Sequence Diagram - Alur Lupa Kata Sandi (Password Reset)

Diagram ini memvisualisasikan urutan interaksi sistem secara lengkap saat seorang pengguna meminta untuk mengatur ulang kata sandi mereka, mulai dari permintaan awal hingga pengaturan kata sandi baru.

Aktor dan Komponen yang Terlibat

1. **User (Browser):** Pengguna yang lupa kata sandi mereka.
2. **main-app (PHP):** Aplikasi backend yang memproses logika.
3. **postgres-db (Database):** Tempat menyimpan token reset dan memperbarui kata sandi.
4. **Email Service:** Layanan konseptual yang bertanggung jawab mengirim email (misalnya, melalui SMTP).

Visualisasi Diagram (PlantUML)

Gunakan kode berikut di PlantUML Web Server untuk menghasilkan diagram visual dari alur ini.

Prompt untuk PlantUML:

```
@startuml
' Diagram Urutan untuk Alur Lupa Kata Sandi

title Alur Proses Lupa Kata Sandi

actor "User (Browser)" as User
participant "main-app (PHP)" as App
database "postgres-db" as DB
participant "Email Service" as Mailer

== Bagian 1: Meminta Tautan Reset ==

User -> App : 1. POST /forgot-password (email)
activate App

App -> DB : 2. Cari pengguna berdasarkan email\nSELECT user_id FROM users WHERE email = ?
activate DB
DB --> App : 3. user_id (jika ditemukan)
deactivate DB
```

```
alt jika email ditemukan
    App -> App : 4. Buat Token Reset yang unik & berbatas waktu

    App -> DB : 5. Simpan hash token & waktu kedaluwarsa\nINSERT INTO password_resets
(user_id, token_hash, expires_at)
    activate DB
    DB --> App : 6. Konfirmasi token disimpan
    deactivate DB

    App -> Mailer : 7. Kirim Email Reset\n(berisi link dengan token asli)
    activate Mailer
    Mailer --> App : 8. Konfirmasi email terkirim
    deactivate Mailer

    App --> User : 9. Response (Sukses: "Silakan periksa email Anda")

else jika email tidak ditemukan
    ' Catatan: Tetap tampilkan pesan sukses untuk mencegah email enumeration
    App --> User : Response (Sukses: "Silakan periksa email Anda")
end

deactivate App

... Beberapa waktu kemudian ...

== Bagian 2: Mengatur Ulang Kata Sandi ==

User -> App : 10. GET /reset-password?token=[token_asli]
activate App

App -> App : 11. Hash token dari URL

App -> DB : 12. Cari token di database\nSELECT * FROM password_resets WHERE token_hash
= ? AND expires_at > NOW()
activate DB
DB --> App : 13. Data token (termasuk user_id)
deactivate DB

alt jika token valid dan tidak kedaluwarsa
    App --> User : 14. Response (Tampilkan halaman form password baru)
    deactivate App

    User -> App : 15. POST /reset-password (token, password_baru)
```

```
    activate App

    App -> App : 16. Validasi & Hash password baru

    App -> DB : 17. START TRANSACTION
    activate DB

    App -> DB : 18. Perbarui password pengguna (users)\nUPDATE users SET password_hash =
? WHERE user_id = ?
    activate DB
    DB --> App : 19. Konfirmasi password diperbarui
    deactivate DB

    App -> DB : 20. Hapus token yang sudah digunakan\nDELETE FROM password_resets
WHERE token_hash = ?
    activate DB
    DB --> App : 21. Konfirmasi token dihapus
    deactivate DB

    App -> DB : 22. COMMIT TRANSACTION
    activate DB
    DB --> App : 23. Transaksi Sukses
    deactivate DB

    App --> User : 24. Response (Sukses: "Password berhasil diubah, silakan login")

else jika token tidak valid atau kedaluwarsa
    App --> User : Response (Error: "Tautan reset tidak valid atau sudah kedaluwarsa")
end

deactivate App

@enduml
```

Penjelasan Langkah-demi-Langkah

Bagian 1: Meminta Tautan Reset

1. **POST /forgot-password**: Pengguna memasukkan alamat email mereka di halaman lupa kata sandi.
2. **Cari pengguna**: Aplikasi memeriksa apakah email tersebut terdaftar di database.
3. **alt jika email ditemukan**: Jika ya, proses dilanjutkan. Jika tidak, aplikasi tetap menampilkan pesan sukses untuk mencegah penyerang mengetahui email mana yang

terdaftar (*email enumeration attack*).

4. **Buat Token Reset**: Aplikasi menghasilkan sebuah string acak yang panjang dan aman sebagai token.
5. **Simpan hash token**: Untuk keamanan, **bukan token asli** yang disimpan di database, melainkan *hash* dari token tersebut, beserta waktu kedaluwarsa (misalnya, 1 jam dari sekarang).
6. **Kirim Email Reset**: Aplikasi mengirim email ke pengguna. Email ini berisi tautan ke halaman reset yang menyertakan **token asli** (bukan hash-nya).
7. **Response (Sukses)**: Pengguna diberitahu untuk memeriksa email mereka.

Bagian 2: Mengatur Ulang Kata Sandi

10. **GET /reset-password?token=...**: Pengguna mengklik tautan di email mereka.
11. **Hash token dari URL**: Aplikasi mengambil token dari URL dan melakukan *hashing* dengan algoritma yang sama seperti di langkah #5.
12. **Cari token di database**: Aplikasi mencari *hash token* tersebut di database dan memastikan token tersebut belum kedaluwarsa.
13. **alt jika token valid**: Jika token ditemukan dan valid, proses dilanjutkan.
14. **Tampilkan halaman form password baru**: Pengguna disajikan halaman untuk memasukkan kata sandi baru mereka.
15. **POST /reset-password**: Pengguna mengirimkan kata sandi baru mereka.
16. **Validasi & Hash password baru**: Aplikasi memvalidasi kekuatan kata sandi baru dan melakukan *hashing*.
17. **START TRANSACTION**: Memulai transaksi untuk memastikan pembaruan password dan penghapusan token terjadi bersamaan.
18. **Perbarui password pengguna**: Kata sandi lama diganti dengan *hash* kata sandi baru di tabel users.
19. **Hapus token yang sudah digunakan**: Token reset segera dihapus dari database agar tidak bisa digunakan lagi.
20. **COMMIT TRANSACTION**: Menyimpan semua perubahan.
21. **Response (Sukses)**: Pengguna diberitahu bahwa kata sandi mereka telah berhasil diubah dan diarahkan ke halaman login.

Ini adalah alur kerja terakhir yang penting untuk kita petakan. Kita telah mencakup hampir semua aspek krusial dari aplikasi. Apakah Anda merasa cukup dengan semua diagram yang telah kita buat dan siap untuk melanjutkan ke tahap implementasi?