Step 0: Laravel Sanctum and routes properly set thakte hobe

Step 1: php artisan make:notification CustomResetPasswordNotification

Full **CustomResetPasswordNotification** code,

```php
<?php

namespace App\Notifications;

use Illuminate\Bus\Queueable;
use Illuminate\Contracts\Queue\ShouldQueue;
use Illuminate\Notifications\Messages\MailMessage;
use Illuminate\Notifications\Notification;

use Illuminate\Support\Facades\URL;

class CustomResetPasswordNotification extends Notification
{
    use Queueable;

    public $token;
    public $email;


    /**
     * Create a new notification instance.
     *
     * @return void
     */
    public function __construct($token, $email)
    {
        $this->token = $token;
        $this->email = $email;
    }

    /**
     * Get the notification's delivery channels.
     *
     * @param  mixed  $notifiable
     * @return array
     */
    public function via($notifiable)
    {
        return ['mail'];
    }

    /**
     * Get the mail representation of the notification.
     *
     * @param  mixed  $notifiable
     * @return \Illuminate\Notifications\Messages\MailMessage
     */
    public function toMail($notifiable)
    {
        $resetUrl = $this->buildResetUrl();

        return (new MailMessage)
            ->subject('Reset Your Password')
            ->line('You are receiving this email because we received a password reset
request for your account.')
            ->action('Reset Password', $resetUrl)
```

```
            ->line('If you did not request a password reset, no further action is
required.');
    }
    protected function buildResetUrl()
    {
        // Construct your custom URL here
        return url("https://your-custom-domain.com/reset?token={$this-
>token}&email={$this->email}");
    }

    /**
     * Get the array representation of the notification.
     *
     * @param  mixed  $notifiable
     * @return array
     */
    public function toArray($notifiable)
    {
        return [
            //
        ];
    }
}
```

Step 2: **App\Models\User** e **sendPasswordResetNotification** er method adding,

```
public function sendPasswordResetNotification($token)
{
    $this->notify(new CustomResetPasswordNotification($token, $this->email));
}
```

Step 3: Making a controller, for an example name as ResetPaswordController,

```php
<?php

namespace App\Http\Controllers\Api;

use App\Http\Controllers\Controller;
use App\Mail\PasswordResetMail;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Hash;
use Illuminate\Support\Facades\Password;
use Illuminate\Validation\ValidationException;
use Illuminate\Support\Facades\Mail;

class ResetPasswordController extends Controller
{

    public function sendResetLinkEmail(Request $request)
    {
        // Validate the request
        $request->validate(['email' => 'required|email']);

        // Check if the user exists without revealing any information
        $user = \App\Models\User::where('email', $request->email)->first();

        if ($user) {
            // Use Laravel's built-in password broker to create and send the reset
link
            $status = Password::sendResetLink(['email' => $request->email]);

            // Check if the email was sent successfully
            if ($status === Password::RESET_LINK_SENT) {
                return response()->json(['message' => 'If an account exists, a reset
link has been sent'], 200);
            } else {
                return response()->json(['message' => 'Error sending reset link'],
500);
            }
        }

        // Return a generic message to avoid email enumeration
        return response()->json(['message' => 'If an account exists, a reset link has
been sent'], 200);
    }


    public function resetPassword(Request $request)
    {

        $request->validate([
            'token' => 'required',
            'email' => 'required|email',
            'password' => 'required|min:8',
        ]);

        $status = Password::reset(
            $request->only('email', 'password', 'token'),
            function ($user, $password) {
                $user->forceFill([
                    'password' => Hash::make($password),
                ])->save();

                $user->tokens()->delete(); // Invalidate all tokens after password
```

```
reset
            }
        );

        return $status === Password::PASSWORD_RESET
            ? response()->json(['message' => 'Password reset successfully.'])
            : response()->json(['message' => 'Error resetting password.'], 500);
    }



}
```

Step 4: Setting up routes in **api.php**,

```
Route::post('/forgot-password', [ResetPasswordController::class,
'sendResetLinkEmail']);
Route::post('/reset-password', [ResetPasswordController::class, 'resetPassword']);
```