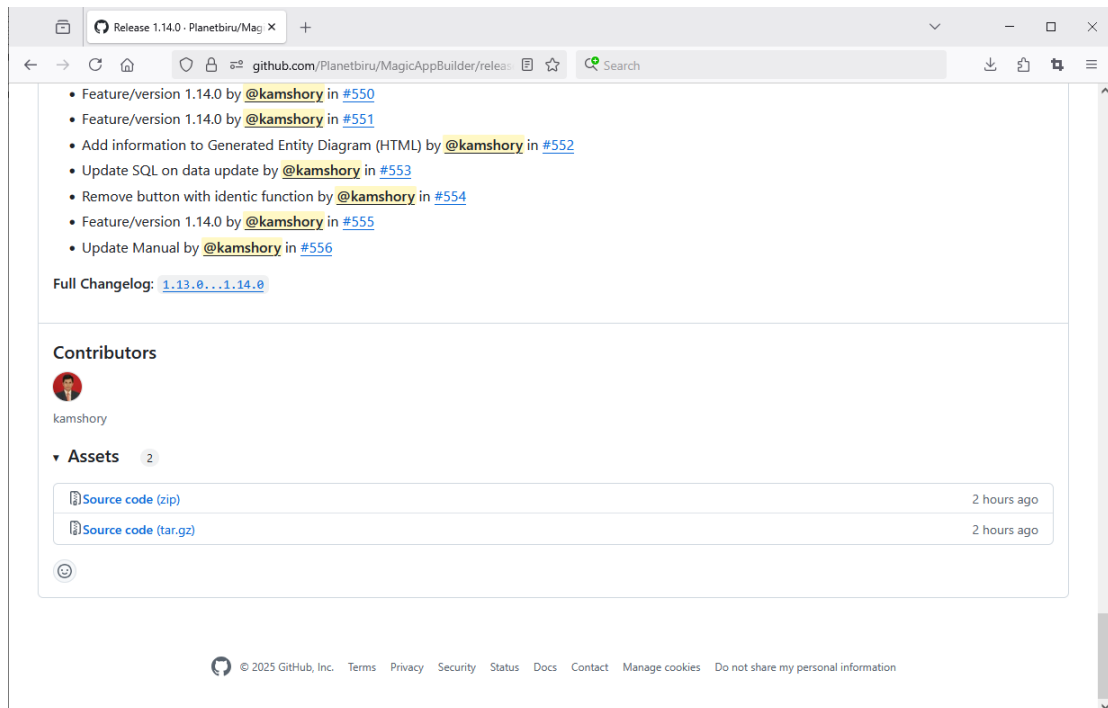


2. Unduh Rilis Terbaru

Navigasikan ke halaman rilis di:

<https://github.com/Planetbiru/MagicAppBuilder/releases>





Unduh file **Source code(zip)** dan simpan di penyimpanan lokal.

3. 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):** D:\xampp\htdocs

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

4. 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.

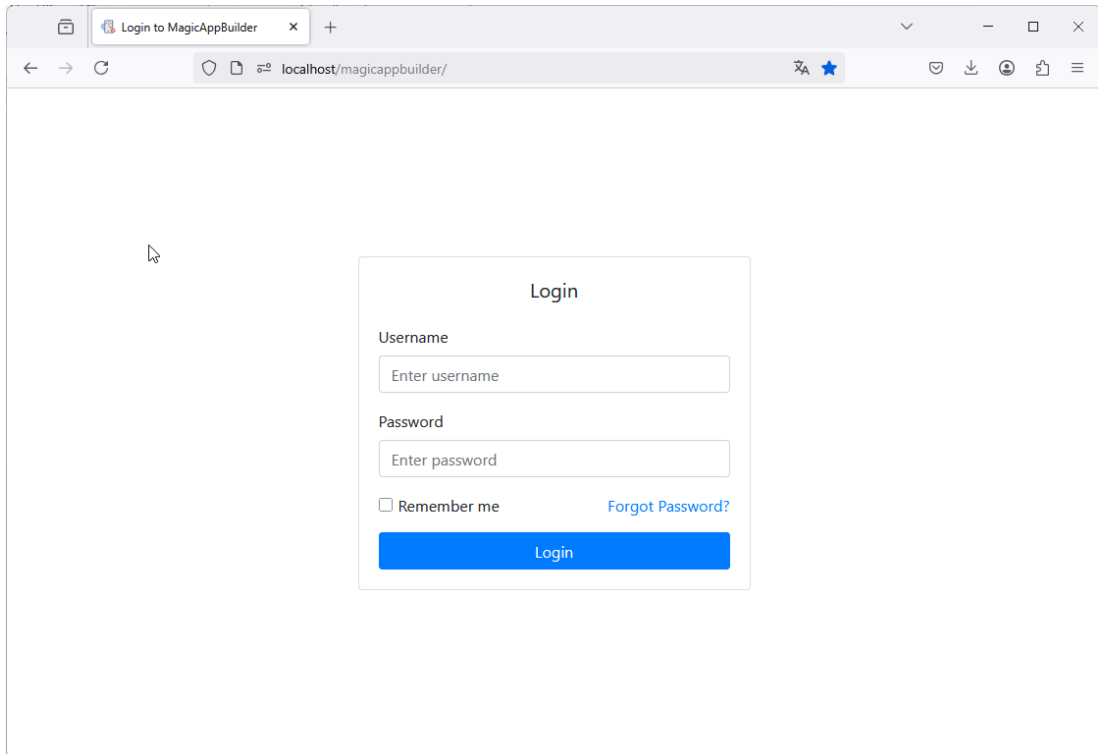
5. Akses MagicAppBuilder dari Browser

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.

Saat MagicAppBuilder diakses pertama kali, MagicAppBuilder akan memeriksa struktur database. Jika database masih kosong, MagicAppBuilder akan secara otomatis membuatnya dengan data-data default termasuk administrator level dan administrator account.



6. 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**.

Entity

Table: - Select Table -

Reload Table

Master Entity Name

Master Primary Key

Approval Table Name

Approval Primary Key

Approval Entity Name

Trash Table Name

Trash Primary Key

Trash Entity Name

Module Option

Target

Update Current Location Manage

Update Entity ☐ Yes

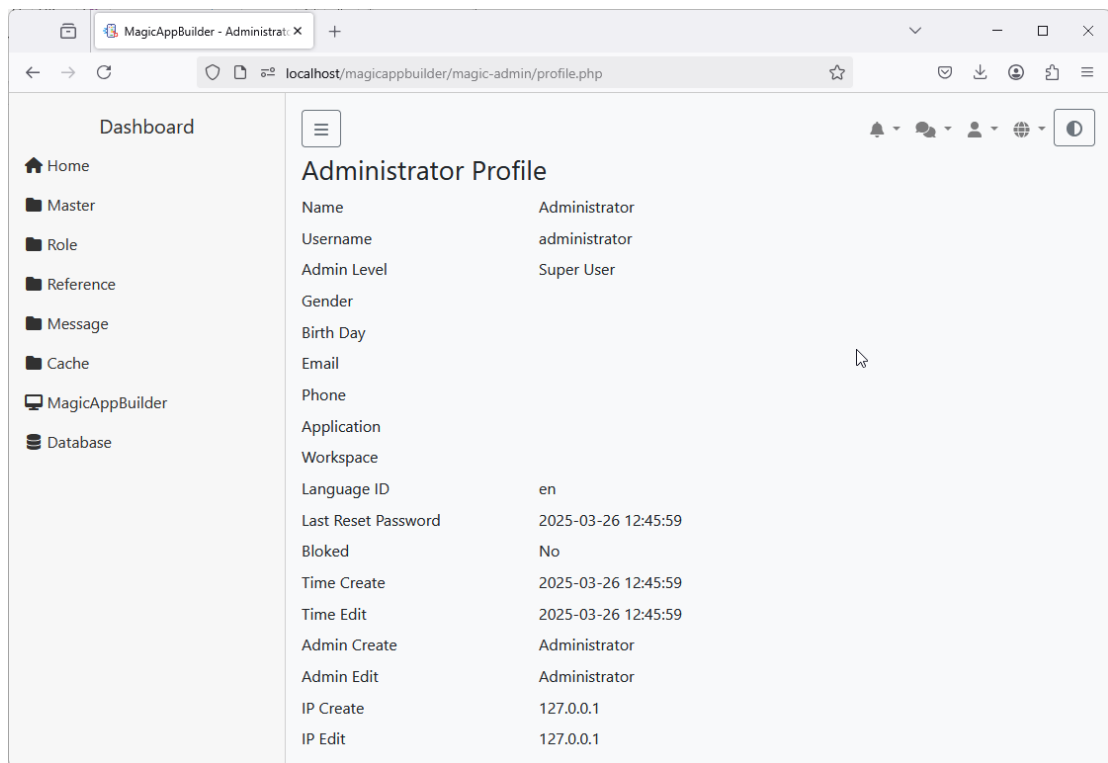
7. Setelah Login

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

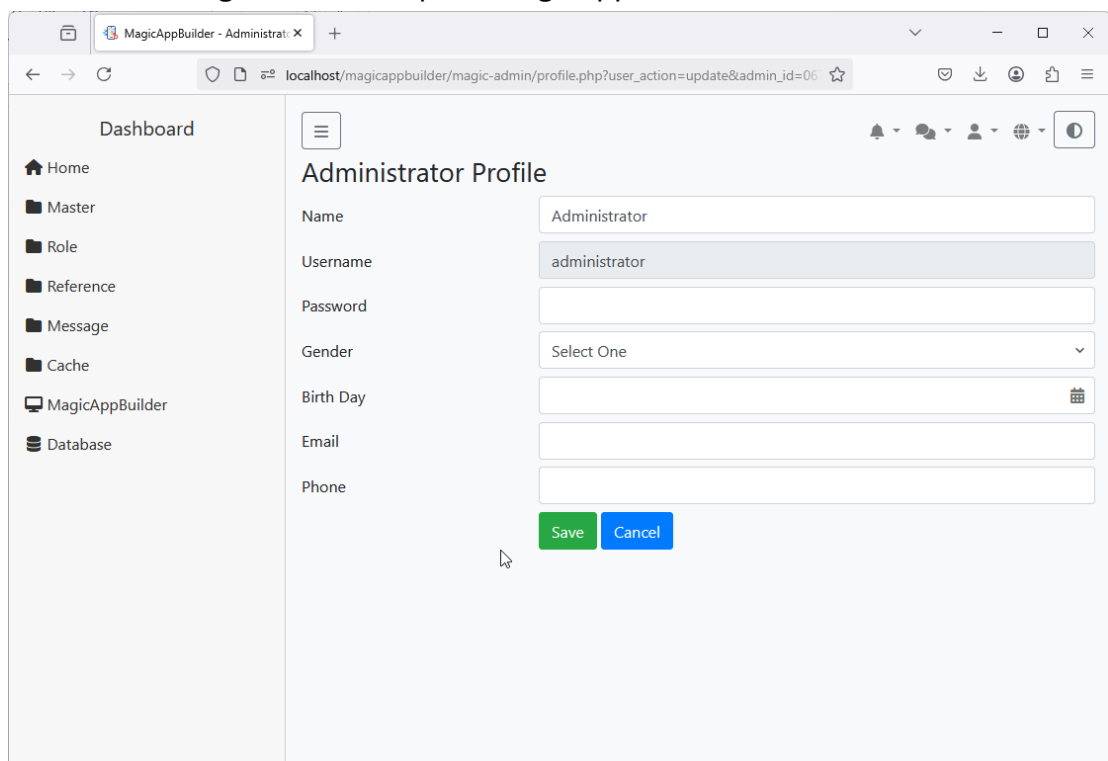
8. Mengganti Password

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.



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.



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

- Name
- Username
- Password
- Gender
- Birthday
- 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.

9. 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 pengaturan 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 dengan 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"
```

Setelah pengguna selesai menulis kode Yaml tersebut, kembali ke browser lalu klik reset password. Pastikan bahwa file **reset-password.yml** terhapus 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 MagicAppBuilder 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.

2. 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'
```

3. Simpan Perubahan

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

4. Verifikasi Koneksi Database

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

5. 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.

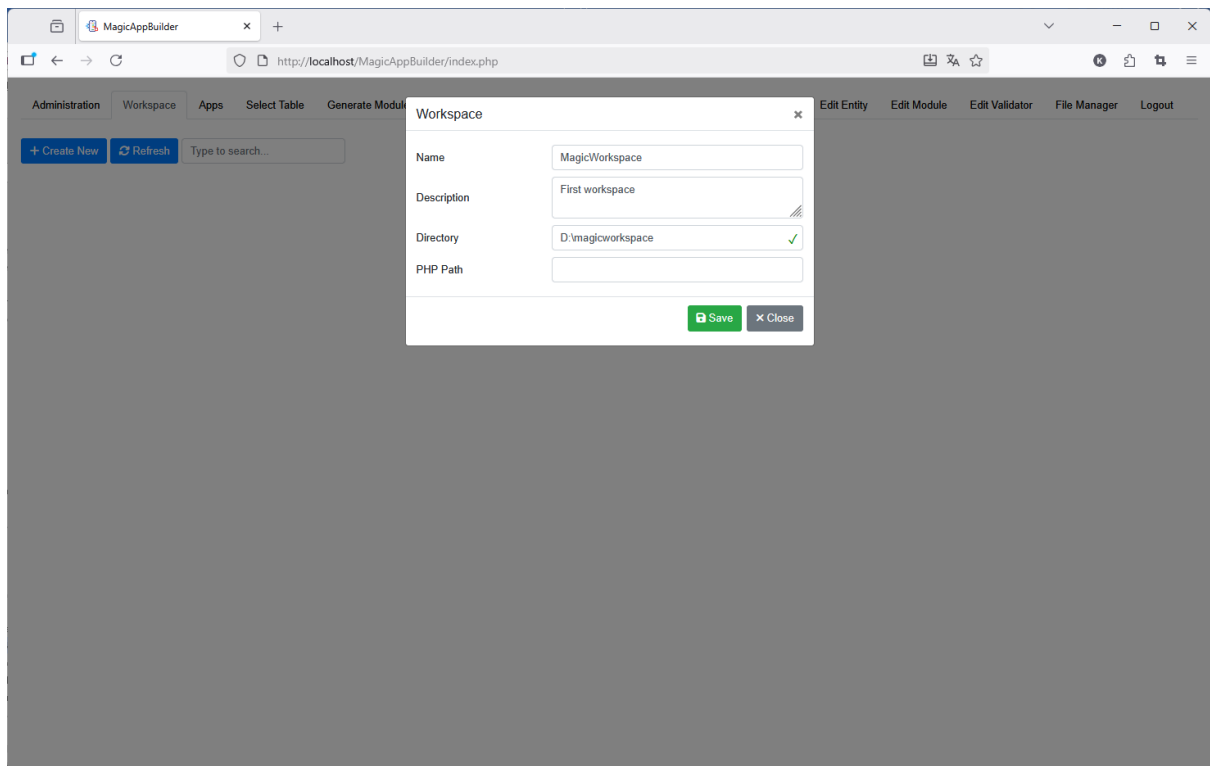
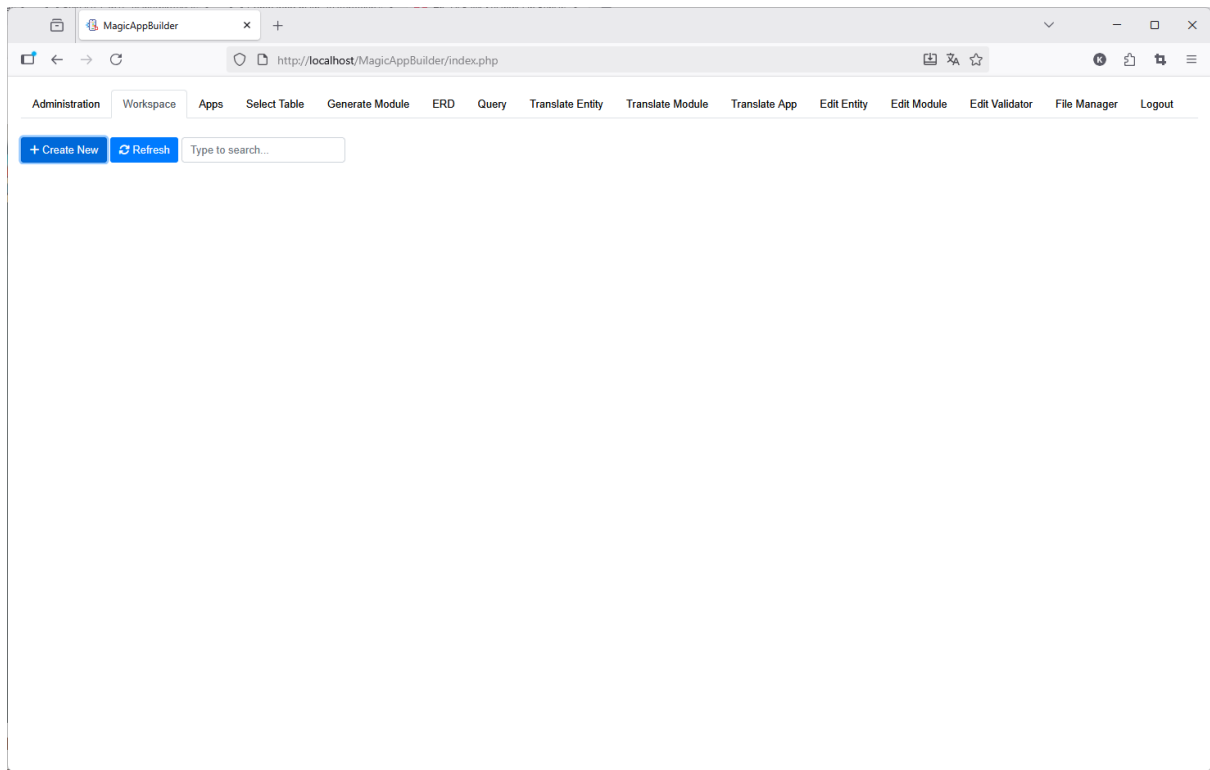
4.1.3 Langkah 3: Membuat Workspace

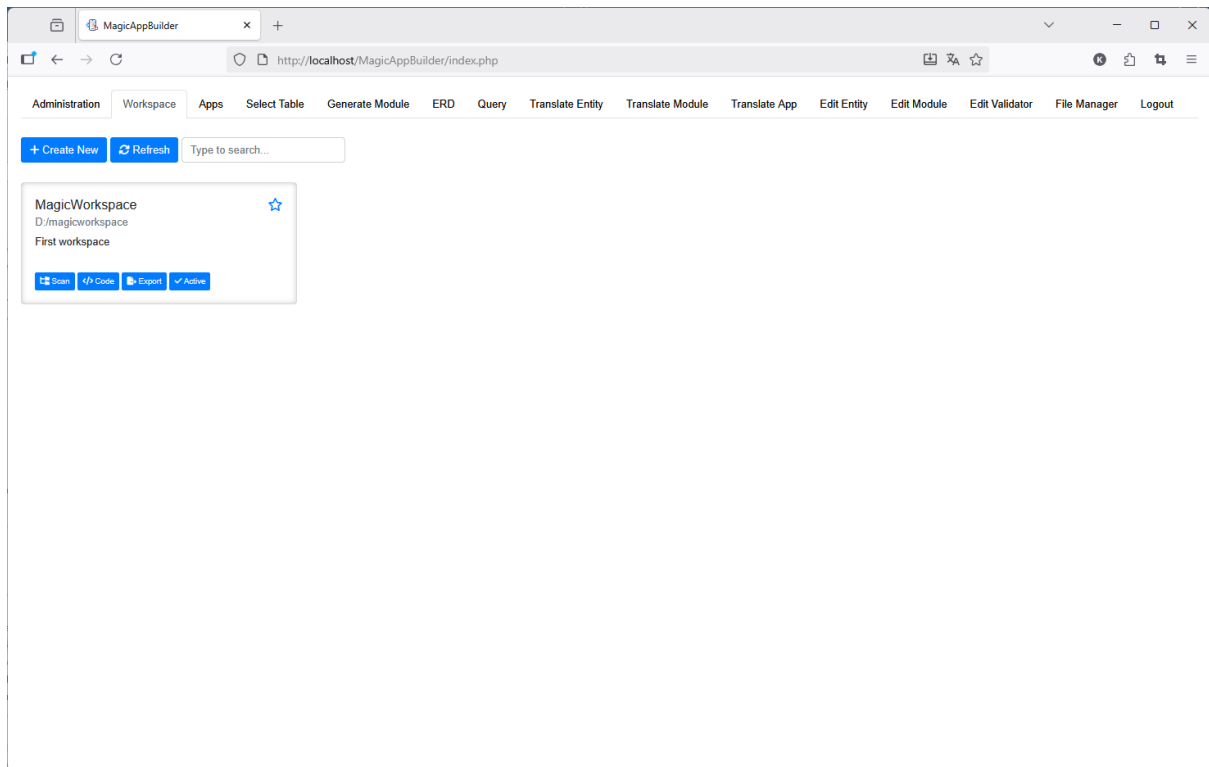
Setelah MagicAppBuilder berhasil diinstal, langkah berikutnya adalah **membuat Workspace**. Workspace adalah direktori yang berisi beberapa proyek. Setiap workspace terisolasi, memungkinkan pengelolaan akses admin yang lebih baik.

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**.





Di sebelah kanan nama workspace terdapat tanda bintang. Tanda bintang ini digunakan untuk memasukkan workspace ke favorit. Workspace favorit akan ditampilkan di paling atas dari daftar workspace. Ini sangat berguna terutama saat mengelola banyak workspace. Pengguna dapat mengeluarkan workspace dari favorit sewaktu-waktu.

Di bagian bawah dari kartu workspace ada beberapa tombol sebagai berikut:

1. Scan

Tombol Scan digunakan untuk memindai semua direktori yang ada di dalam direktori workspace. MagicAppBuilder akan memeriksa apakah direktori tersebut mengandung konfigurasi aplikasi atau tidak. Jika ya, MagicAppBuilder akan memeriksa apakah aplikasi tersebut sah atau tidak. MagicAppBuilder akan menambahkannya sebagai aplikasi jika direktori tersebut merupakan direktori aplikasi namun akan menandai apakah direktori source code ada atau tidak dan apakah aplikasi tersebut sah atau tidak. Pengguna dapat membangun ulang aplikasi yang tidak sah dari Application Option.

2. Validate

Tombol Validate digunakan untuk memeriksa keabsahan aplikasi yang terdaftar di database pada workspace. Berbeda dengan Scan, Validate hanya memeriksa keabsahan aplikasi yang tersimpan di database. Pengguna dapat membangun ulang aplikasi yang tidak sah dari Application Option.

3. Code

Tombol Code digunakan untuk membuka direktori workspace menggunakan Visual Studio Code.

4. Export

Tombol Export digunakan untuk mengekspor semua konfigurasi aplikasi dalam sebuah workspace. Source code tidak ikut diekspor.

5. Active

Tombol Active digunakan untuk menandai workspace mana yang aktif untuk pengguna. Tombol ini tidak mempengaruhi workspace yang aktif bagi pengguna lain.

4.5.1 Langkah 4: Membuat Aplikasi

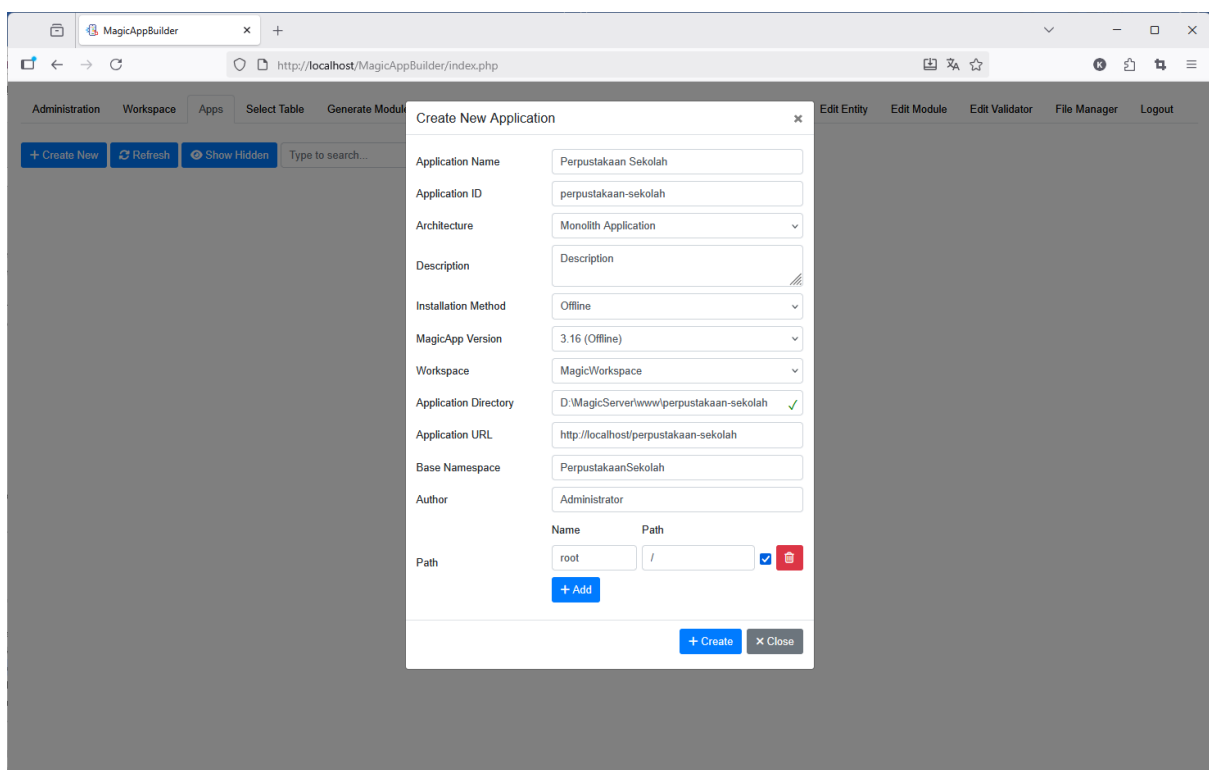
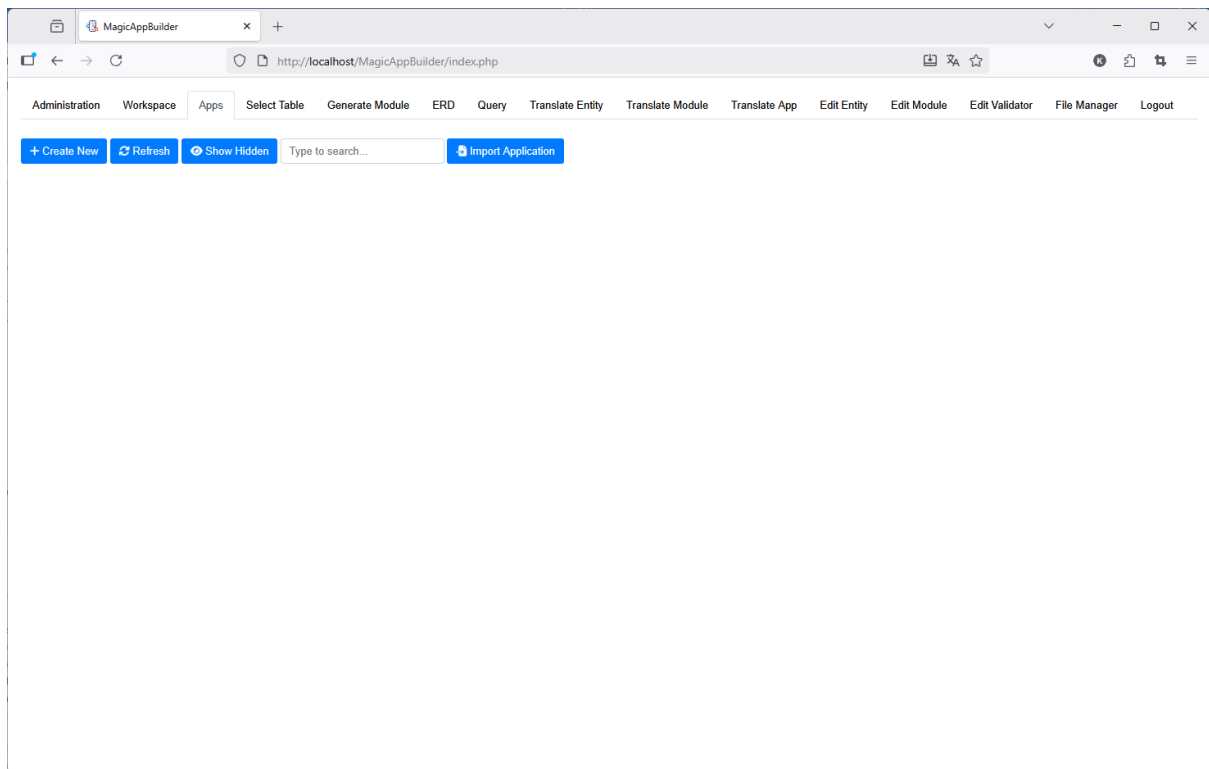
Setelah workspace dibuat, pengguna bisa melanjutkan ke langkah berikutnya, yaitu membuat aplikasi.

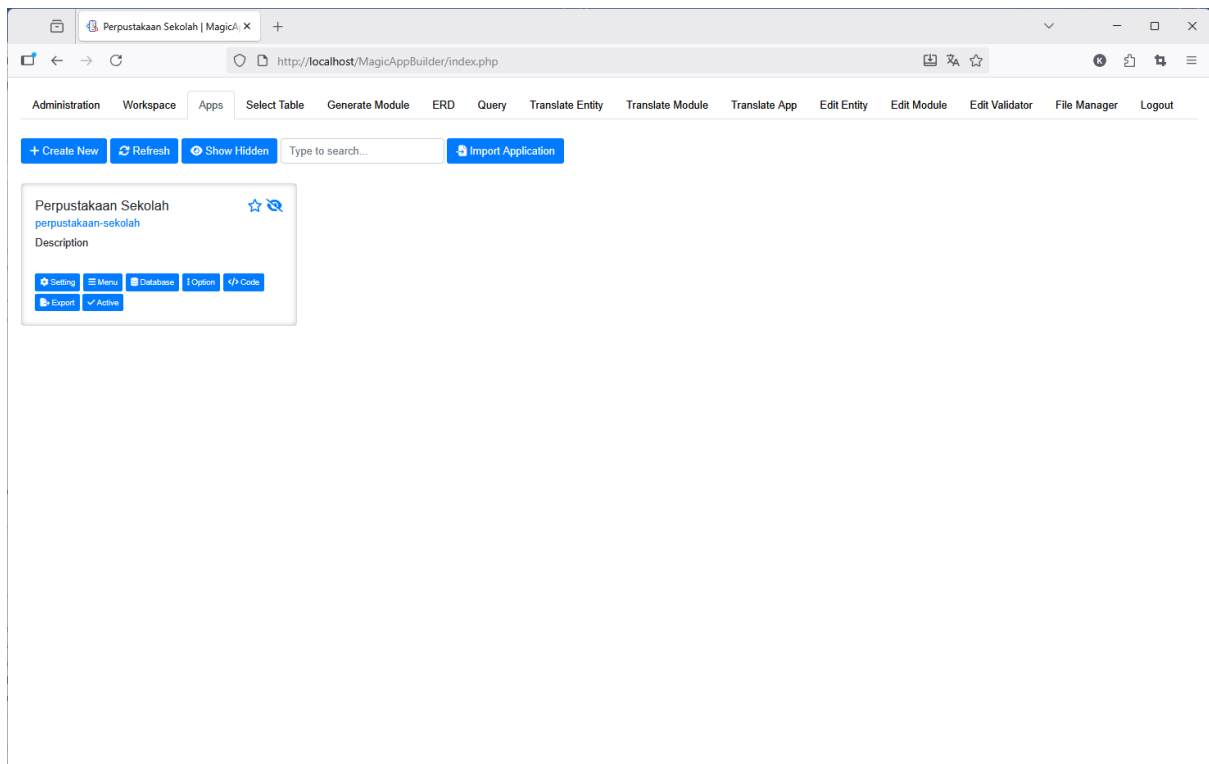
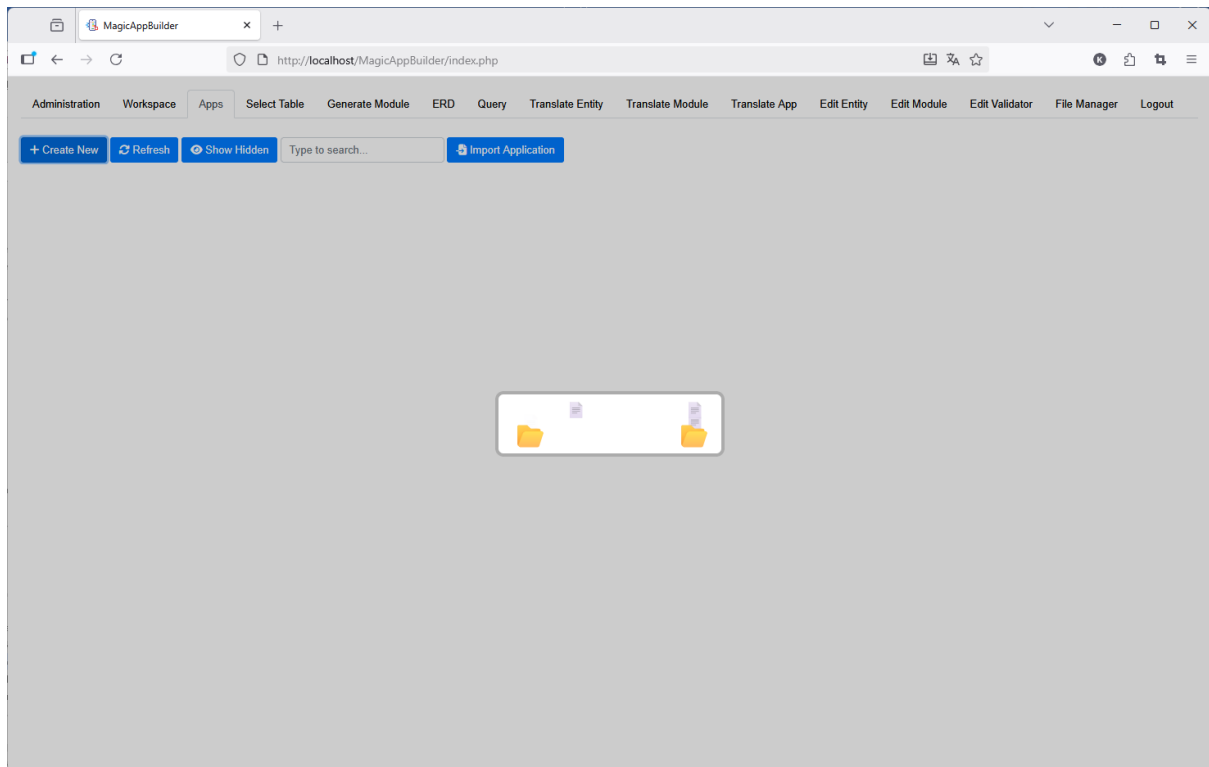
Untuk membuat aplikasi baru, navigasi ke tab **Apps**. MagicAppBuilder akan menampilkan halaman baru yang berisi dua tombol dan satu filter. Klik tombol **Create New**. MagicAppBuilder akan menampilkan sebuah dialog dengan formulir. Isi formulir dengan informasi berikut:

- **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 diperbarui 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.





Di sebelah kanan nama aplikasi terdapat tanda bintang. Tanda bintang ini digunakan untuk memasukkan aplikasi ke favorit. Aplikasi favorit akan ditampilkan di paling atas dari daftar aplikasi. Ini sangat berguna terutama saat mengelola banyak aplikasi. Pengguna dapat mengeluarkan aplikasi dari favorit sewaktu-waktu.

4.5.2 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.

- **Application Name:** adalah nama aplikasi.
- **Architecture:** adalah arsitektur aplikasi, yaitu Monolith Application dan Microservices Application.
- **Description:** adalah deskripsi aplikasi.
- **Application Directory:** adalah direktori aplikasi. File-file dalam direktori inilah yang nantinya akan menjadi file aplikasi yang dapat dengan aman dipindahkan ke server aplikasi di lingkungan produksi.
- **Application URL:** adalah URL aplikasi yang dapat diakses dari komputer pengguna pada fase pengembangan aplikasi.
- **Path:** adalah path di mana file-file modul akan dibuat. Sebuah aplikasi dapat terdiri dari banyak direktori yang akan menjadi prefix atau awalan path dari aplikasi dalam sudut pandang klien.
- **Multi-Level Menu:** adalah pilihan apakah aplikasi akan menggunakan menu dengan tingkatan tanpa batas. Pilihan ini dapat diubah sewaktu-waktu.
- **Active Theme:** adalah tema yang digunakan aplikasi sesuai dengan pilihan tema yang tersedia. Pilihan ini dapat diubah sewaktu-waktu.

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.

- **Driver:** Menentukan jenis database driver yang akan digunakan (misalnya, sqlite, mysql, pgsql).
- **Host:** adalah alamat IP atau nama domain dari server database (misalnya, 127.0.0.1 atau localhost). Ini adalah parameter yang paling penting untuk koneksi ke server selain SQLite.
- **Database File Path:** adalah path file database yang digunakan oleh SQLite. Parameter ini tidak diperlukan untuk MySQL atau PostgreSQL.
- **Port:** adalah nomor port untuk koneksi ke server database (standar: MySQL 3306, PostgreSQL 5432).
- **Username:** adalah nama pengguna untuk otentikasi ke database.

- **Password:** Kata sandi yang terkait dengan nama pengguna.
- **Database Name:** adalah nama database yang akan diakses di dalam server.
- **Schema:** adalah skema database yang akan digunakan (terutama penting untuk PostgreSQL).
- **Time Zone:** adalah zona waktu aplikasi untuk koneksi database, memastikan konsistensi waktu dengan aplikasi.
- **Time Zone System:** adalah zona waktu yang digunakan oleh server database. Parameter ini digunakan pada database SQLite dan SQL Server untuk melakukan konversi waktu antar pengguna yang berada di daerah waktu yang berbeda.

3. Account Security

- **Algorithm:** digunakan untuk mengatur algoritma hash dari password. Tersedia pilihan sha1, sha256, sha384, sha512 dan md5. Di lingkungan produksi, pengguna dapat menggunakan algoritma yang lain.
- **Salt:** digunakan untuk menambahkan plain text sebelum dilakukan operasi hash. Salt akan meningkatkan keamanan password pengguna.

4. 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**.

- **Session Name:** adalah nama parameter dari cookie yang digunakan untuk menyimpan ID session. Baik nama maupun nilai cookie diberikan oleh server untuk dikirimkan ulang oleh klien pada akses berikutnya.
- **Session Life Time:** adalah lama hidup dari sesi. Waktu ini juga digunakan oleh server untuk mengatur waktu hidup dari cookie dengan nama yang sesuai pada session name. Sebagai contoh: jika pengguna mengatur session life time dengan angka **1440**, maka server akan menghapus session setelah 24 menit dan browser juga akan menghapus cookie yang berisi ID session setelah 24 menit terhitung sejak akses terakhir yang memperbarui umur baik session maupun cookie.

Perlu diingat bahwa PHP mempunyai mekanisme garbage collector untuk session. Jika pengguna mengatur lifetime selama 24 jam namun di php.ini diset 24 menit, session dengan lifetime 24 jam bisa saja dihapus oleh proses lain yang menggunakan lifetime 24 menit setelah umurnya mencapai 24 menit.

Ada beberapa solusi yang dapat dilakukan yaitu:

1. Mengubah konfigurasi php file php.ini
Atur ke 24 jam atau waktu terlama yang diinginkan


```
session.gc_maxlifetime = 86400 ; // 24 jam dalam detik  
session.cookie_lifetime = 86400 ; // 24 jam dalam detik
```

2. Mengganti lokasi penyimpanan file session

Set Sassion Save Path ke lokasi lain yang tidak sama dengan lokasi yang ditunjuk di file php.ini

3. Menggunakan penyimpanan alternatif

Gunakan redis atau sqlite alih-alih files

- **Session Save Handler:** ada 3 pilihan yaitu **files**, **redis** dan **sqlite**.

Pilihan Session Save Handler akan mempengaruhi arti dari Sassion Save Path yang digunakan.

- **Session Save Path:** adalah path di mana data session disimpan.

Jika pengguna memilih **files**, pengguna dapat mengatur session save path dengan direktori di mana file-file session disimpan. Jika pengguna melakukan horizontal scaling, maka pengguna wajib menggunakan *Network Attached Storage* (NAS) untuk menyimpan file-file session di satu server sehingga jika pengguna masuk dari satu server, tidak diminta masuk lagi saat akses dialihkan ke server lain.

Jika pengguna memilih **redis**, pengguna wajib menuliskan URL dari Redis, misalnya `tcp://anydomain.tld:6379?auth=anypassword` di mana:

anydomain.tld adalah nama host yang dapat diakses dari server aplikasi.

Nama host juga dapat dihanti dengan Alamat Internet Protocol (IP Address),

6379 adalah nomor port TCP yang digunakan oleh server Redis,

anypassword adalah kode keamanan Redis. Jika Redis tidak memiliki kode keamanan, cukup tulis dengan **tcp://anydomain.tld:6379**. Jika pengguna

penulis dengan **tcp://anydomain.tld** saja, MagicAppBuilder akan

menganggap bahwa server Redis menggunakan port 6379. Penggunaan

Redis di localhost tanpa password dapat ditulis dengan salah satu dari

pilihan berikut:

1. `tcp://localhost`
2. `tcp://127.0.0.1`
3. `tcp://::1`

Jika pengguna memilih **sqlite**, ekstensi **sqlite** di PHP wajib ada dan aktif.

Session Save Path adalah file database dari SQLite. Path ini harus berjenis

file dan harus bisa dibaca dan ditulis oleh PHP. Jika file belum pernah ada,

ekstensi **sqlite** pada PHP akan membuat file baru saat PHP membuat menginisiasi objek PDO ke file tersebut.

PERHATIAN!

Jangan menyimpan session di database aplikasi demi keamanan.

- **Cookie Path:** adalah path dari aplikasi di mana cookie dengan nama yang sesuai pada Session Name berlaku. Jika ingin agar cookie berlaku di mana saja, gunakan tanda garis miring (/) tanpa karakter apapun.

- **Cookie Domain:** adalah domain di mana cookie berlaku. Jika ingin agar cookie berlaku di semua subdomain dari aplikasi, beri awalan titik (.) pada nama domain. Misalnya nama domainnya adalah “anydomain.tld”, maka atur Cookie Domain dengan “.anydomain.tld”.
- **Cookie Secure:** adalah pilihan agar cookie berlaku pada HTTPS saja. Browser tidak akan mengirimkan cookie kembali ke server jika pengguna mengaksesnya tanpa HTTPS.
- **Cookie HTTP Only:** adalah pilihan untuk mencegah pengguna mengambil, memanipulasi, dan menghapus cookie dari JavaScript atau konsol browser.
- **Cookie Same Site:** adalah pilihan keamanan untuk mengirimkan cookie ke domain lain.

1. SameSite=Strict

Ini adalah tingkat keamanan tertinggi. Cookie hanya akan dikirim jika permintaan datang dari situs web yang **benar-benar sama** (memiliki domain yang sama persis).

- **Contoh:** Anda login ke bank.com. Saat Anda melakukan transfer, cookie akan dikirim. Namun, jika Anda membuka tab baru dan mengklik tautan dari situs-lain.com yang mengarahkan ke bank.com, cookie tidak akan dikirim. Ini secara efektif mencegah serangan CSRF.

2. SameSite=Lax

Ini adalah tingkat keamanan yang seimbang dan paling umum digunakan. Cookie akan dikirim dalam permintaan yang aman seperti saat Anda melakukan navigasi normal (misalnya, mengklik tautan). Namun, cookie tidak akan dikirim jika permintaan datang dari metode yang berpotensi berbahaya seperti POST dari domain lain.

- **Contoh:** Anda mengklik tautan dari google.com ke toko-online.com. Cookie Anda akan ikut terkirim. Namun, jika ada formulir tersembunyi di situs-lain.com yang mencoba mengirim data POST ke toko-online.com, cookie Anda tidak akan dikirim.

3. SameSite=None

Ini adalah tingkat keamanan terendah, yang berarti cookie akan selalu dikirim, bahkan dalam permintaan lintas-situs (cross-site). Untuk menggunakan SameSite=None, cookie harus memiliki atribut **Secure** (yang berarti hanya dikirim melalui koneksi HTTPS yang aman).

- **Contoh:** Digunakan untuk layanan yang memang dirancang untuk berinteraksi dengan situs lain, seperti widget pihak ketiga (misalnya tombol "Bagikan" di Facebook) atau iklan.

5. Reserved Columns

Terdapat bagian penting yang disebut **Reserved Columns** (Kolom yang Direservasi) yang harus dikonfigurasi sejak awal. **Kolom ini tidak boleh 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.

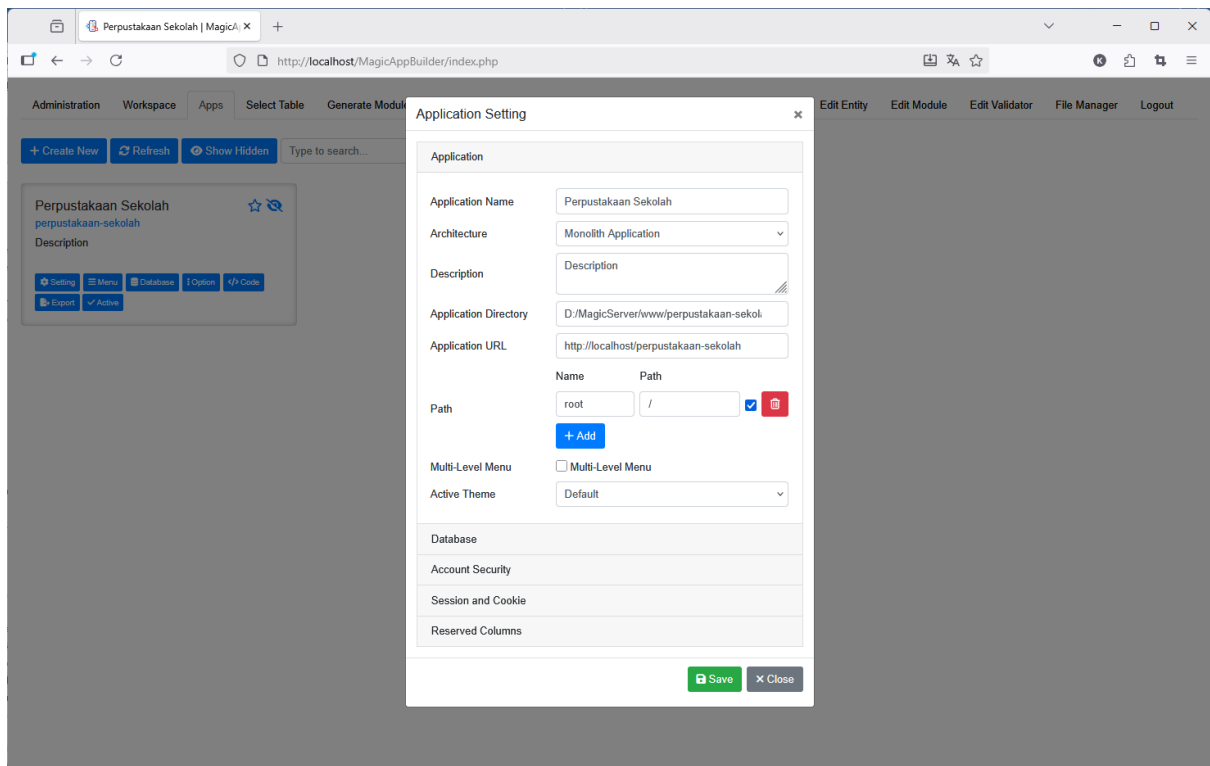
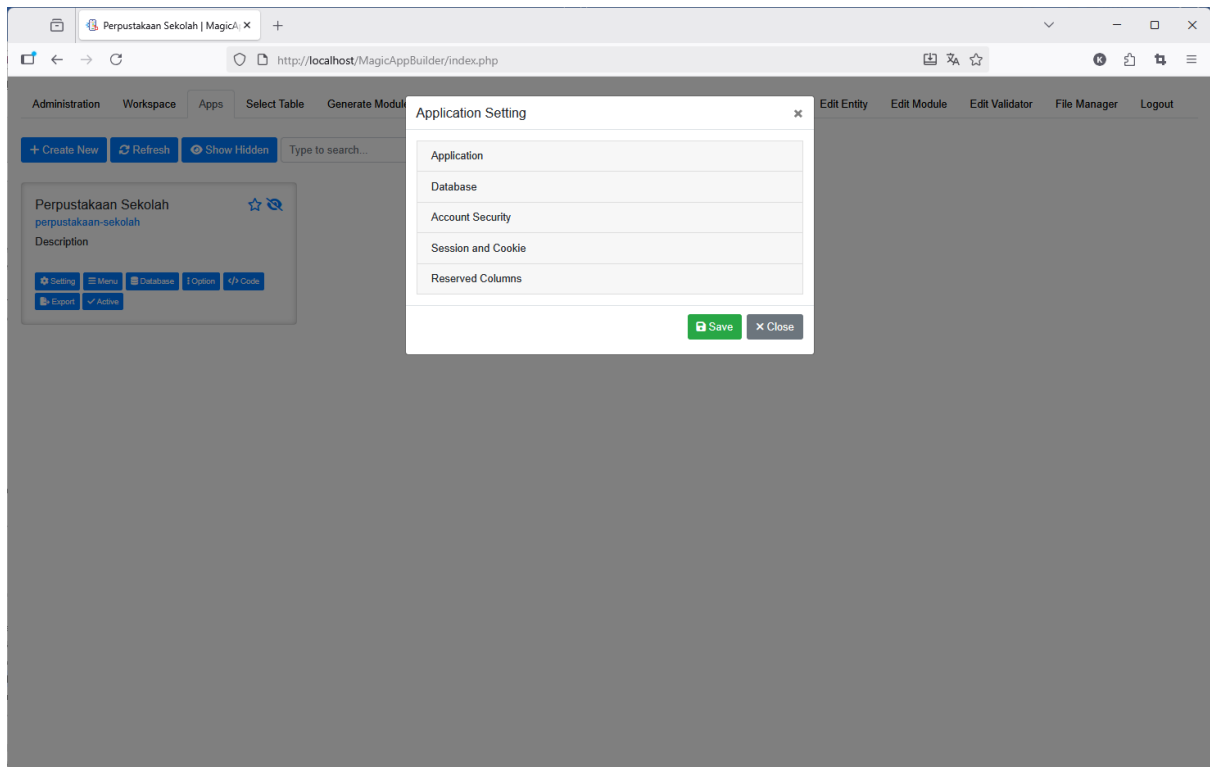
2. Entitas Artis

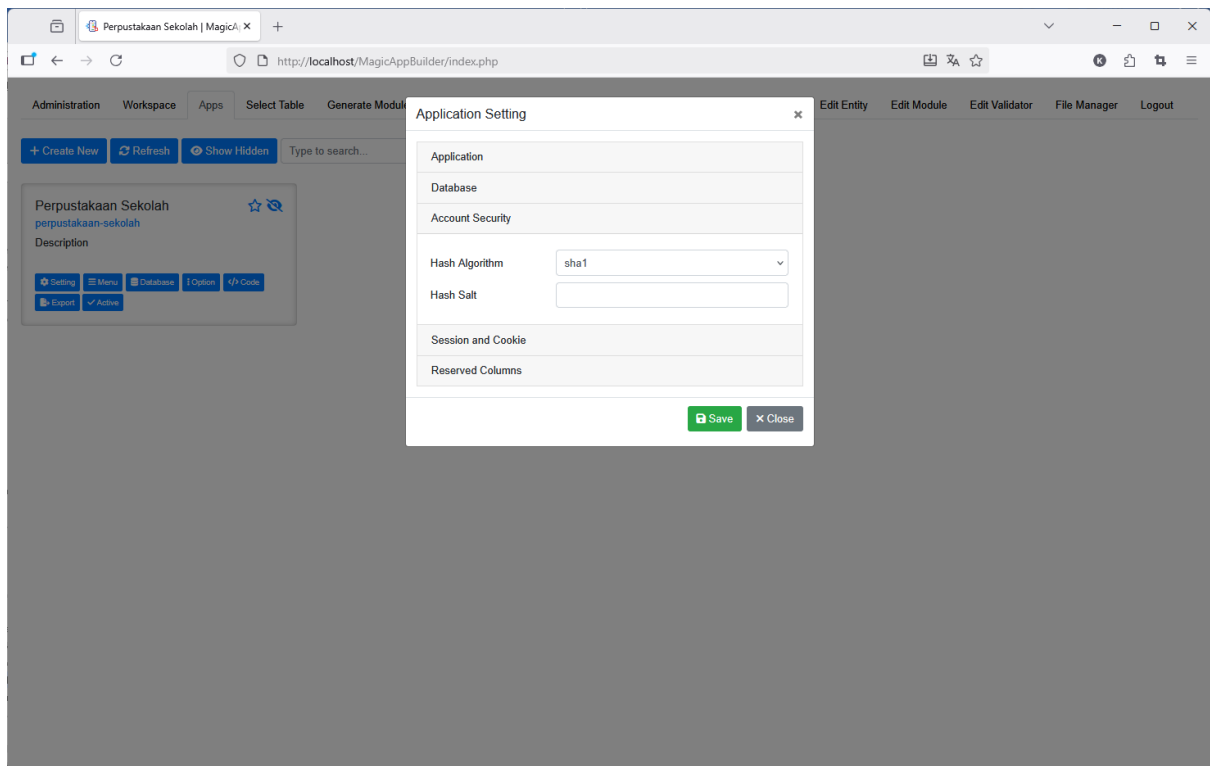
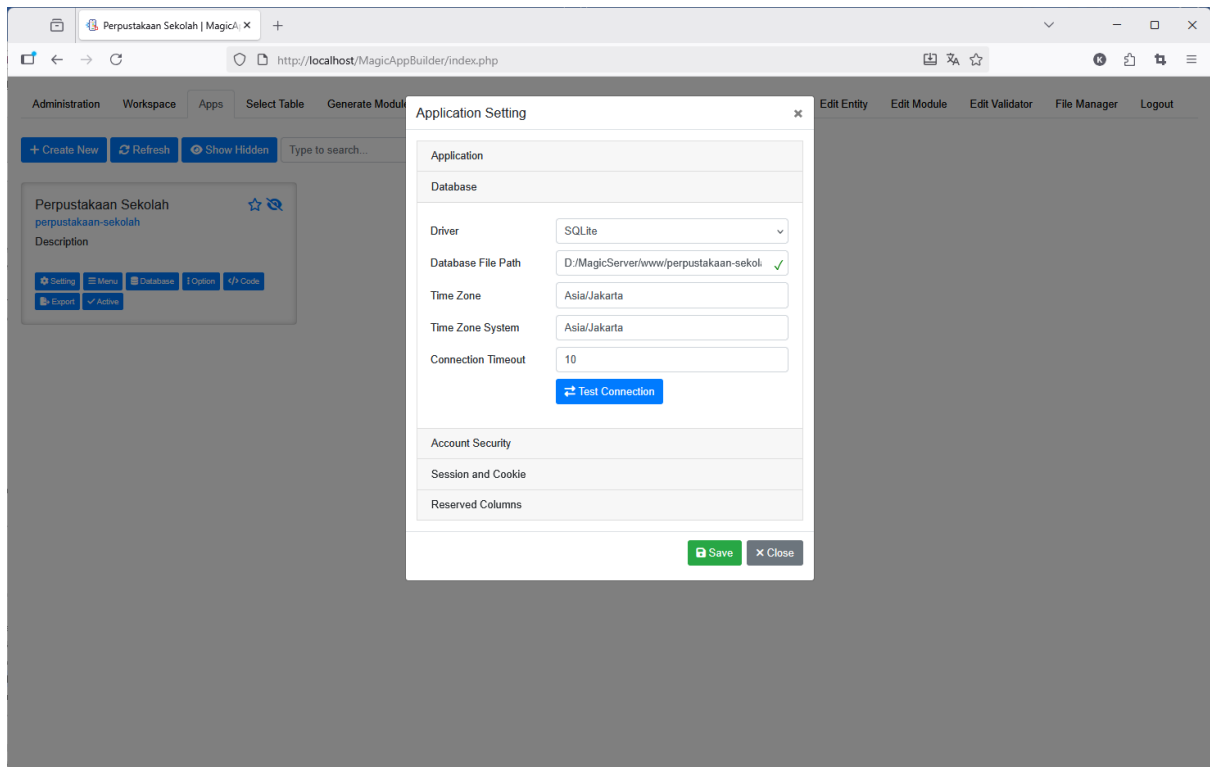
- **Tidak memerlukan** kolom **sort_order** karena data artis tidak perlu diurutkan secara default oleh pengguna.
- **Tetap menggunakan** kolom **active** untuk mengaktifkan dan menonaktifkan artis.

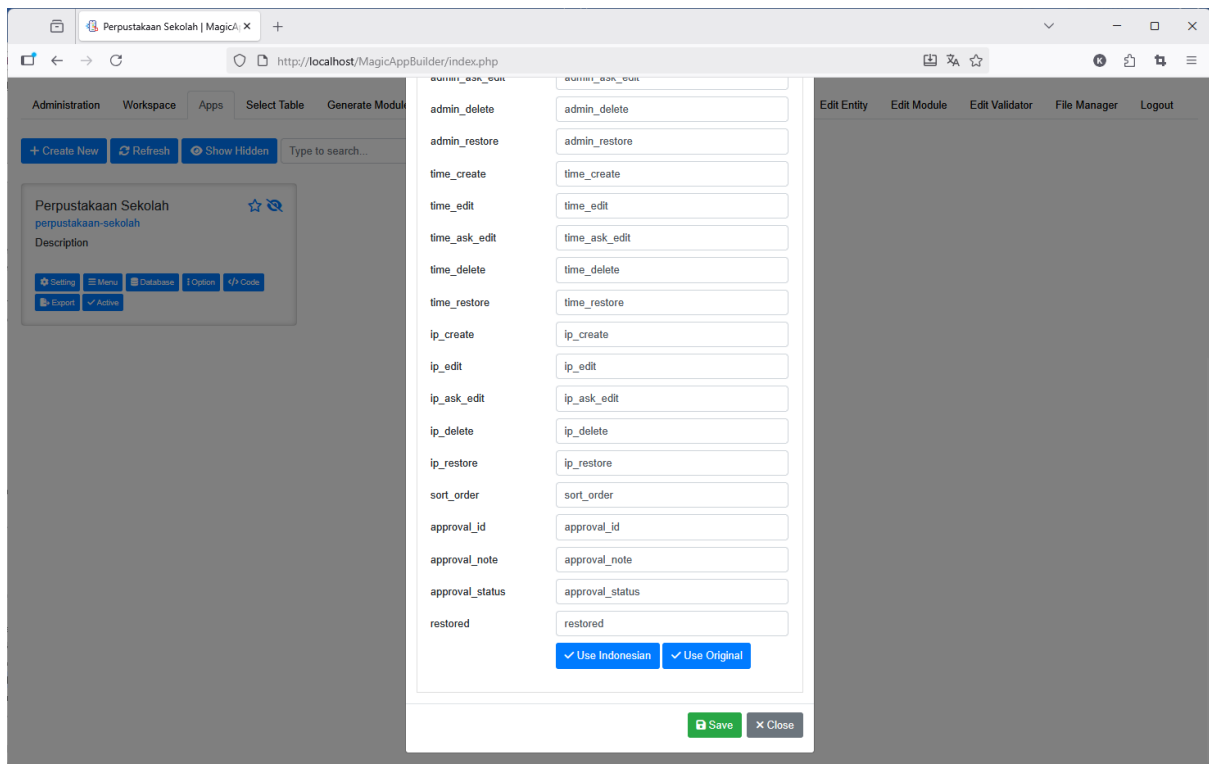
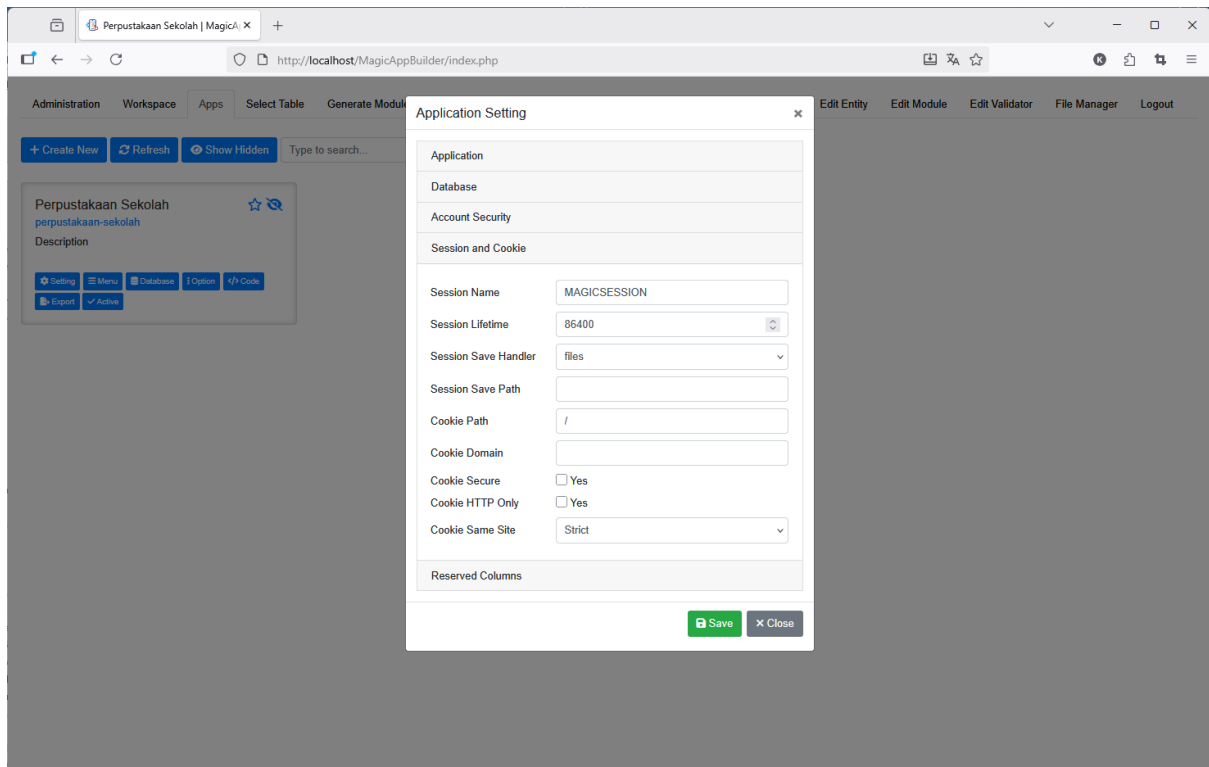
Pemetaan Nama Kolom ke Bahasa Indonesia

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.

4.5.3 Langkah 6: Mempersiapkan Struktur Database

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

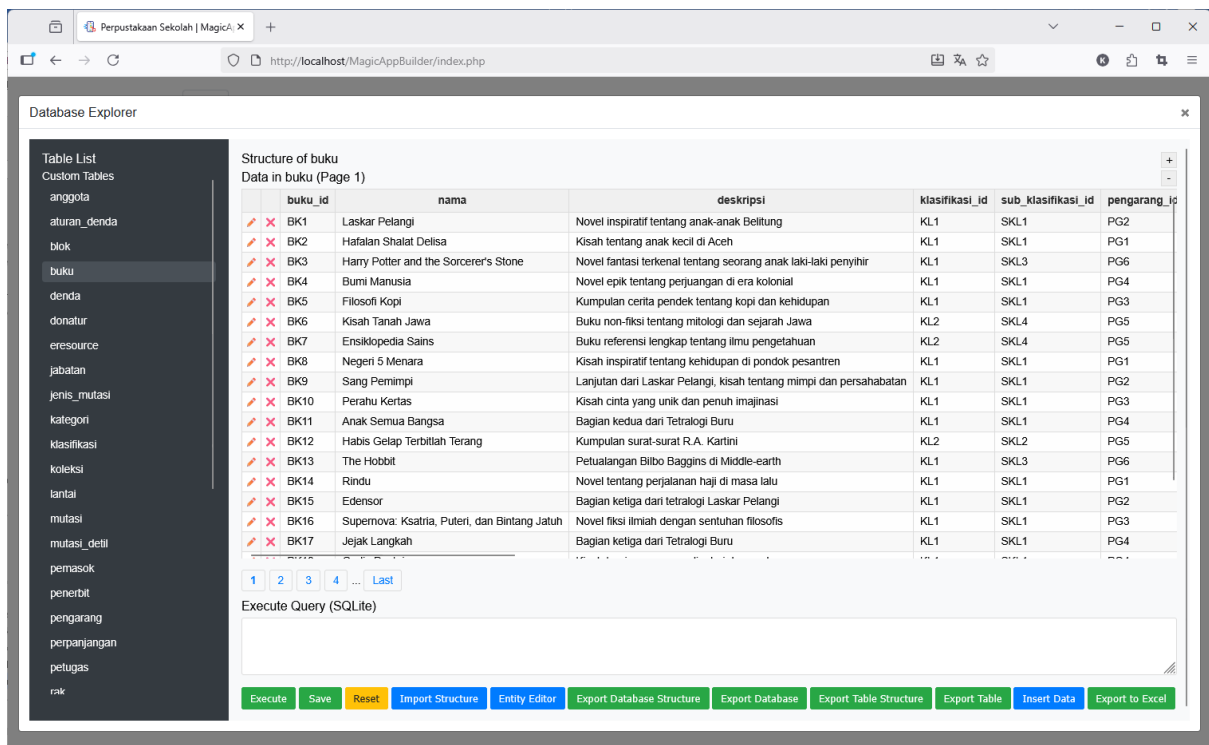
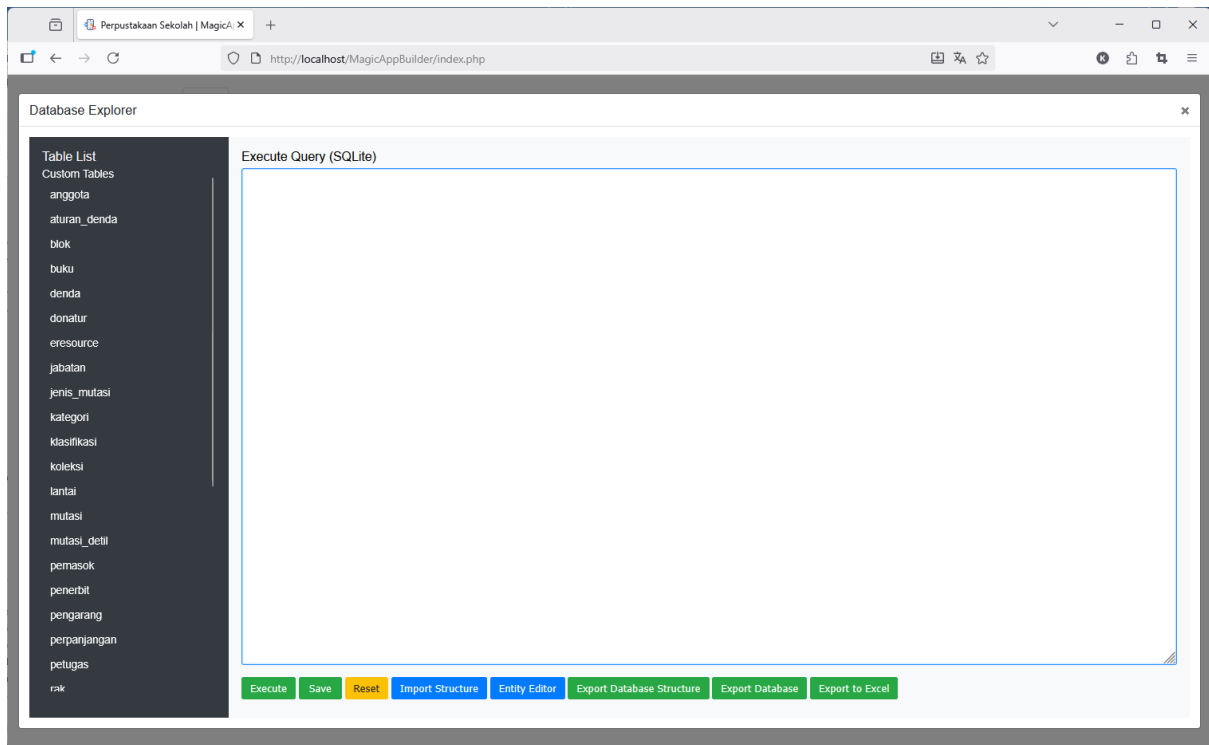
1. Database Explorer
2. Entity Editor

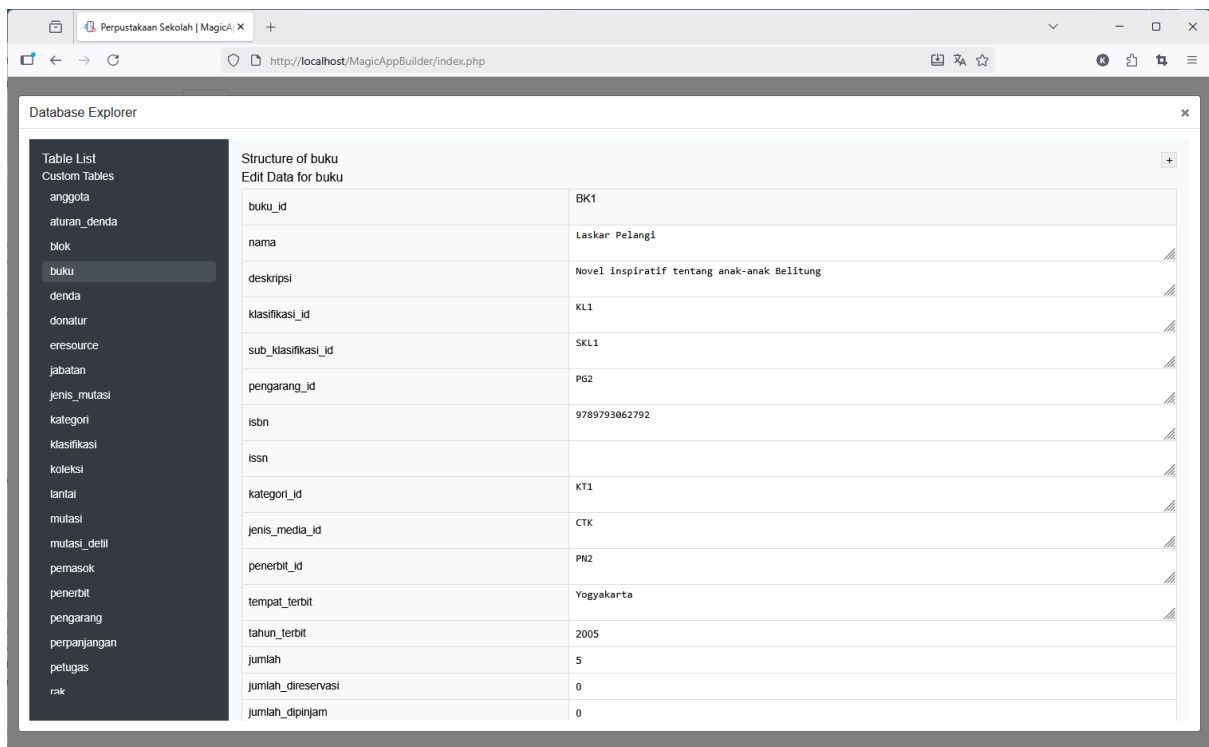
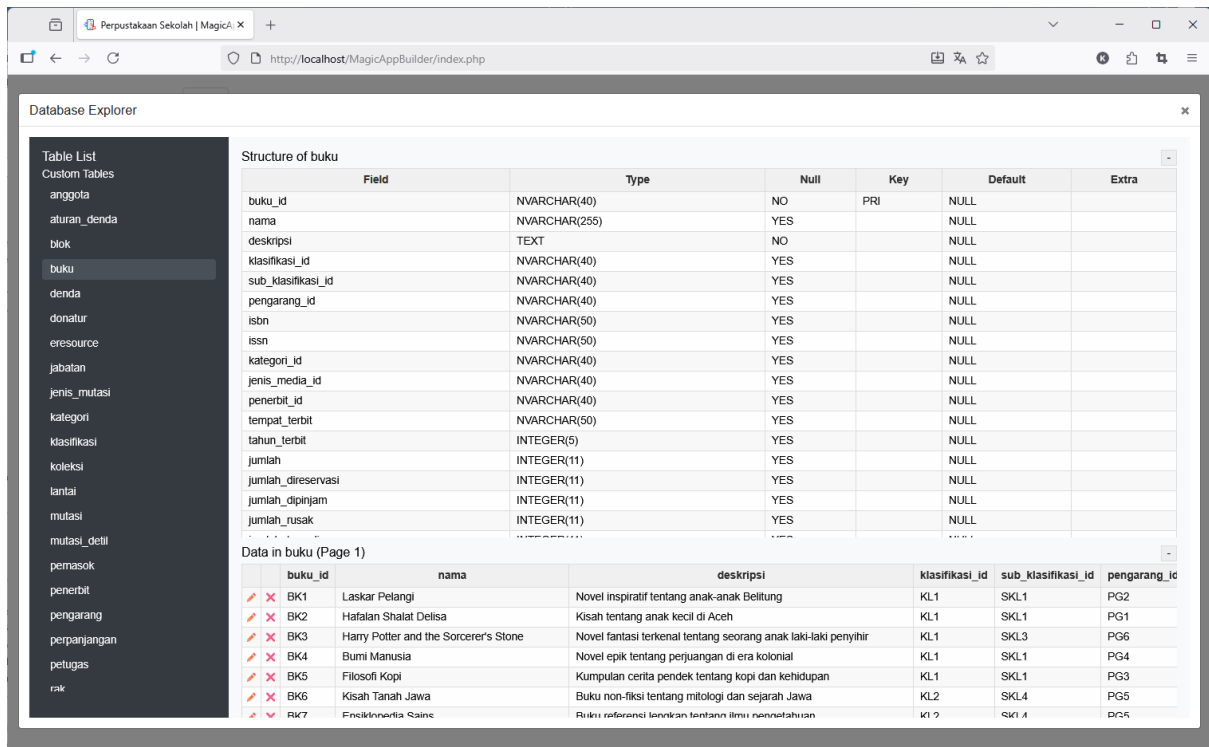
Pengguna juga dapat menggunakan aplikasi pihak ketiga untuk membuat struktur database jika lebih disukai.

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.





Bagian-Bagian Database Explorer

a. Sidebar

Sidebar berada di sebelah kiri halaman. Pada database MySQL dan MariaDB, pengguna dapat memilih database pada server. Pengguna dapat menjalankan query pada database yang dipilih. Jika Database Explorer dibuka dari tab Query

MagicAppBuilder, pengguna tidak dapat memilih database. Database yang dibuka akan sesuai dengan konfigurasi aplikasi.

Di sidebar terdapat daftar tabel dari database yang sedang dibuka. Tabel dikelompokkan menjadi dua yaitu System Tables dan Custom Tables. System Tables berisi tabel-tabel bawaan MagicAppBuilder sedangkan Custom Tables berisi tabel-tabel yang dibuat oleh pengguna. Custom Tables berada di atas System Tables.

b. Main Bar

Main bar adalah bagian utama dari Database Explorer. Saat sebuah tabel dipilih, Database Explorer akan menampilkan struktur tabel dan isi table jika ada. Pengguna dapat melakukan beberapa operasi pada sebagai berikut:

1. Membuat baris baru
2. Mengubah baris
3. Menghapus baris

Di bagian bawah terdapat sebuah text area yang dapat diisi dengan query yang akan dieksekusi. Di bawah text area terdapat beberapa tombol sebagai berikut:

1. Execute: digunakan untuk mengeksekusi query pada text area di atasnya.
2. Save: digunakan untuk menyimpan query sebagai file. Ini sangat penting untuk dokumentasi.
3. Reset: untuk mereset isi text area
4. Import Structure: digunakan untuk mengimpor struktur tabel dari database lain seperti MySQL, PostgreSQL, SQLite, SQL Server dan file database SQLite dalam bentuk file biner. Query yang diimpor akan disesuaikan dengan database yang sedang dibuka.
5. Entity Editor: untuk membuka entity editor yang akan dijelaskan di bagian terpisah.
6. Export Database Structure: digunakan untuk mengekspor struktur database tanpa data dan tanpa perubahan. Hasil ekspor berupa query **CREATE TABLE** sesuai dengan database yang sedang dibuka.
7. Export Database: digunakan untuk mengekspor database ke database lain yaitu: MySQL, MariaDB, PostgreSQL, SQLite dan SQL Server beserta dengan isinya. Saat tombol ini dipilih, Database Explorer akan menampilkan sebuah dialog yang berisi tabel dari database yang sedang dibuka. Pengguna dapat memilih apakah akan mengekspor strukturnya saja, isinya saja, atau keduanya. Pilihan tersebut dapat diterapkan pada setiap tabel.

Tabel dibagi dua kelompok yaitu Custom Tables dan System Tables. Custom tables adalah tabel-tabel yang dibuat oleh pengguna sesuai dengan kebutuhan bisnis, sedangkan system tables adalah tabel-tabel bawaan dari MagicAppBuilder.

Database Explorer akan mengekspor database secara bertahap sehingga memungkinkan untuk mengekspor database dengan ukuran besar. Hasil ekspor berupa query **CREATE TABLE** dan **INSERT INTO** sesuai dengan database tujuan.

8. Export Table Structure: digunakan untuk mengekspor strktur tabel tanpa data dan tanpa perubahan. Hasil ekspor berupa query **CREATE TABLE** sesuai dengan database yang sedang dibuka. Tombol ini hanya tersedia saat sebuah tabel dibuka.
9. Export Table Structure: digunakan untuk mengekspor strktur tabel beserta data tanpa perubahan. Hasil ekspor berupa query **CREATE TABLE** dan **INSERT INTO** sesuai dengan database yang sedang dibuka. Tombol ini hanya tersedia saat sebuah tabel dibuka.
10. Export to Excel: digunakan untuk mengekspor isi database ke format Excel. Setiap tabel akan disimpan di dalam sheet terpisah. Fitur ini berguna untuk melakukan pemrosesan data-data tabel yang diperlukan pada fase pengembangan aplikasi. Perlu dicatat bahwa sheet tanpa data tidak akan bisa diimpor oleh MagicAppBuilder.
11. Insert Data: digunakan untuk menambah baris baru pada tabel yang sedang dibuka. Saat tombol ini dipilih, Database Explorer akan menampilkan sebuah formulir sesuai dengan struktur tabel yang sedang dipilih.

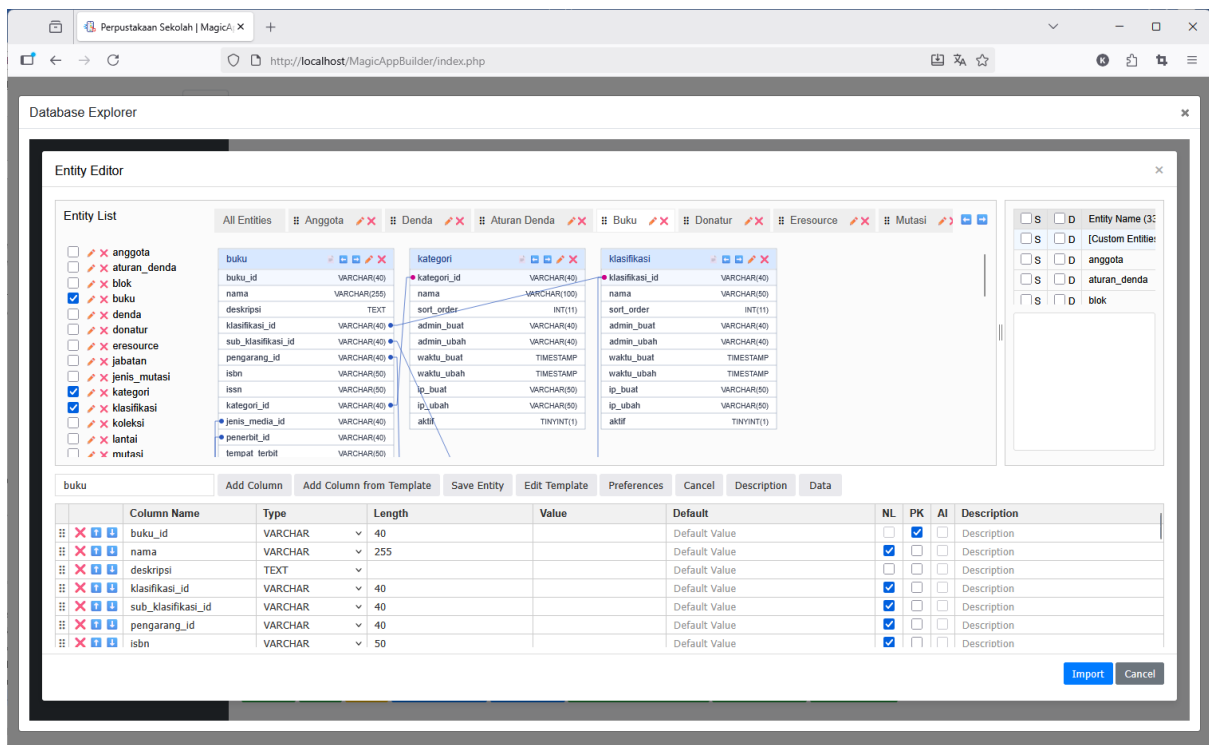
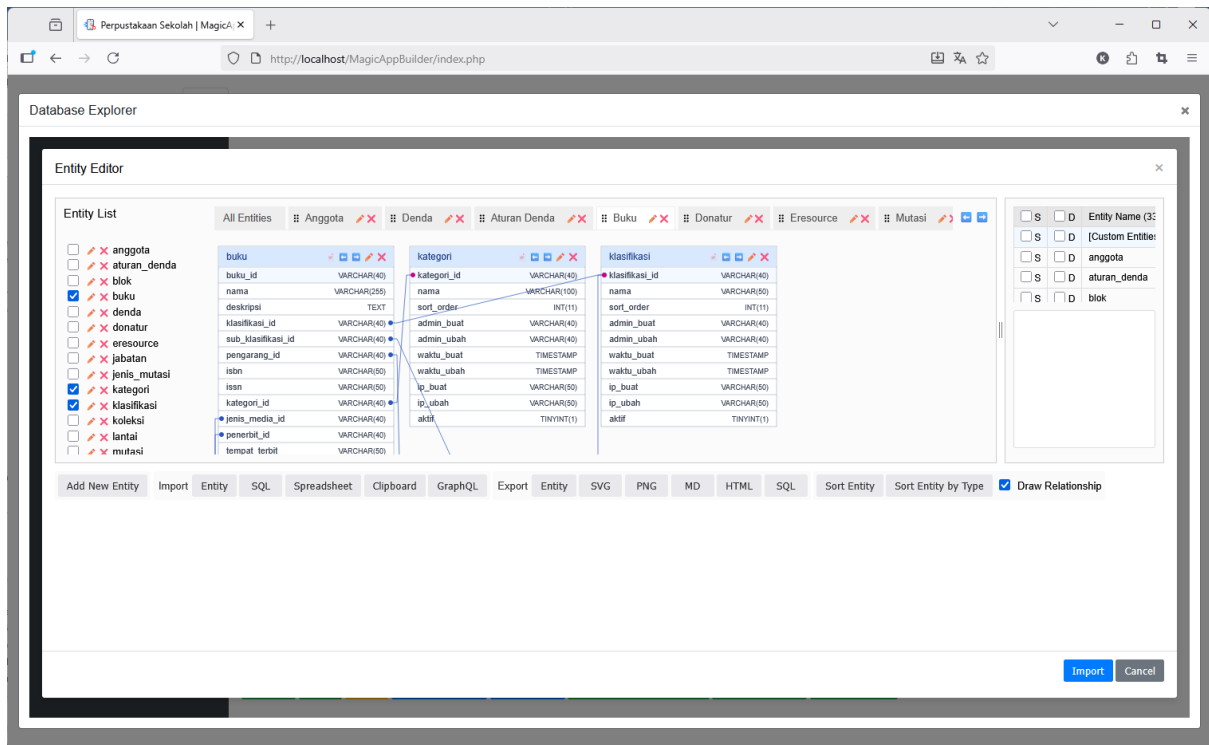
Jika tabel yang dipilih memiliki isi, Database Explorer akan menampilkan baris data secara detil sesuai dengan strktur tabel. Pengguna dapat mengubah dan menghapus baris data.

Untuk mengubah baris data, pilih ikon pensil di sebelah kiri baris. Database Explorer akan menampilkan formulir untuk mengubah data. Ubah data dari kolom yang disediakan. Kunci utama atau primary key dari baris data tidak dapat diubah. Jika pengguna ingin mengubahnya, pengguna dapat melakukannya dengan cara mengeksekusi query. Pilih tombol Update untuk menyimpan perubahan.

Untuk menghapus baris data, pilih ikon silang di sebelah kiri baris data. Database Explorer akan membuat query untuk menghapus baris data. Pilih tombol Execute untuk mengeksekusi query tersebut.

4.5.4 Langkah 7: Membuat Entitas dengan Entity Editor

Untuk membuat dan mengelola entitas menggunakan **Entity Editor** dalam **MagicAppBuilder**.



Bagian-Bagian Entity Editor

Entity Editor 5 bagian. Bagian-bagian tersebut adalah sebagai berikut:

a. Left bar

Left bar berisi Entity List, yaitu daftar entitas yang dibuat oleh pengguna, baik dibuat manual maupun diimpor dari file.

b. Main bar

Main bar berisi diagram entitas yang terdiri dari nama entitas, nama kolom, tipe kolom, panjang kolom (jika ada), ikon untuk menggeser ke kiri, ikon untuk menggeser ke kanan, ikon untuk memperbarui entitas dan ikon untuk menghapus entitas. Antara entitas dihubungkan dengan garis bantu. Garis ini bukanlah relasi entitas yang sebenarnya melainkan hanya menggambarkan kolom-kolom yang sesuai. Garis bantu dapat dihilangkan jika pengguna membuang centang Draw Relationship di bagian kanan bawah.

Di bagian atas terdapat tab. Di bagian kiri ada tab **All Entities** dan di sebelah kanan ada tombol “+”. Tombol “+” digunakan untuk membuat diagram. Saat pengguna memiliki tombol “+”, Entity Editor akan meminta pengguna untuk membuat nama diagram. Tekan tombol Enter di keyboard atau pilih tombol centang untuk melanjutkan.

Pengguna juga dapat menggeser tab entitas dengan cara memutar roda mouse ke atas atau ke bawah. Tab hanya akan bergeser jika sudah melebihi tempatnya atau overflow.

Ikon ekspor dokumen HTML digunakan untuk mengeksport diagram ke format HTML. Dokumen berisi diagram dan struktur tabel pada diagram tersebut. Dokumen ini dapat dicetak atau ditampilkan di browser.

Jika diperlukan pengeditan, pengguna dapat mengubah format diagram saat mengeksport dokumen. Format PNG dapat dipilih agar dokumen dapat diedit menggunakan aplikasi seperti Microsoft Word.

buku	
buku_id	VARCHAR(40)
nama	VARCHAR(255)
deskripsi	TEXT
klasifikasi_id	VARCHAR(40)
sub_klasifikasi_id	VARCHAR(40)
pengarang_id	VARCHAR(40)
isbn	VARCHAR(50)
issn	VARCHAR(50)
kategori_id	VARCHAR(40)
jenis_media_id	VARCHAR(40)
penerbit_id	VARCHAR(40)
tempat_terbit	VARCHAR(50)

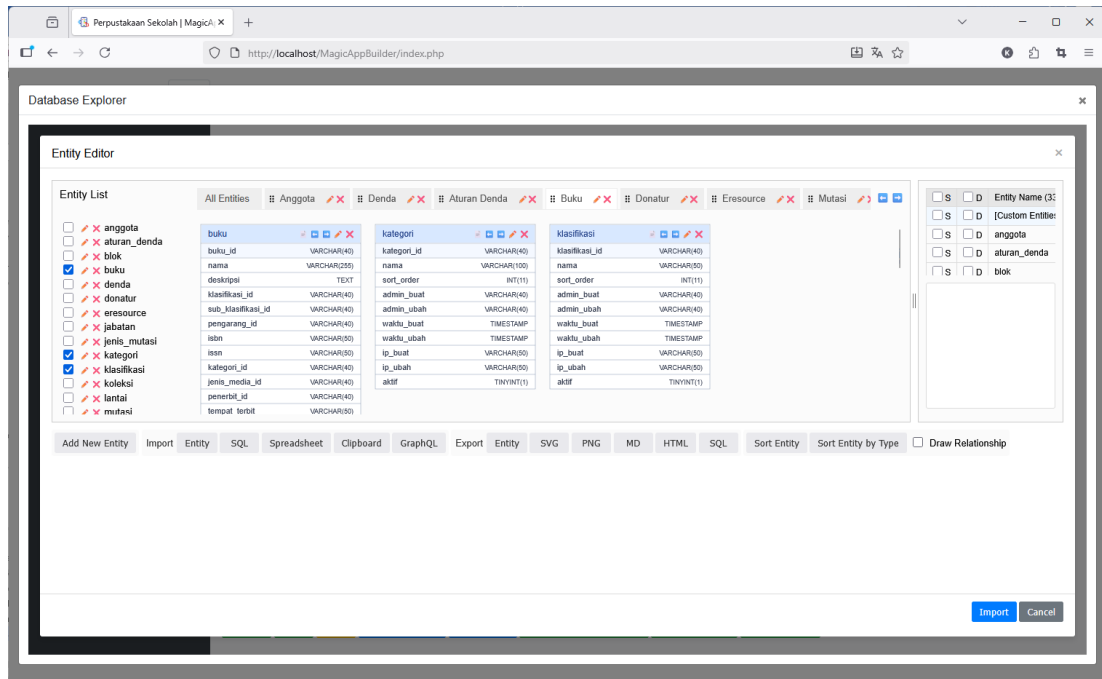
kategori	
kategori_id	VARCHAR(40)
nama	VARCHAR(100)
sort_order	INT(11)
admin_buat	VARCHAR(40)
admin_ubah	VARCHAR(40)
waktu_buat	TIMESTAMP
waktu_ubah	TIMESTAMP
ip_buat	VARCHAR(50)
ip_ubah	VARCHAR(50)
aktif	TINYINT(1)

klasifikasi	
klasifikasi_id	VARCHAR(40)
nama	VARCHAR(50)
sort_order	INT(11)
admin_buat	VARCHAR(40)
admin_ubah	VARCHAR(40)
waktu_buat	TIMESTAMP
waktu_ubah	TIMESTAMP
ip_buat	VARCHAR(50)
ip_ubah	VARCHAR(50)
aktif	TINYINT(1)

Pengguna dapat mengubah urutan tab dengan cara meyorot ikon :: di depan nama tab. Tahan klik, geser ke kanan atau ke kiri lalu lepas di posisi baru.

Pengguna hanya dapat mengubah urutan dari tab yang memiliki ikon ::.

Untuk memasukkan entitas ke dalam diagram, centang entitas yang ada di sebelah kiri. Entitas tersebut akan dimasukkan ke dalam diagram. Pengguna dapat mencentang entitas lain atau klik kanan di atas entitas yang sudah masuk ke dalam diagram lalu mencentang entitas lain. Entity Editor akan membuat garis bantu yang menghubungkan antar entitas dengan kolom yang berkaitan. Garis bantu dapat dihilangkan jika pengguna membuang centang Draw Relationship di bagian kanan bawah.



Pengguna dapat mengubah posisi entitas, mengubah struktur entitas dan menghapus entitas melalui ikon yang ada pada masing-masing entitas. Pengguna juga dapat mengubah struktur entitas dan menghapus entitas melalui ikon dari daftar entitas di left bar.

c. Right bar

Rightbar berisi daftar entitas yang sama dengan di bagian left bar dan text area. Perbedaannya adalah bahwa right bar dapat iatur lebarnya. Di sebelah kiri nama entitas terdapat centang. Jika entitas dicentang, Entity Editor akan membuat sebuah query **CREATE TABLE** sesuai dengan entitas apa saja yang dicentang. Query yang dibuat menggunakan dialek MySQL. Query ini akan secara otomatis dikonversi ke database yang digunakan jika pengguna memilih tombol Import di bagian kaki Entity Editor.

d. Button area

Di bagian bawah left bar dan main bar terdapat beberapa tombol yaitu sebagai berikut:

1. Add New Entity: digunakan untuk membuat entitas baru
2. Import Entity: digunakan untuk mengimpor entitas dari file hasil ekspor entitas dengan format JSON yang pernah dibuat baik di aplikasi yang sama maupun berbeda.

3. Import SQL: digunakan untuk mengimpor entitas dari file SQL dengan dialek MySQL, PostgreSQL, SQLite, SQL Server dan file database SQLite dalam bentuk file biner. *
4. Import Spreadsheet: digunakan untuk membuat entitas berdasarkan data dari file CSV, DBF, ODS, XLS dan XSLX. Untuk file Excel, format yang diterima hanyalah .xlsx, .xls dan .ods. MagicAppBuilder akan meminta pengguna untuk memilih sheet mana yang akan diimpor. *
5. Import Clipboard: digunakan untuk mengimpor entitas dari clipboard. MagicAppBuilder mendukung sumber data dari struktur data MagicAppBuilder, HTML dan markdown.
6. Import GraphQL: digunakan untuk mengimpor entitas dari skema GraphQL.
7. Export Entity: digunakan untuk mengekspor entitas ke format JSON.
8. Export SVG: mengekspor diagram yang sedang dibuka ke format SVG. Format ini dapat dirender menggunakan semua browser modern dan beberapa aplikasi pengolahan grafis. Gambar dapat diperbesar dan diperkecil ke ukuran berapapun tanpa menurunkan kualitasnya.
9. Export SVG: mengekspor diagram yang sedang dibuka ke format SVG. Format ini dapat dirender menggunakan semua browser modern dan beberapa aplikasi pengolahan grafis. Kualitas gambar akan turun baik diperbesar maupun diperkecil.
10. Export SQL: digunakan untuk mengekspor entitas ke format SQL. SQL yang dihasilkan sama dengan isidari text area di right bar.
11. Export HTML: digunakan untuk mengekspor diagram hubungan entitas (ERD) dengan format HTML. Pengguna dapat memilih apakah gambar akan diekspor dengan format PNG atau SVG. Format PNG memungkinkan pengguna membuka dan mengubah dokumen menggunakan aplikasi seperti Microsoft Word. Format SVG menjaga kualitas gambar tetap tinggi meskipun dimensinya diubah ke ukuran berapapun.
Di bagian bawah, terdapat tabel **Entities Ordered by Dependency Depth** yang memuat urutan ketergantungan entitas. Tabel ini adalah rujukan saat membuat modul. Pengguna harus membuat modul dengan entitas yang memiliki ketergantungan terkecil terlebih dahulu.
12. Sort Entity: digunakan untuk mengurutkan entitas yang ada di left bar. Urutan entitas pada tab All Entities juga akan berubah mengikuti urutan entitas pada left bar.
13. Sort Entity by Type: digunakan untuk mengelompokkan entitas berdasarkan jenis dan mengurutkan entitas yang ada di left bar. Entitas yang dibuat oleh pengguna (custom) akan diletakkan di atas entitas bawaan (system). Pengelompokan ini dimaksudkan untuk mempermudah pengelolaan entitas karena perubahan lebih banyak terjadi pada entitas custom. Urutan entitas

pada tab All Entities juga akan berubah mengikuti urutan entitas pada left bar.

14. Draw Relationship: centang untuk menggambar atau tidak garis bantu antar entitas.

*) Catatan

MagicAppBuilder juga akan mengimpor data dari tabel dan akan disimpan di entitas terkait. Jika sebuah kolom dihapus, maka data pada kolom tersebut akan terhapus saat entitas disimpan. Jika sebuah kolom baru ditambahkan, maka nilai kolom tersebut adalah **null** hingga datanya diubah dari editor. Jika sebuah kolom yang sebelumnya dihapus lalu ditambahkan lagi tanpa diselingi penyimpanan entitas, maka data pada kolom tersebut tidak akan hilang. Jika sebuah kolom yang sebelumnya dihapus lalu ditambahkan lagi setelah diselingi penyimpanan entitas, maka data pada kolom tersebut akan hilang.

Sebelum melakukan impor, penting untuk memperhatikan banyaknya data dari file sumber. Mengimpor data yang terlalu besar mungkin akan menyebabkan masalah karena keterbatasan memory.

e. Editor area

Editor area adalah bagian paling bawah dari Entity Editor tepat di atas kaki. Editor ini akan muncul dalam beberapa kondisi.

Membuat Entitas Baru

Saat pengguna memilih tombol Add New Entity, Entity Editor akan mengubah isi dari button area dengan input teks untuk membuat nama entitas dan beberapa tombol.

1. Tombol Add Colum: digunakan untuk membuat kolom baru. Kolom baru akan berada di bagian paling bawah tabel. Di sebelah kiri terdapat ikon untuk mengatur posisi kolom, menghapus kolom, dan memindahkan kolom selangkah demi selangkah.
Sejajar dengan Colum Name, pengguna dapat menentukan nama kolom.
Sejajar dengan Type, pengguna dapat menentukan tipe kolom
Sejajar dengan Length, pengguna dapat menentukan ukuran kolom untuk beberapa tipe kolom
Sejajar dengan Value, pengguna dapat menentukan nilai kolom untuk beberapa tipe kolom
Sejajar dengan Default, pengguna dapat menentukan nilai default kolom untuk beberapa tipe kolom
Sejajar dengan NL, pengguna dapat menentukan apakah kolom tersebut dapat berisi NULL atau tidak

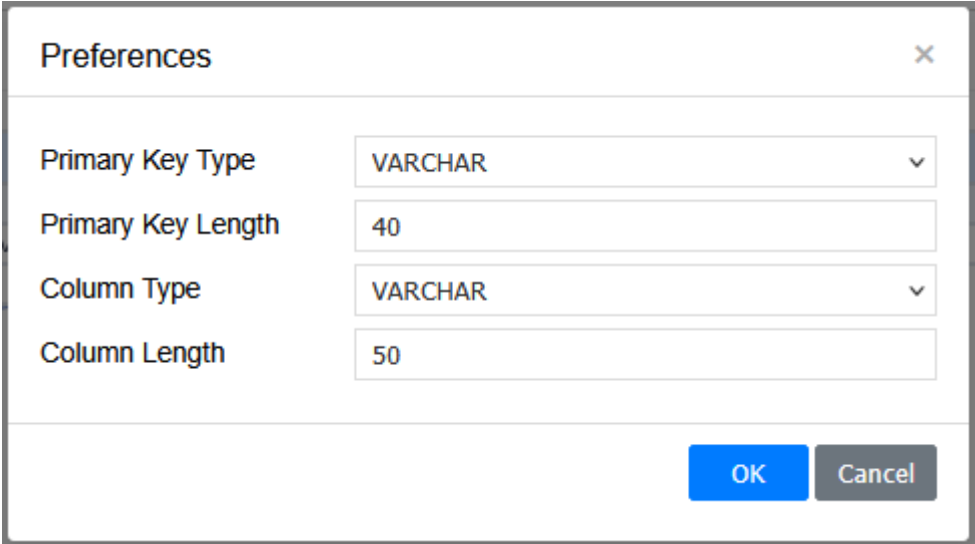
Sejajar dengan PK, pengguna dapat menentukan apakah kolom tersebut merupakan primary key atau tidak

Sejajar dengan AI, pengguna dapat menentukan apakah kolom tersebut auto increment atau tidak

Pengguna hanya dapat mencentang NL jika kolom bukan primary key

Pengguna hanya dapat mencentang AI jika kolom merupakan primary key dan bertipe integer

2. Add Column from Template: digunakan untuk memasukkan kolom-kolom dari template secara otomatis.
3. Save Entity: digunakan untuk menyimpan entitas
4. Edit Template: digunakan untuk mengubah template. Untuk mengubah template akan dijelaskan di bagian selanjutnya.
5. Preferences: digunakan untuk mengatur tipe data kolom primary key, panjang kolom primary key, tipe data kolom non primary key, dan panjang kolom non primary key. Pengaturan ini akan mempengaruhi saat pengguna membuat kolom baru. Meskipun demikian, pengguna tetap dapat mengubahnya untuk entitas tertentu jika tidak sesuai dengan pengaturan yang telah dibuat.



Preferences	
Primary Key Type	VARCHAR
Primary Key Length	40
Column Type	VARCHAR
Column Length	50
<div>OK Cancel</div>	

6. Cancel: digunakan untuk membatalkan perubahan. Saat pengguna membuat entitas baru, entitas tersebut tidak akan disimpan jika pengguna memilih tombol ini.

Memperbarui Entitas

Saat pengguna mengubah sebuah entitas, formulir yang akan tampil akan lebih kurang sama dengan saat pengguna membuat entitas baru. Perbedaannya adalah saat pengguna mengubah entitas, tabel editor sudah berisi kolom sesuai dengan kolom pada entitas yang diubah.

Di sebelah kanan tombol Cancel, ada 2 tombol yang hanya muncul pada saat memperbarui entitas.

- e. Description: digunakan untuk membuat deskripsi entitas.
- f. Data: digunakan untuk melihat, menambah, mengubah atau menghapus data entitas.

Mengimpor Entitas

Entity Editor mempunyai banyak pilihan untuk mengimpor entitas, di antaranya adalah sebagai berikut:

1. Impor dari file JSON

Impor dari file JSON dilakukan dengan mengimpor file JSON yang merupakan ekspor dari Entity Editor. Saat mengimpor file dari file JSON, Entity Editor akan menghapus semua entitas yang ada di editor dan akan membuat entitas dari awal.

Untuk mengimpor entitas dari file JSON, pilih tombol Import Entity lalu pilih file sumber. Entity Editor akan menghapus semua entitas yang ada sekarang dan menggantinya dengan entitas dari file sumber.

PERHATIKAN!

Karena Entity Editor akan menyimpan semua data ke server setiap kali ada perubahan, ekspor terlebih dahulu entitas yang ada sebelum mengimpor entitas baru sehingga Anda memiliki cadangan.

2. Impor dari file SQL

Impor dari file SQL dilakukan dengan mengimpor file SQL yang merupakan cadangan dari sebuah tabel atau database. Saat mengimpor file dari file SQL, Entity Editor akan menghapus semua entitas yang ada di editor dan akan membuat entitas dari awal.

Untuk mengimpor entitas dari file SQL, pilih tombol Import SQL lalu pilih file sumber.

3. Impor dari Spreadsheet

Impor dari spreadsheet dilakukan dengan mengimpor file Excel (xlsx dan xls), ODS, DBF dan CSV. Entity Editor tidak akan menghapus entitas yang ada tapi akan menambahkan entitas dari file spreadsheet. Untuk menambahkan entitas dari file SQL, pilih tombol Import Spreadsheet lalu pilih file sumber. Jika file sumber adalah CSV, maka nama default dari entitas adalah nama file.

Jika file sumber adalah Excel, maka nama default entitas adalah nama sheet. Jika file sumber adalah Excel dan memiliki lebih dari satu sheet, maka pengguna diminta untuk memilih sheet yang ada.

Data pada spreadsheet harus bersih. Baris pertama adalah nama kolom dan harus unik (tidak boleh ada kolom yang sama). Baris berikutnya adalah data dan tidak boleh ada baris kosong. Data harus memiliki sebuah kunci utama atau *primary key*. Jika data tidak memiliki kunci utama, buat sebuah kolom baru di sebelah kiri data kemudian beri nomor pada data yaitu 1, 2, 3, dan seterusnya hingga data terakhir.

4. Impor dari Clipboard

Impor dari clipboard dilakukan dengan menyalin (Ctrl+C) tabel dari aplikasi Excel, Word dan web (HTML) lalu paste (Ctrl+V) pada daerah entitas. Jika data yang diimpor tidak sesuai dengan yang diinginkan, salin dulu datanya ke aplikasi Excel lalu salin kembali data dari aplikasi Excel.

Data pada tabel harus bersih. Baris pertama adalah nama kolom dan harus unik (tidak boleh ada kolom yang sama). Baris berikutnya adalah data dan tidak boleh ada baris kosong. Data harus memiliki sebuah kunci utama atau *primary key*. Jika data tidak memiliki kunci utama, buat sebuah kolom baru di sebelah kiri data kemudian beri nomor pada data yaitu 1, 2, 3, dan seterusnya hingga data terakhir.

Setelah data sesuai persyaratan, salin data dengan cara memilih data lalu salin ke clipboard dengan tombol Ctrl+C di keyboard atau pilih ikon Copy dari aplikasi, kemudian tempelkan ke daerah entitas di Entity Editor. Entity Editor akan menampilkan formulir pembuatan entitas baru dengan nama "new_table", "new_table_1", "new_table_2" dan seterusnya. Ubah nama entitas dan beri primary key kemudian simpan entitas dengan tombol Save Entity.

Selain dengan cara Ctrl+V di daerah entitas, pengguna juga dapat memilih tombol Import Clipboard. Pengguna akan diminta memilih tombol Paste atau Tempel yang muncul di atas tombol tersebut. Ini merupakan fitur keamanan dari browser.

5. Impor dari Skema GraphQL

Impor dari skema GraphQL dapat dilakukan jika pengguna tidak memiliki akses terhadap database. Pengguna dapat merekonstruksi skema GraphQL menjadi struktur database. Perlu dicatat bahwa pengguna harus mendefinisikan ulang tipe data pada masing-masing entitas yang diimpor dari GraphQL. Penggunaan skema GraphQL adalah alternatif terakhir jika pengguna tidak memiliki sumber lain selain skema GraphQL.

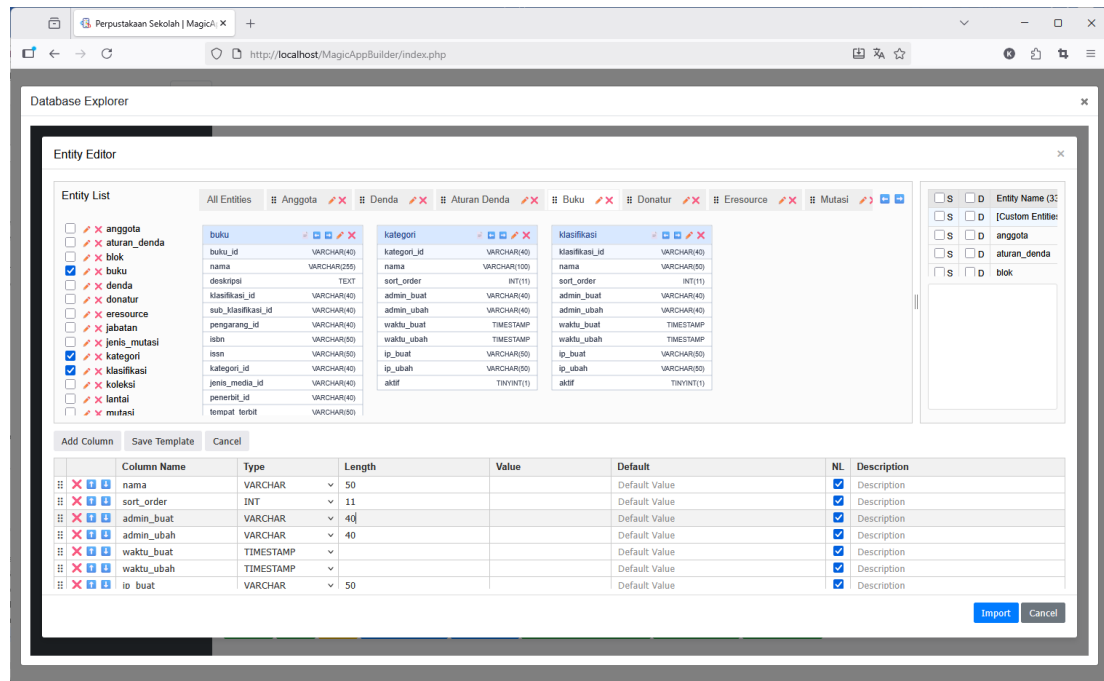
Mengubah Template

Tampilan editor untuk mengubah template sekilas mirip dengan editor entitas. Saat pengguna memilih tombol Edit Template, button area akan diisi oleh tombol Add Column, Save Template dan Cancel.

Tabel editor secara default berisi beberapa kolom yang ditetapkan sebagai Reserved Colum di pengaturan aplikasi. Pengguna dapat menambahkan kolom baru pada template yang bukan merupakan reserved column.

Adapun penjelasan tombol-tombol pada button area adalah sebagai berikut:

1. Add Column: digunakan untuk menambahkan kolom baru
2. Save Template: digunakan untuk menyimpan template
3. Cancel: digunakan untuk membatalkan perubahan



f. Data Entitas

Pengguna dapat menampilkan, menambah, mengubah dan menghapus data entitas. Saat pengguna mengimpor entitas baik dari spreadsheet, file SQLite, atau query yang mengandung INSERT INTO, Entity Editor juga akan mengimpor data tersebut ke dalam entitas. Data ini dapat ditambah, diubah, atau dihapus.

Pengguna dapat menambahkan data dari file XLS, XLSX, ODS, DBF, maupun CSV. Pengguna juga dapat mengubah data dengan cara mengetikkannya secara manual pada baris dan kolom tertentu.

Di bawah tabel terdapat alat untuk memperbaiki format tanggal dan jam. Data yang diimpor dari file XLS, XLSX, ODS, DBF, maupun CSV berpotensi memiliki format yang tidak dapat diterima oleh database. Data ini harus diperbaiki sebelum menyimpan data tersebut ke database.

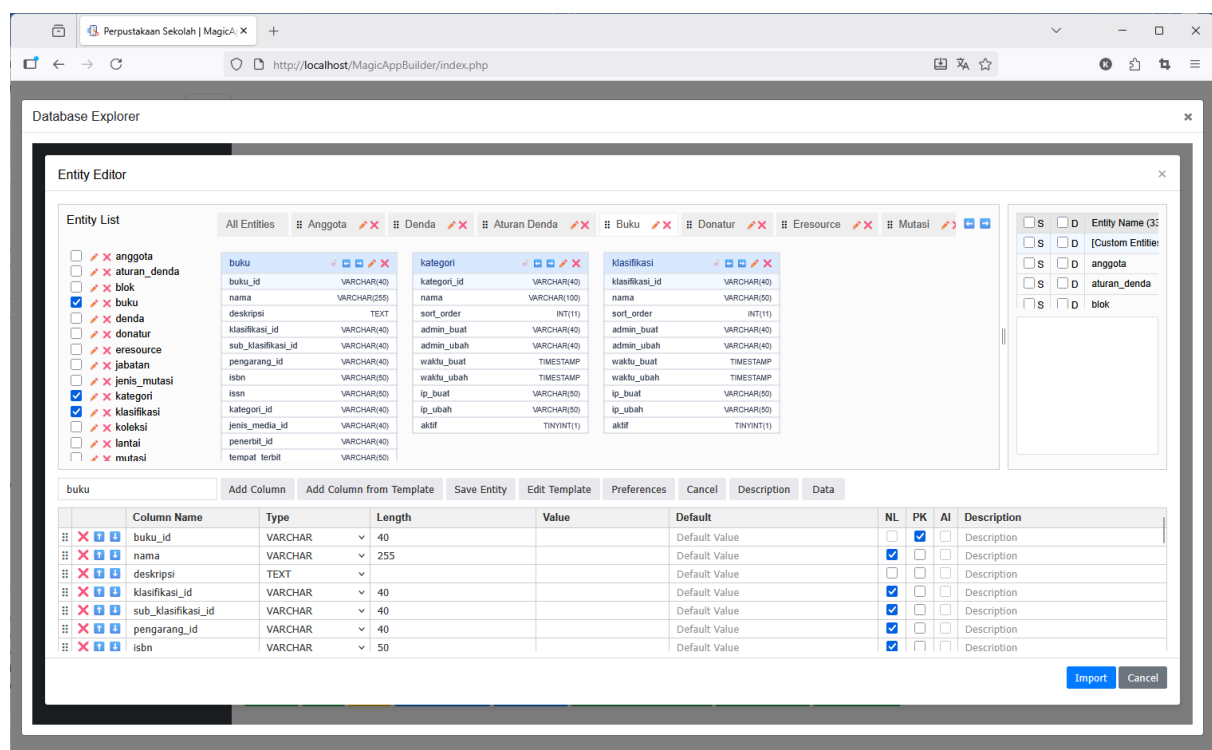
Untuk memperbaiki format tanggal dan jam, pilih kolom tanggal dan jam yang akan diperbaiki, format tanggal saat ini, dan mode.

Fix Date Time Select Column Format mdy Mode datetime Fix Data

Pilihan kolom tergantung dengan kolom entitas dan diperbarui setiap pengguna menampilkan data entitas. Format adalah format tanggal yang akan diperbaiki. Pengguna harus memilih format tanggal karena pada praktiknya ada perbedaan format tanggal yang digunakan. mdy (month, date, year) untuk urutan bulan, tanggal dan tahun. dmy (date, month, year) untuk urutan tanggal, bulan dan tahun. ymd (year, month, date) untuk urutan tahun, bulan dan tahun. Entity Editor menyediakan tiga mode yaitu datetime untuk tanggal dan jam, date untuk tanggal saja, dan time untuk jam saja. Pilih salah satu dari mode yang tersedia.

Pilih tombol Fix Data untuk memperbaikinya. Entity Editor akan memperbaiki semua baris dari kolom tersebut. Data yang telah diperbaiki akan memiliki format YYYY-MM-DD HH:mm:ss. Setelah data diperbaiki, pilih tombol Save. Lakukan untuk setiap entitas dan setiap kolom yang akan diperbaiki.

Di bagian kaki dari Entity Editor terdapat tombol Import dan Cancel.



Tombol Import digunakan untuk mengimpor SQL dari Entity Editor ke Database Explorer dengan mengkonversi dialeknya. Jika entitas memiliki data, maka query **INSERT** akan disertakan setelah semua query **CREATE TABLE** selesai dibuat. Query **INSERT** juga akan menyesuaikan dengan dialek database yang digunakan. Nilai **null** hanya akan dibuat jika kolom mendukung data **null**. Untuk kolom dengan **AUTO INCREMENT**, nilai awal wajib diisi karena MagicAppBuilder akan menset dengan nilai **0** jika kosong.

Tombol Cancel digunakan untuk menutup Entity Editor tanpa melakukan tindakan lain.

4.5.5 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.

4.5.6 Langkah 9: Membuat Modul

1. Pilih Tabel
 - Di tab **Select Table**, klik tombol **Reload Table** untuk memuat tabel yang telah pengguna buat sebelumnya. Lakukan ini setiap kali pengguna membuat tabel baru di database.
 - Tabel dikelompokkan menjadi dua yaitu System Tables dan Custom Tables. System Tables berisi tabel-tabel bawaan MagicAppBuilder sedangkan Custom Tables berisi tabel-tabel yang dibuat oleh pengguna. Custom Tables berada di atas System Tables.
 - Pilih salah satu tabel yang akan digunakan untuk membuat modul. **Anda harus memulai dengan tabel dengan Dependency Depth yang paling kecil terlebih dahulu.**
2. 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.
3. 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.

4. 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.

4.5.7 Langkah 10: Penjelasan Tab Generate Module

Tab **Generate Module** adalah tab utama dalam pembuatan setiap modul dan entitas. 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).

2. 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.

3. 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.
- **EX:** Menandakan bahwa pencarian menggunakan **pencocokan nilai secara persis** (exact match). Pengguna harus memasukkan **seluruh nilai dengan tepat** sebagai kriteria pencarian. Misalnya, saat mencari judul lagu “Bahagia Bersamamu”, pengguna **tidak bisa** hanya mengetikkan “Bahagia” saja. Selain itu, **huruf besar dan kecil juga dibedakan**, sehingga “bahagia bersamamu” tidak sama dengan “Bahagia Bersamamu”.

Mengapa ini penting?

Pencarian exact berguna ketika hasil yang diinginkan harus **benar-benar spesifik dan tidak ambigu**. Ini sangat membantu dalam:

- **Pencarian berdasarkan kode unik**, seperti kode transaksi atau nomor referensi.
- **Validasi data yang sensitif terhadap huruf besar/kecil**, seperti username atau tag identifikasi.
- **Meningkatkan performa kueri**, karena exact match dapat memanfaatkan indeks database secara lebih efisien dibanding pencarian berbasis LIKE atau wildcard.

Dengan mode ini, pengguna memiliki kendali lebih besar terhadap **akurasi hasil pencarian** dan **konsistensi data** yang ditampilkan.

4. Tipe Data

Isian ini akan menentukan tipe data dari bidang teks input.

5. Format Data

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

6. Tipe Filter

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

7. 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 diisi. 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.

Specification

Just leave it blank if it doesn't exist. Click Remove button to remove value.







Column Name	Comp	Value	Rem	Move
active	=	true		 
draft	=	false		 
album.active	=	true		 
+ Add Row				

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.

Sortable

Use at least one column to sort.

Column	Value	Rem	Move
sortOrder	ASC		 
name	ASC		 
+ Add Row			

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. Pengelompokan dengan Entity

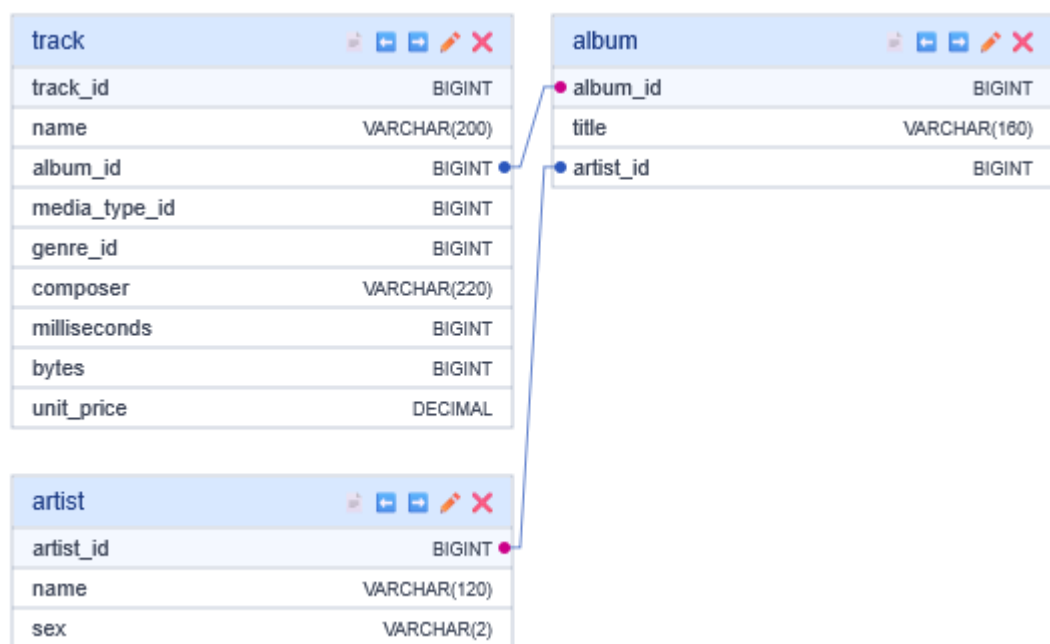
Pengelompokan dengan entity atau entitas dilakukan dengan cara menggunakan entitas sebagai sumber.

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.



Grouping	
Value	<input type="text" value="artist_id"/>
Label	<input type="text" value="name"/>
Source	<input checked="" type="radio"/> Entity <input type="radio"/> Map
Reference	<input type="text" value="artist"/>

Dalam contoh di atas, saat pengguna membuat modul “Track” dari tabel “track” dan membuat relasi pada kolom “album_id”, pada bagian Grouping, “Value” diisi kolom dari tabel “album” yang akan digunakan untuk mengelompokkan data “Album”. Kolom “artist_id” dapat digunakan karena

satu artis mungkin memiliki lebih dari satu album. “Label” adalah kolom yang akan diambil nilainya. Karena pilihan pada “Source” adalah “Entity”, maka “Reference” adalah sebuah entitas. Nama objek dari entitas tersebut adalah “artist”. Dari mana pengguna mengetahui nama entitas tersebut adalah “artist”? “artist” adalah nama objek referensi saat membuat modul “Album”. “Label”, yang diisi dengan “name”, merujuk pada kolom “name” pada entitas Artist.

Create Data Reference

☒ Entity ☐ Map ☐ Yes/No ☐ True/False ☐ 1/0

Entity

Entity Name

Table Name

Primary Key

Value Column

Reference Object Name

Reference Property Name

2. 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.

Pengelompokan dengan pemetaan ini dilakukan jika sebuah kolom akan dijadikan acuan namun kolom tersebut hanya berisi sebuah kode. Misalnya saat pengguna membuat modul Album, pada kolom “artist_id”, pengguna akan mengelompokkannya berdasarkan kolom “sex”. “sex” tidak merujuk ke entitas lain. Nilainya hanya “M” dan “F” atau kosong.

Value	Label
M	Male
F	Female

Artist dengan “sex” adalah “M” akan masuk pada kelompok “Male”, sedangkan artist dengan “sex” adalah “M” akan masuk pada kelompok “Female”. Artist dengan “sex” di luar itu akan berada di bawahnya dan tidak masuk ke dalam kelompok manapun.

Bagian Additional Output

Bagian Additional Output digunakan untuk membuat atribut 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.

JavaScript memiliki API khusus untuk mengambil atribut dengan awalan “data-” yaitu dengan properti **dataset**.

2. **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.

3. **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.

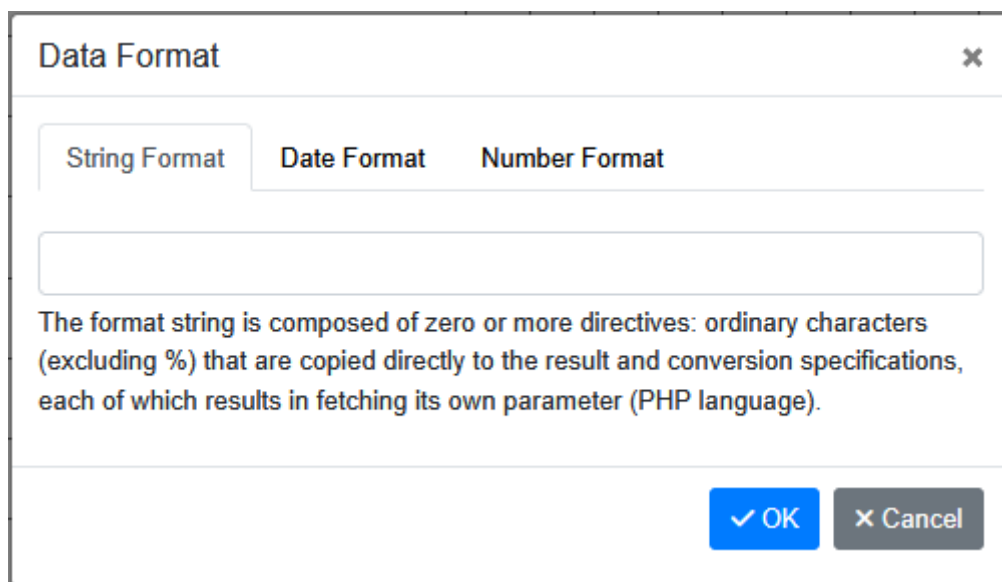
Nilai	Kode HTML	Deskripsi Singkat	Output
text	<input type='text'>	Teks	Teks
email	<input type='email'>	Alamat email	Teks
url	<input type='url'>	URL	Teks
tel	<input type='tel'>	Nomor telepon	Teks
password	<input type='password'>	Kata sandi	Tidak ada
int	<input type='number'>	Bilangan bulat	Teks*
float	<input type='number' step='any'>	Bilangan desimal	Teks*

Nilai	Kode HTML	Deskripsi Singkat	Output
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:



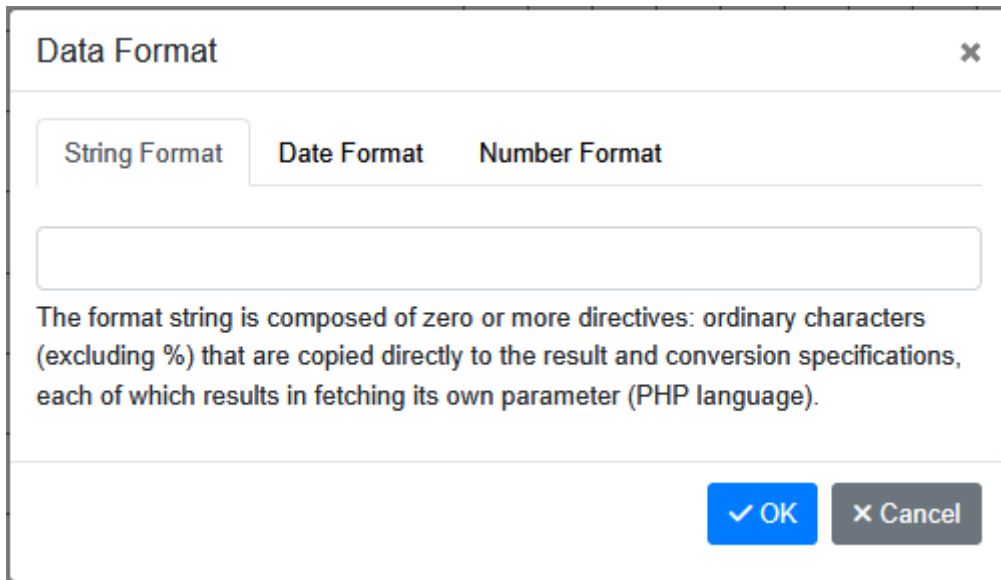
Dialog terdiri dari 3 tab yaitu sebagai berikut:

4. Tab String Format
5. Tab Date Format
6. 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

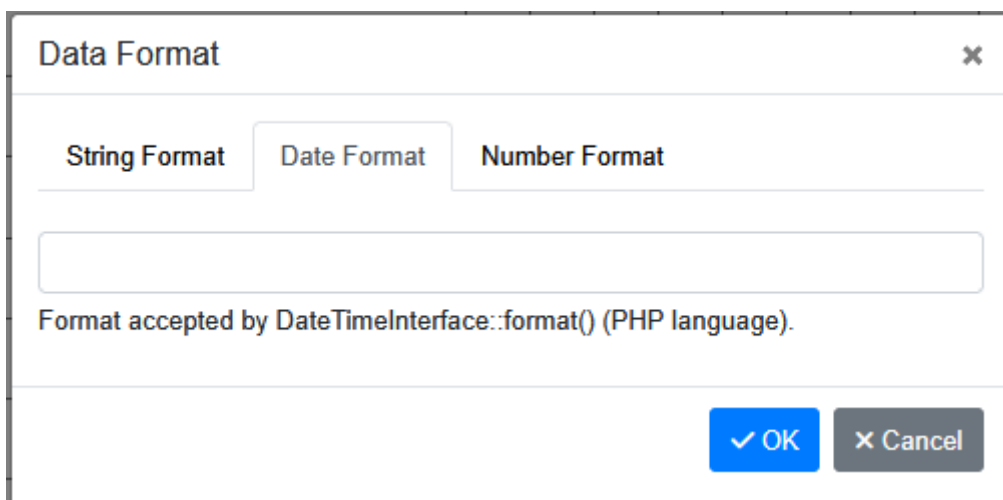
demikian, pada aplikasi yang sama, dua orang pengguna dapat menyimpan format yang berbeda.

Tab String Format digunakan untuk memformat 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**.



The screenshot shows a dialog box titled "Data Format" with a close button (X) in the top right corner. It has three tabs: "String Format" (selected), "Date Format", and "Number Format". Below the tabs is a text input field. Under the input field, there is a descriptive text: "The format string is composed of zero or more directives: ordinary characters (excluding %) that are copied directly to the result and conversion specifications, each of which results in fetching its own parameter (PHP language)." At the bottom right, there are two buttons: "OK" (with a checkmark icon) and "Cancel" (with an X icon).

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



The screenshot shows the same "Data Format" dialog box, but with the "Date Format" tab selected. The text input field is empty. The descriptive text below it reads: "Format accepted by DateTimeInterface::format() (PHP language)." The "OK" and "Cancel" buttons are at the bottom right.

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

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

Nilai	Deskripsi Singkat
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 perilaku 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 eksplisit 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 Diijinkan

- **@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) tidak 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**. Pilih 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:
 - **Activate/Deactivate:** adalah fitur untuk mengaktifkan dan menonaktifkan data. Data nonaktif tidak akan ditampilkan sebagai pilihan dan pada modul-modul tertentu mungkin tidak diakui sebagai data yang sah. Akan tetapi, data lama yang merujuk kepada data tersebut dari entitas lain akan tetap menggunakan data tersebut. Fitur ini sangat penting untuk menandai sebuah data agar tidak digunakan lagi tanpa harus menghapusnya.
 - **Manual Sort Order:** adalah fitur untuk mengubah urutan data secara manual dengan cara seret dan jatuhkan (drag and drop). Fitur ini sangat penting untuk mengurutkan data seperti pada data-data referensi yang memerlukan urutan tertentu, baik untuk kebutuhan teknis aplikasi maupun kebutuhan estetika semata.
 - **Export to Excel:** adalah fitur untuk mengekspor data ke format Excel. Data yang diekspor sesuai dengan data pada daftar tanpa pemisahan halaman. Detil data yang diekspor sesuai dengan pilihan pada kolom **E**.

Pengguna hanya bisa memilih salah satu dari “Export to Excel” dan “Export to CSV”.

- **Export to CSV:** adalah fitur untuk mengekspor data ke format *Comma Separated Value* (CSV). Data yang diekspor sesuai dengan data pada daftar tanpa pemisahan halaman. Detil data yang diekspor sesuai dengan pilihan pada kolom **E**. Pengguna hanya bisa memilih salah satu dari “Export to Excel” dan “Export to CSV”.
- **Use Temporary File:** adalah pilihan saat mengekspor data dengan format CSV. Jika User Temporary File dipilih, aplikasi akan membuat file sementara di server lalu mentransfernya ke klien saat file tersebut selesai. Pengguna dapat mengoptimalkan penggunaan sumber daya dengan memilih apakah akan menggunakan file sementara atau tidak, tergantung pada ukuran file yang dibuat. Pengguna perlu menganalisa perbedaan tersebut untuk kinerja sistem yang paling efisien.
- **User Activity Log:** adalah pilihan untuk mencatat aktivitas pengguna. Pemilik aplikasi mungkin tidak memerlukan semua catatan dari aktivitas pengguna dengan alasan efisiensi sumber daya. Pengguna dapat memilih modul apa saja yang akan mencatat aktivitas pengguna.
- **Approval:** adalah fitur untuk alur kerja. Jika pilihan Approval dicentang, aplikasi akan menerapkan alur sebagai berikut:
 - a. Saat membuat data baru, data memiliki status sebagai draf (kolom draft bernilai true atau 1 dan kolom waiting_for bernilai 1) dan modul lain akan menganggap data tersebut belum sah. Jika pembuatan data tersebut disetujui, status draf akan dihapus (kolom draft akan bernilai false atau 0 dan kolom waiting_for akan bernilai 0). Jika pembuatan data tersebut ditolak, data akan dihapus.
 - b. Saat mengubah data yang sudah ada, informasi pada tabel primer tidak akan diubah melainkan statusnya saja yang berubah yaitu memerlukan persetujuan untuk diubah (kolom waiting_for akan bernilai 2). Semua informasi perubahan disimpan pada tabel persetujuan dengan nama yang sama dan diakhiri dengan suffix “_apv”. Jika perubahan data tersebut disetujui, maka semua perubahan informasi yang ada di tabel persetujuan akan disalin ke tabel primer dan status dari data akan dinormalkan kembali (kolom waiting_for akan bernilai 0). Jika perubahan data tersebut ditolak, status dari data akan dinormalkan kembali (kolom

waiting_for akan bernilai 0) tanpa mengubah informasi pada tabel primer.

- c. Saat mengaktifkan data yang sudah ada, status data akan berubah, yaitu memerlukan untuk diaktifkan (kolom waiting_for akan bernilai 3). Jika tindakan tersebut disetujui, status dari data akan dinormalkan kembali (kolom waiting_for akan bernilai 0) dan kolom active akan bernilai true atau 1. Jika tindakan tersebut ditolak, status dari data akan dinormalkan kembali (kolom waiting_for akan bernilai 0) tanpa mengubah informasi pada tabel primer.
 - d. Saat menonaktifkan data yang sudah ada, status data akan berubah, yaitu memerlukan untuk dinonaktifkan (kolom waiting_for akan bernilai 4). Jika tindakan tersebut disetujui, status dari data akan dinormalkan kembali (kolom waiting_for akan bernilai 0) dan kolom active akan bernilai false atau 0. Jika tindakan tersebut ditolak, status dari data akan dinormalkan kembali (kolom waiting_for akan bernilai 0) tanpa mengubah informasi pada tabel primer.
 - e. Saat menghapus data yang sudah ada, status data akan berubah, yaitu memerlukan untuk penghapusan (kolom waiting_for akan bernilai 5). Jika tindakan tersebut disetujui, data akan dihapus. Akan tetapi, jika fitur Trash digunakan, aplikasi akan menyalin data yang dihapus ke tabel daur ulang dengan nama yang sama dengan tabel primer dan diakhiri dengan suffix “trash” baru kemudian menghapus data tersebut dari tabel primer. Jika tindakan tersebut ditolak, status dari data akan dinormalkan kembali (kolom waiting_for akan bernilai 0) tanpa adanya penghapusan data.
- **Approval Type:** adalah gaya untuk melakukan persetujuan. Jika pilihan Separated yang dipilih, aplikasi akan memisahkan tombol **Setujui** dan **Tolak** pada daftar. Pengguna akan diarahkan ke sebuah formulir baru untuk melanjutkan. Jika pilihan Combined yang dipilih, aplikasi hanya akan menampilkan satu tombol untuk menyetujui atau menolak tindakan. Pengguna dapat melanjutkan persetujuan di formulir berikutnya.
 - **Approval Position:** adalah posisi tombol persetujuan. Jika pilihan Before Data yang dipilih, aplikasi akan menampilkan tombol persetujuan di sebelah kiri data. Jika pilihan After Data yang dipilih, aplikasi akan menampilkan tombol persetujuan di sebelah kanan data. Fitur ini memberikan fleksibilitas kepada pengguna terutama untuk data dengan sirkulasi yang tinggi yang membutuhkan persetujuan dengan cepat.

- **Approval by Another User:** adalah pilihan apakah sebuah tindakan harus disetujui oleh pengguna yang berbeda. Jika pilihan ini dipilih, pengguna yang melakukan tindakan tidak dapat menyetujui tindakan tersebut. Fitur ini sangat penting untuk memastikan bahwa data yang dimasukkan diperiksa minimal oleh dua orang.
- **Bulk Approval:** adalah pilihan untuk melakukan persetujuan terhadap banyak data sekaligus. Fitur ini sangat penting terutama untuk data dengan sirkulasi yang tinggi yang membutuhkan persetujuan dengan cepat.
- **Trash:** adalah fitur untuk mendaur ulang data yang dihapus. Fitur ini dapat dikombinasikan dengan Approval. Data yang dihapus dapat dikembalikan atau dihapus permanen dari modul Data Restoration.
- **Validation:** adalah fitur untuk membuat validasi data.
Jika fitur ini diaktifkan dan pengguna mendefinisikan Validation pada setiap kolom, maka aplikasi akan menggunakan definisi validasi dari setiap kolom yang diatur pengguna. Kolom yang tidak diatur tidak akan divalidasi.
Jika fitur ini diaktifkan sedangkan pengguna tidak mendefinisikan satu pun kolom, maka aplikasi hanya akan memvalidasi panjang dari data yang dimasukkan. Misalnya, nama didefinisikan sebagai VARCHAR(50), maka saat pengguna memasukkan nama dengan panjang lebih dari 50 karakter, aplikasi akan menolak data tersebut dan meminta pengguna memperbaikinya.
- **Backend Only:** adalah fitur untuk hanya membuat kode untuk pemrosesan data yang dikirim saja dan untuk ekspor data. Fitur ini dapat digunakan pada aplikasi dengan arsitektur microservices meskipun pengguna memilih arsitektur monolith atau membuat aplikasi.
- **AJAX Support:** adalah fitur yang memungkinkan browser menggunakan AJAX saat pengguna saat:
 - a. melakukan pencarian data dalam daftar
 - b. berpindah halaman
 - c. mengaktifkan data
 - d. menonaktifkan data
 - e. menghapus data
 - f. mengubah urutan data

Fitur ini sangat penting untuk meningkatkan kecepatan akses dan menghemat sumber daya karena tingginya interaksi pengguna saat membuka daftar pada setiap modul.

- **Subquery:** adalah fitur untuk menggunakan subquery alih-alih menggunakan JOIN pada saat mengakses database. Tabel besar dengan banyak kolom dan banyak baris serta memiliki hubungan ke banyak tabel akan membutuhkan sumberdaya yang besar jika dilakukan JOIN.
Kapan subquery digunakan?
Subquery dapat digunakan jika pengguna hanya ingin mengambil satu kolom dari tabel referensi dan tidak menginginkan data lebih dari 2 tingkat. Tanpa subquery, aplikasi akan terus menelusuri data sesuai dengan rujukan yang didefinisikan pada entitas. Untuk relasi data yang kompleks, berpotensi terjadinya perulangan tanpa henti jika sebuah tabel rujukan kembali merujuk tabel lain yang akan merujuk kepada tabel itu lagi.

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.

4.5.8 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).

4.5.9 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.

4.5.10 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.

4.5.11 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 izin. Aplikasi hanya dapat diakses dari localhost. Application Option memungkinkan pengguna untuk mengatur semua hal di atas.

Saat pengguna memilih tombol “Option” pada kartu aplikasi, Application Option menampilkan dialog yang isinya tergantung pada status aplikasi.

Jika aplikasinya sah, maka pada Application Option akan muncul sebagai dialog yang memiliki 6 akordion yaitu sebagai berikut:

1. Application Menu
2. Application User
3. Application Mode
4. Application Icon
5. Dependency Update
6. Data Restoration

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 sebagai “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.

2. 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 melalui 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.

3. 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 pembangunan aplikasi.

Application Icon

Pada akordion ini, pengguna dapat mengganti ikon aplikasi. Tombol Upload Icon digunakan untuk memilih gambar yang akan dijadikan ikon. Gambar harus memiliki ukuran persegi dengan lebar dan tinggi minimal 512 pixel. MagicAppBuilder akan membuat beberapa file dengan format ICO dan PNG dan file JSON yang akan dibaca oleh browser.

Ikon aplikasi harus diatur untuk mencegah eror 404 jika aplikasi diakses menggunakan browser dari smartphone. Browser iPhone dan Android akan mengakses ikon dengan dimensi tertentu. Jika ikon tidak ditemukan, tentu saja akan menimbulkan eror 404. Di server, eror ini akan menumpuk di **error_log** server dan menimbulkan masalah jika tempat penyimpanan habis.

Untuk mengatur ikon aplikasi, pilih tombol **Upload Icon** lalu unggah gambar persegi dalam format PNG atau SVG (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"  
}
```

Dependency Update

Pada akordion ini, pengguna dapat melakukan pembaruan dependency. MagicAppBuilder menyediakan 3 tombol yaitu sebagai berikut:

1. Update MagicObject
2. Update MagicApp
3. Update Classes

Update Classes digunakan untuk memperbarui kelas-kelas dari MagicAppTemplate namun hanya berlaku untuk file yang berada satu tingkat di dalam direktori MagicAppTemplate namun tidak berlaku untuk file-file yang berada di dalam subdirektornya. Pembaruan ini bertujuan untuk memastikan defisi terbaru dari kelas-kelas tersebut.

Data Restoration

Pada akordion ini, pengguna dapat menentukan tabel mana yang dapat dipulihkan. Saat pengguna aplikasi menghapus data, data tersebut dipindahkan dari tabel primer ke tabel **trash**. Data tersebut dapat dikembalikan ke tabel primer.

Data dapat dikembalikan dengan syarat bahwa modul di mana data dihapus dilengkapi dengan fitur trash. MagicAppBuilder akan membuat entitas sesuai dengan nama tabel. Akordion Data Restoration menyediakan dua tombol sebagai berikut:

1. Update

Tombol Update digunakan untuk membuat entitas primary dan entitas trash untuk kebutuhan pengembalian data. Entitas ini berbeda dengan entitas lain di mana pengguna tidak diijinkan untuk mengubahnya secara manual. Jika terjadi perubahan struktur tabel, maka pengguna harus memperbarui entitas dengan cara memilih tabel apa saja yang akan diperbarui lalu memilih tombol Update. MagicAppBuilder akan memperbarui entitas primer dan trash dari tabel yang dipilih. Data dari tabel yang memiliki pasangan entitas primer dan trash dapat dikembalikan.

Sangat penting untuk selalu memperbarui entitas secara berkala selama tahap pengembangan aplikasi. Kolom pada entitas harus sama persis dengan kolom yang ada di dalam tabel terkait agar data dapat dikembalikan secara utuh. Jika ada kolom di database yang tidak ada pada entitas, maka data pada kolom tersebut tidak dapat dikembalikan. Sebaliknya, jika ada kolom pada entitas yang tidak ada di dalam tabel, maka akan terjadi error saat proses pengembalian data dan data tidak dapat dikembalikan sama sekali.

2. Delete

Tombol Delete digunakan untuk menghapus entitas primer dan trash dari tabel yang dipilih dan juga menghapusnya dari konfigurasi aplikasi sehingga pengguna tidak mungkin mengembalikan data yang dihapus meskipun data tersebut tersimpan di tabel trash.

Menghapus entitas dengan tombol Delete di halaman ini tidak akan menghapus data di dalam table trash. Data yang dihapus dari modul tetap tersimpan di tabel trash karena entitas trash pada halaman ini berada di direktori tersendiri dan tidak akan mengganggu fitur soft delete pada modul.

Jika aplikasinya tidak sah, MagicAppBuilder akan menampilkan status aplikasi dan tombol Validate. Tombol Validate dapat digunakan untuk memperbarui keabsahan dari sebuah aplikasi. Jika pengguna telah mencoba memeriksa keabsahan sebuah aplikasi, namun aplikasinya tetap ydak sah, MagicAppBuilder akan menampilkan tombol baru yaitu Rebuild. Tombol Rebuild dapat digunakan untuk membangun ulang aplikasi. MagicAppBuilder akan menampilkan dialog konfirmasi kepada pengguna untuk mencegah pengguna membangun ulang aplikasi tanpa sengaja.

Rebuild Application

Membangun ulang aplikasi berarti membangun ulang aplikasi dari awal.

MagicAppBuilder akan menganggap direktori aplikasi kosong dan akan membuat file-file aplikasi yang diperlukan tanpa mempedulikan isi dari direktori aplikasi tersebut.

Dengan kata lain, jika ada file dengan nama dan ekstensi yang sama, maka isinya akan ditimpa tanpa adanya konfirmasi terlebih dahulu.

4.5.12 Langkah 15: 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 3 tema standard. Ada dua tema dengan menu ditampilkan dalam dua tingkat dan 1 tema dengan menu ditampilkan dalam tingkatan tidak terbatas. Pengguna dapat mengembangkan tema sendiri sesuai dengan kebutuhan dan preferensi.

4.5.13 Langkah 16: 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 konfigurasi 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
 - PostgreSQLPilih salah satu yang paling sesuai dengan kebutuhan dan keahlian tim Anda, lalu pastikan DBMS tersebut sudah terinstal dan berjalan dengan baik di server.
4. 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.
5. Buat struktur database sesuai dengan DBMS yang digunakan.
6. 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.
7. Setelah struktur database di lingkungan produksi selesai dibuat, salin semua kode sumber aplikasi ke lingkungan produksi.
8. Atur konfigurasi aplikasi melalui file /inc.cfg/application.yml.
9. Jika data session akan disimpan di Redis, persiapkan server Redis di lingkungan produksi dan pastikan keamanannya.

10. Pastikan konfigurasi session dan cookie telah memenuhi standar keamanan yang baik.
11. Uji coba aplikasi dengan cara memasukkan data baik data percobaan maupun data nyata.
12. 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.

4.4 Impor Aplikasi

Impor aplikasi diperlukan untuk memindahkan aplikasi dari satu server ke server lain. Impor aplikasi yang sempurna membutuhkan konfigurasi aplikasi dan source code aplikasi yang telah dibuat. MagicAppBuilder akan membawa informasi direktori aplikasi. Pengguna perlu memindahkan aplikasi dari server lama ke server baru secara manual. MagicAppBuilder tidak menyediakan alat untuk memindahkan aplikasi untuk alasan keamanan.

Dari dialog Import Application, pengguna dapat membuat ulang aplikasi dari direktori aplikasi. MagicAppBuilder akan mendaftarkan aplikasi ke database dan membuat ulang konfigurasi aplikasi. Akan tetapi, konfigurasi modul dan database tidak secara otomatis dibuat oleh MagicAppBuilder.

4.5 Ekspor Aplikasi

Ekspor aplikasi diperlukan untuk mendownload file-file konfigurasi aplikasi dari MagicAppBuilder. Ekspor aplikasi dapat dilakukan dengan dua cara yaitu dari kartu aplikasi dan dari kartu workspace.

Ekspor aplikasi dari kartu aplikasi akan mengekspor aplikasi satu persatu. File-file konfigurasi dikompres dan dipaketkan dalam sebuah file ZIP. File ZIP tersebut kemudian dapat diimpor ke server lain. MagicAppBuilder akan membawa informasi direktori aplikasi. Pengguna perlu memindahkan aplikasi dari server lama ke server baru secara manual. MagicAppBuilder tidak menyediakan alat untuk memindahkan aplikasi untuk alasan keamanan.

Ekspor aplikasi dari kartu workspace akan mengekspor semua aplikasi dalam sebuah workspace. Setiap aplikasi akan dipaketkan menjadi sebuah file ZIP dan file-file tersebut kemudian dipaketkan lagi menjadi sebuah file ZIP yang lebih besar. Untuk mengimpor aplikasi, pengguna harus mengekstrak file ZIP besar ini kemudian mengimpor aplikasi satu persatu.

4.6 Aplikasi GraphQL

Mulai versi 2.0.0, MagicAppBuilder dilengkapi dengan fitur untuk membuat aplikasi berbasis GraphQL berdasarkan entitas yang dibuat.

Jika aplikasi monolith dibuat permodul dengan konfigurasi yang berbeda-beda, aplikasi GraphQL dapat dibuat sekaligus. MagicAppBuilder akan membuat sebuah aplikasi GraphQL lengkap dalam beberapa detik saja. Pengguna dapat memilih entitas dan kolom mana yang akan digunakan oleh aplikasi GraphQL dan menentukan filter yang akan digunakan pada masing-masing entitas. Selain itu, pengguna juga dapat menentukan strategi pemberian nilai kunci utama pada masing-masing entitas.

Sebelum membuat aplikasi GraphQL, pengguna harus memastikan reserved column yang digunakan di aplikasi agar sesuai dengan penamaan kolom masing-masing tabel. Selain itu, pengguna juga harus memastikan relasi antar entitas mengikuti aturan pada MagicAppBuilder.

Pengguna dapat memilih bahasa pemrograman yang digunakan pada aplikasi GraphQL, yaitu sebagai berikut:

1. PHP
2. Java (versi profesional)
3. Kotlin (versi profesional)
4. Node.js (versi profesional)
5. TypeScript (versi profesional)
6. Python (versi profesional)
7. Go (versi profesional)

Untuk versi **PHP**, aplikasi dapat dijalankan secara langsung, bahkan dalam mode **offline**, tanpa perlu mengimpor pustaka tambahan. Untuk versi **Java** dan **Kotlin**, diperlukan **Java Runtime** versi **21** ke atas serta **Maven**. Sementara itu, untuk versi **Node.js**, **TypeScript**, **Python**, dan **Go**, pengguna perlu mengunduh dan memasang pustaka terlebih dahulu menggunakan perintah tertentu. Perintah untuk mengunduh pustaka disertakan di manual yang digenerate otomatis saat membuat aplikasi. Buka [manual.html](#) untuk melihat petunjuk bagaimana menjalankan dan menggunakan aplikasi.

Pengguna dapat menyimpan konfigurasi aplikasi GraphQL ke dalam beberapa profil. Setiap profil disimpan dengan nama unik dan memuat detail konfigurasi berikut:

- **Bahasa Pemrograman:** Bahasa backend yang dipilih (PHP, Java, Kotlin, Node.js, Python, atau Go).
- **Pemilihan Entitas:** Entitas tertentu yang disertakan dalam aplikasi.
- **Pemilihan Kolom:** Kolom yang dipilih untuk setiap entitas.

- **Konfigurasi Filter:** Filter yang didefinisikan untuk setiap entitas.
- **Strategi Pembuatan Primary Key:** Cara primary key ditangani (autogenerated, manual-insert, atau manual-all).

Dengan profil, pengguna dapat menghasilkan beberapa aplikasi yang berbeda menggunakan berbagai bahasa pemrograman dari skema database yang sama. Jika pengguna perlu memperbarui sebuah aplikasi, pengguna cukup memilih profilnya dan menghasilkannya kembali tanpa harus mengonfigurasi semuanya dari awal.

GraphQL Generator

Profile

GraphQL Application

Manage

Programming Language

PHP

☒ Custom Entities
 ☐ System Entities
 ☐ In-Memory Cache

☒ anggota

	Column	Type	PK	Serial	Null	TA	Filter	PK Value
<input checked="" type="checkbox"/>	anggota_id	VARCHAR(40)	YES	NO	NO		Not Applicable	Autogenerated
<input checked="" type="checkbox"/>	nama	VARCHAR(50)	NO	NO	YES	<input type="checkbox"/>	Text Contains	
<input checked="" type="checkbox"/>	nomor_anggota	VARCHAR(50)	NO	NO	YES	<input type="checkbox"/>	Not Applicable	
<input checked="" type="checkbox"/>	password	VARCHAR(255)	NO	NO	YES	<input type="checkbox"/>	Not Applicable	
<input checked="" type="checkbox"/>	pin	VARCHAR(255)	NO	NO	YES	<input type="checkbox"/>	Not Applicable	
<input checked="" type="checkbox"/>	jenis_keanggotaan_id	VARCHAR(40)	NO	NO	YES		Select Equals	
<input checked="" type="checkbox"/>	status_keanggotaan_id	VARCHAR(40)	NO	NO	YES		Select Equals	
<input checked="" type="checkbox"/>	alamat	VARCHAR(100)	NO	NO	YES	<input type="checkbox"/>	Not Applicable	
<input checked="" type="checkbox"/>	jenis_kelamin	VARCHAR(2)	NO	NO	YES	<input type="checkbox"/>	Not Applicable	
<input checked="" type="checkbox"/>	telepon	VARCHAR(15)	NO	NO	YES	<input type="checkbox"/>	Not Applicable	
<input checked="" type="checkbox"/>	email	VARCHAR(100)	NO	NO	YES	<input type="checkbox"/>	Not Applicable	
<input checked="" type="checkbox"/>	foto	VARCHAR(500)	NO	NO	YES	<input type="checkbox"/>	Not Applicable	

Generate GraphQL

Generate GraphQL with Frontend

Cancel

Untuk membuat aplikasi GraphQL, ikuti langkah-langkah sebagai berikut:

8. Pilih tombol Database di kartu aplikasi.
9. Setelah halaman Database Explorer tampil, pilih tombol Entity Editor.
10. Setelah halaman Entity Editor tampil, pilih tombol GraphQL di bagian Export.
11. Setelah dialog GraphQL Generator tampil, pilih kelompok entitas yang akan dibuat aplikasi GraphQL. Secara default, Custom Entity akan dipilih sedangkan System Entity tidak.
12. Dari setiap entitas yang dipilih, pilih kolom yang akan digunakan di aplikasi GraphQL.

13. Dari setiap entitas dan kolom yang dipilih, pengguna dapat mencentang TA (jika tersedia) jika pengguna ingin menggunakan TEXTAREA, yaitu input teks multi baris.
14. Dari setiap entitas dan kolom yang dipilih, tentukan filter untuk masing-masing halaman List. Secara default, field yang mewakili name (display field) akan dipilih sebagai filter dengan tipe Text Contains sedangkan field dengan akhiran “_id” akan dipilih sebagai filter dengan tipe Select Equals. Pengguna dapat menambahkan atau menghapus filter ini.
15. Di bagian bawah, ada 3 tombol yaitu Generate GraphQL, Generate GraphQL with Frontend dan Canel. Pilih Generate GraphQL with Frontend jika ingin membuat aplikasi GraphQL lengkap dengan frontend, pilih Generate GraphQL jika hanya ingin membuat API (Application Programming Interface) tanpa frontend.
16. Aplikasi akan mengunduh sebuah file ZIP. Simpan ke lokasi yang diinginkan.
17. Ekstrak file ZIP tersebut ke lokasi yang diinginkan.
18. Masuk ke direktori di mana ZIP tersebut diekstrak.
19. Untuk membaca dokumentasi dari API, buka file manual.html menggunakan browser.
20. Secara default, konfigurasi database di file database.php sama dengan aplikasi yang dibuat di MagicAppBuilder. Sesuaikan dengan kebutuhan.
21. Akses aplikasi menggunakan browser sesuai dengan path dari aplikasi.
22. Pilih entitas yang ada di sebelah kiri halaman aplikasi untuk mulai menggunakan.

4.7 Siklus Hidup Pengembangan Perangkat Lunak

4.7.1 Pengembangan Fitur Aplikasi

Setelah aplikasi berjalan di lingkungan produksi, pengguna mungkin memerlukan penambahan atau perubahan fitur. Perubahan ini sepenuhnya dilakukan di lingkungan 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 deployment fitur baru. Struktur database bayangan dari lingkungan 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.

4.1.1 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:

6. Mempercepat Proses

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

7. 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.

4.1.2 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. **Penskalaan 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. **Penskalaan 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.

4.1.3 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.

4.1.4 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.

4.2 Command Line Interface (CLI)

Command Line Interface (CLI) adalah cara untuk berinteraksi dengan komputer menggunakan teks, bukan antarmuka grafis (GUI) yang umum. Pengguna mengetikkan perintah ke dalam antarmuka berbasis teks untuk menjalankan tugas, mengelola sistem, atau menjalankan program.

MagicServer merupakan gabungan beberapa sistem yang dipaketkan menjadi satu kesatuan sistem untuk menjalankan MagicAppBuilder dan aplikasi yang dibuat pada lingkungan pengembangan. Ada beberapa subsistem dari MagicServer yang *memiliki command line* interface yaitu sebagai berikut:

1. php

PHP (Hypertext Preprocessor) adalah bahasa pemrograman sisi server yang umum digunakan untuk pengembangan web. Selain digunakan pada web server, PHP juga dapat dijalankan melalui command line interface (CLI) untuk keperluan skrip otomatisasi dan pemrosesan non-web.

Saat pengguna menjalankan script PHP untuk instalasi, sebenarnya pengguna sedang menjalankan PHP melalui command line interface dan memasukkan path dari skrip PHP sebagai parameternya. Di dalam script juga terdapat command line untuk menjalankan file executable lain seperti mariadb-install-db.exe, httpd.exe, mysqld.exe, redis-server.exe, dan lain-lain.

Pengguna dapat menjalankan file php.exe pada command line untuk berbagai keperluan. Karena *installer* MagicServer tidak memasukkan direktori dari php.exe ke environment variable **PATH**, maka pengguna harus menulis path php.exe secara lengkap baik secara absolut maupun relatif.

Jika pengguna berada di direktori dari MagicServer, maka path relatif dari php.exe adalah:

php\php.exe

Pengguna dapat mengetik perintah dari direktori MagicServer dengan

php\php {argumen}

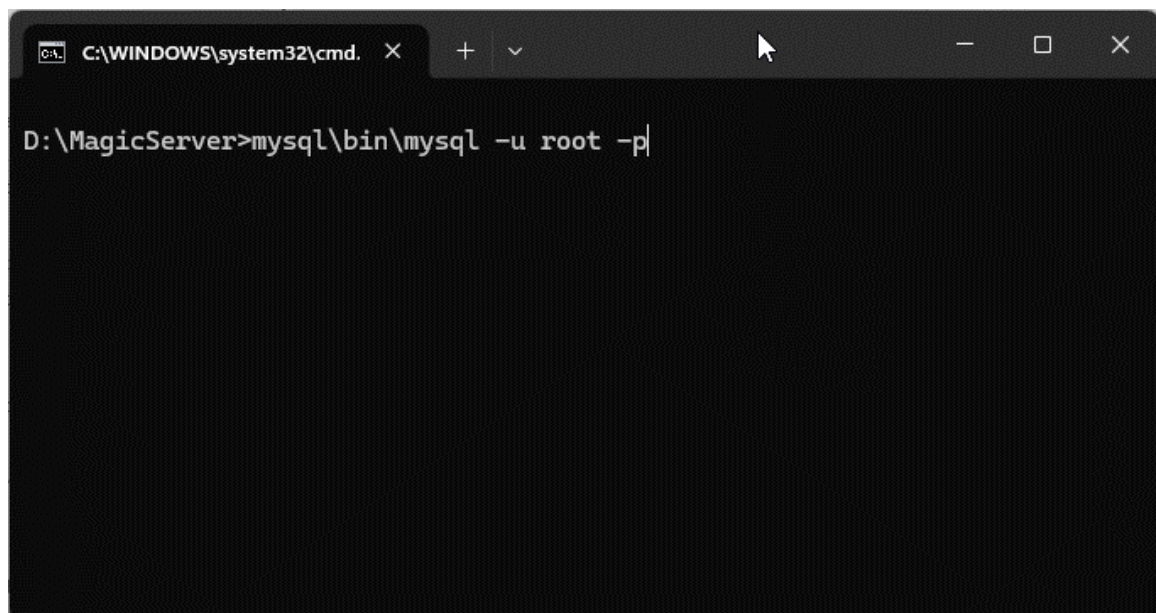
2. mysql

MariaDB adalah sistem manajemen basis data relasional (RDBMS) yang merupakan turunan dari MySQL dan sepenuhnya kompatibel secara fungsional. Dalam MagicServer, MariaDB digunakan untuk menyimpan dan mengelola data dari aplikasi yang dibuat menggunakan MagicAppBuilder.

MagicServer menyertakan executable seperti mysqld.exe (server MariaDB) dan mysql.exe (client CLI MariaDB). Melalui command line, pengguna dapat menjalankan berbagai perintah SQL langsung ke server database. Karena direktori MariaDB tidak dimasukkan ke dalam environment variable PATH, maka pengguna perlu menyebutkan path lengkap ke mysql.exe jika ingin mengaksesnya.

Contoh perintah dari direktori MagicServer:

mysql\bin\mysql -u root -p



Masukkan password dari root. Jika Anda belum pernah mengganti password secara manual setelah instalasi, password dari root adalah “password” tanpa tanda kutip. Setelah Anda berhasil masuk, Anda dapat menjalankan SQL pada command line interface.

3. redis-cli

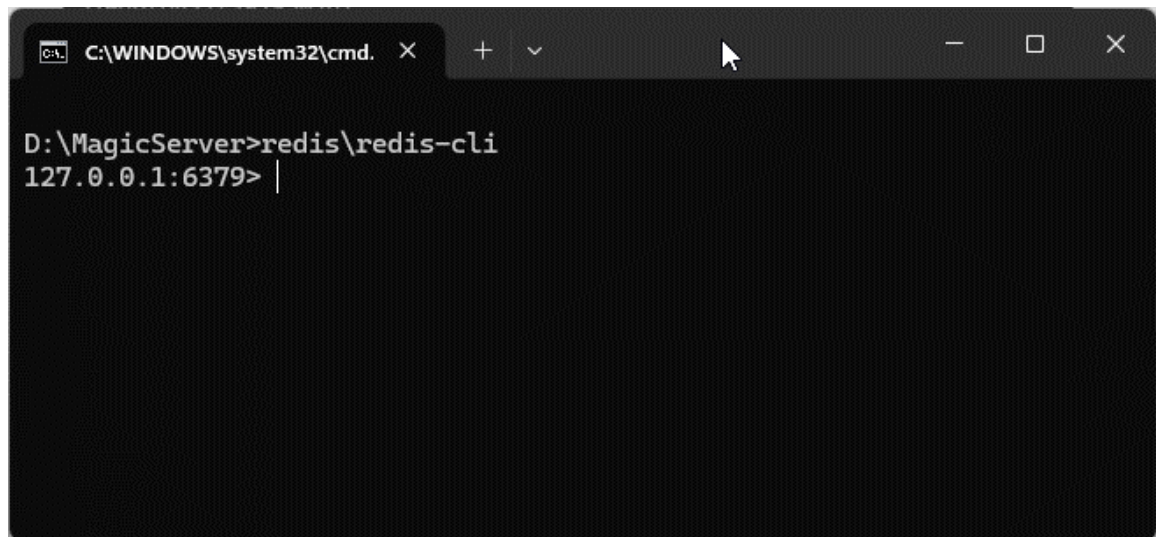
redis-cli adalah command line interface untuk Redis, yaitu sistem penyimpanan data dalam memori (in-memory key-value store) yang sangat cepat dan sering digunakan untuk caching, manajemen sesi, dan antrian pesan.

Dalam MagicServer, Redis digunakan terutama untuk menyimpan data sesi aplikasi dan proses caching yang membutuhkan akses cepat. Program redis-server.exe akan menjalankan server Redis, sementara redis-cli.exe digunakan untuk mengakses Redis secara langsung melalui command line.

Melalui redis-cli, pengguna dapat melihat isi key Redis, melakukan query, atau menghapus data tertentu secara manual.

Jika pengguna ingin menjalankan redis-cli dari direktori MagicServer, perintahnya dapat ditulis seperti berikut:

redis\redis-cli



Contoh penggunaan untuk melihat semua key yang tersimpan:

KEYS *

```
C:\WINDOWS\system32\cmd. X + - □ X
D:\MagicServer>redis\redis-cli
127.0.0.1:6379> KEYS *
1) "PHPREDIS_SESSION:0nbs2i6ikjia4jf7hu38vclanc"
127.0.0.1:6379> |
```

Untuk melihat isi dari sebuah key yang tersimpan:

GET session:abc123

```
C:\WINDOWS\system32\cmd. X + - □ X
D:\MagicServer>redis\redis-cli
127.0.0.1:6379> KEYS *
1) "PHPREDIS_SESSION:0nbs2i6ikjia4jf7hu38vclanc"
127.0.0.1:6379> GET PHPREDIS_SESSION:0nbs2i6ikjia4jf7hu38vclanc
"username|s:9:\"superuser\";userPassword|s:40:\"8e67bb26b358e2ed20fe552ed6fb832f397a507d\";"
127.0.0.1:6379> |
```

Atau untuk menghapus key tertentu:

DEL session:abc123

```
C:\WINDOWS\system32\cmd. X + v - □ X
D:\MagicServer>redis\redis-cli
127.0.0.1:6379> KEYS *
1) "PHPREDIS_SESSION:0nbs2i6ikjia4jf7hu38vclanc"
127.0.0.1:6379> GET PHPREDIS_SESSION:0nbs2i6ikjia4jf7hu38vclanc
"username|s:9:\"superuser\";userPassword|s:40:\"8e67bb26b358e2ed20fe552ed6fb832f397a507d\";"
127.0.0.1:6379> DEL PHPREDIS_SESSION:0nbs2i6ikjia4jf7hu38vclanc
(integer) 1
127.0.0.1:6379> |
```

Untuk keluar dari redis-cli, ketik perintah:

QUIT -

```
C:\WINDOWS\system32\cmd. X + v - □ X
D:\MagicServer>redis\redis-cli
127.0.0.1:6379> KEYS *
1) "PHPREDIS_SESSION:0nbs2i6ikjia4jf7hu38vclanc"
127.0.0.1:6379> GET PHPREDIS_SESSION:0nbs2i6ikjia4jf7hu38vclanc
"username|s:9:\"superuser\";userPassword|s:40:\"8e67bb26b358e2ed20fe552ed6fb832f397a507d\";"
127.0.0.1:6379> DEL PHPREDIS_SESSION:0nbs2i6ikjia4jf7hu38vclanc
(integer) 1
127.0.0.1:6379> QUIT -

D:\MagicServer>|
```

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.