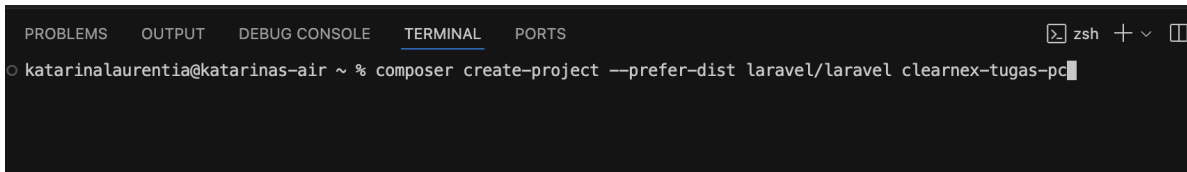


A. Langkah-Langkah pembuatan Aplikasi

1. Create Project Laravel, cd clearnex-tugas-pc

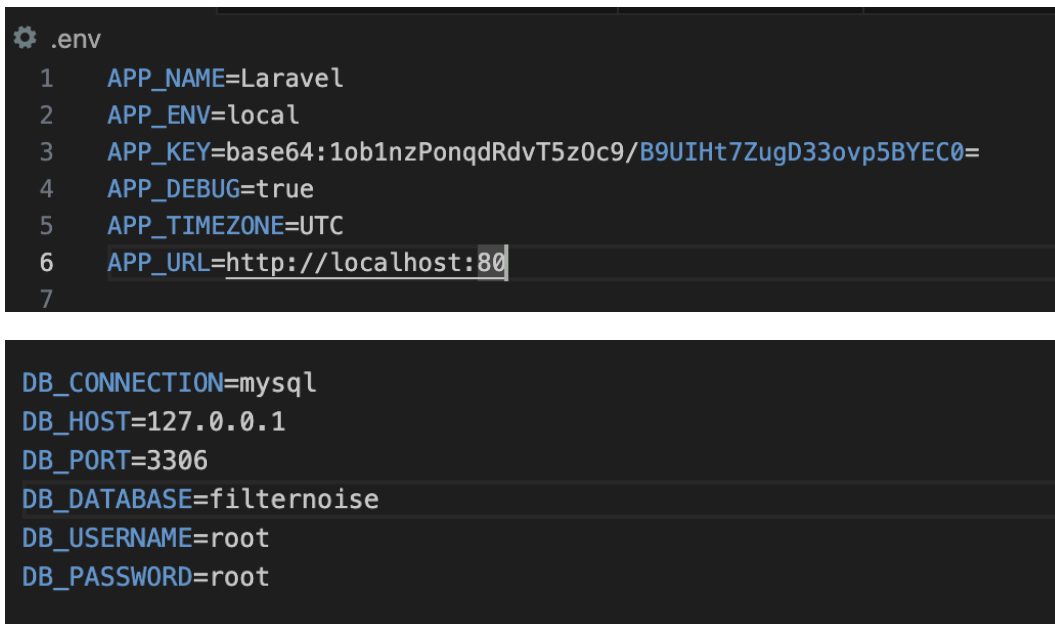


```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
katarinalaurentia@katarinas-air ~ % composer create-project --prefer-dist laravel/laravel clearnex-tugas-pc
```

2. Create Database di halaman phpMyAdmin



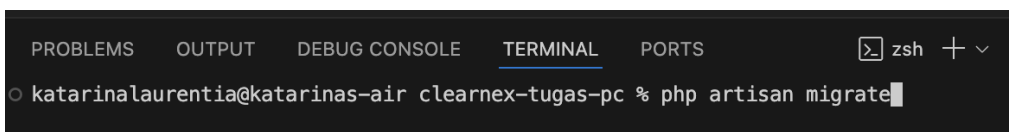
3. Lakukan konfigurasi pada file .env



```
.env
1 APP_NAME=Laravel
2 APP_ENV=local
3 APP_KEY=base64:1ob1nzPonqdRdvT5z0c9/B9UIHt7ZugD33ovp5BYEC0=
4 APP_DEBUG=true
5 APP_TIMEZONE=UTC
6 APP_URL=http://localhost:80
7

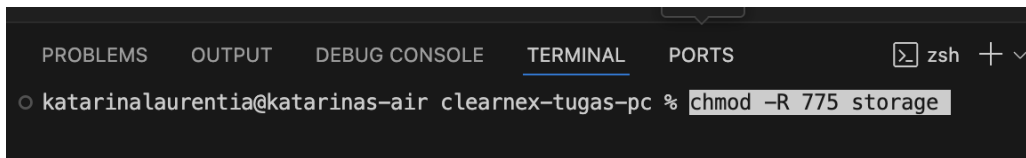
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=filternoise
DB_USERNAME=root
DB_PASSWORD=root
```

4. Jalankan perintah “Php artisan migrate”



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
katarinalaurentia@katarinas-air clearnex-tugas-pc % php artisan migrate
```

5. Jalankan perintah “chmod -R 775 storage”



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS [x] zsh + v
○ katarinalaurentia@katarinas-air clearnex-tugas-pc % chmod -R 775 storage
```

6. Buat Folder Public/storage/upload untuk menyimpan gambar hasil upload
7. Menjalankan perintah “php artisan storage:link”
8. Buat controller dengan menjalankan perintah “php artisan make:controller ImageController” dan isi dengan perintah berikut

```
<?php
namespace App\Http\Controllers;

use Illuminate\Http\Request;
use Illuminate\Support\Facades\Storage;
use Symfony\Component\Process\Process;
use Symfony\Component\Process\Exception\ProcessFailedException;

class ImageProcessingController extends Controller
{
    // Menampilkan form upload gambar
    public function index()
    {
        return view('image-upload');
    }

    // Proses gambar yang diupload
    public function processImage(Request $request)
    {
        // Validasi file
        $request->validate([
            'image' => 'required|image|mimes:jpeg,png,jpg|max:2048',
        ]);

        // Menyimpan gambar yang diupload
        $image = $request->file('image');
        $imagePath = $image->store('uploads', 'public');

        // Path gambar yang disimpan
```

```

        $fullImagePath = storage_path('app/public/' . $imagePath);
        set_time_limit(60);

        // Jalankan Python Script untuk memproses gambar
        $process = new Process(['python3', base_path('process_image.py'),
        $fullImagePath]);
        $process->run();

        // Jika gagal
        if (!$process->isSuccessful()) {
            throw new ProcessFailedException($process);
        }

        // Ambil output dari Python script (gambar yang sudah diproses)
        $output = $process->getOutput();

        return view('image-result', [
            'originalImage' => asset('storage/' . $imagePath),
            'outputImage' => asset('storage/processed_image.jpg'),
        ]);
    }
}

```

Penjelasan Kode Program

1. namespace App\Http\Controllers; Menyatakan bahwa ImageProcessingController berada di dalam namespace App\Http\Controllers.
2. use Illuminate\Http\Request; Memuat kelas Request dari Laravel yang berfungsi untuk menangani input HTTP.
3. use Illuminate\Support\Facades\Storage; Memuat Storage yang memungkinkan penyimpanan file di sistem.
4. use Symfony\Component\Process\Process; Memuat kelas Process dari Symfony untuk menjalankan perintah eksternal (seperti menjalankan skrip Python).
5. use Symfony\Component\Process\Exception\ProcessFailedException; Memuat ProcessFailedException untuk menangani error ketika perintah eksternal gagal.
6. Method index(), digunakan untuk menampilkan form upload gambar.
7. Method processImage(), Method ini bertanggung jawab untuk memproses gambar yang diupload
8. Menggabungkan storage_path dengan \$imagePath untuk mendapatkan path lengkap file.
9. set_time_limit(60); memberi batas waktu eksekusi skrip selama 60 detik untuk

mencegah timeout.

10. Membuat objek Process yang menjalankan skrip Python process_image.py dengan path lengkap dari gambar sebagai argumen.
11. run() menjalankan proses tersebut.
12. Mengecek apakah proses Python berhasil. Jika gagal, melempar ProcessFailedException yang akan memberikan pesan error.
13. getOutput() digunakan untuk mengambil hasil output dari skrip Python
14. Mengirim data gambar asli dan gambar yang sudah diproses ke tampilan image-result. Atribut asset membuat URL untuk mengakses gambar yang disimpan di storage.

9. Buat file process_image.py yang berisi perintah/filter domain spasial yang digunakan

```
import cv2
import numpy as np
from matplotlib import pyplot as plt
from matplotlib.widgets import Button
import sys
import os

# Mendapatkan path gambar dari argumen
image_path = sys.argv[1]

# Membaca citra yang diupload
image = cv2.imread(image_path, cv2.IMREAD_GRAYSCALE)

# Menerapkan filter rata-rata (average filter) untuk mengurangi noise
kernel_size = (5, 5)
average_filtered_image = cv2.blur(image, kernel_size)

# Menerapkan filter median untuk mengurangi noise
median_filtered_image = cv2.medianBlur(image, 5)

# Simpan gambar hasil proses
average_image_path = 'storage/app/public/average_filtered_image.jpg'
median_image_path = 'storage/app/public/median_filtered_image.jpg'
cv2.imwrite(average_image_path, average_filtered_image)
cv2.imwrite(median_image_path, median_filtered_image)

# Fungsi untuk mengunduh gambar (membuka file hasil)
def download_average(event):
```

```

os.startfile(average_image_path) # Opens the file, simulating a download

def download_median(event):
    os.startfile(median_image_path) # Opens the file, simulating a download

# Menampilkan hasil
fig, axs = plt.subplots(1, 3, figsize=(15, 8))

# Tampilkan gambar asli
axs[0].imshow(image, cmap='gray')
axs[0].set_title('Original Image with Noise')
axs[0].axis('off')

# Tampilkan gambar hasil average filtering
axs[1].imshow(average_filtered_image, cmap='gray')
axs[1].set_title('Average Filtered Image')
axs[1].axis('off')

# Tampilkan gambar hasil median filtering
axs[2].imshow(median_filtered_image, cmap='gray')
axs[2].set_title('Median Filtered Image')
axs[2].axis('off')

plt.show()

```

Penjelasan Kode Program

1. **cv2, numpy, pyplot**, dan **Button** dari **matplotlib** digunakan untuk memproses citra, menampilkan hasil, dan membuat tombol interaktif.
2. **sys** dan **os** digunakan untuk mendapatkan path gambar dari argumen dan membuka file hasil.
3. **image_path = sys.argv[1]** mengambil path gambar dari argumen pertama yang diberikan saat menjalankan skrip ini.
4. **cv2.imread(image_path, cv2.IMREAD_GRAYSCALE)** membaca citra dalam mode grayscale (hitam putih) agar lebih mudah diproses.
5. Menggunakan **cv2.blur(image, kernel_size)**, di mana **kernel_size = (5, 5)**, untuk menerapkan filter rata-rata (average filter) yang dapat mengurangi noise dengan meratakan nilai piksel dalam area tertentu.
6. Menggunakan **cv2.medianBlur(image, 5)** untuk menerapkan filter median yang efektif dalam mengurangi noise tipe salt-and-pepper (bintik-bintik hitam dan putih).
7. **cv2.imwrite()** digunakan untuk menyimpan citra hasil filter rata-rata (**average_filtered_image.jpg**) dan filter median (**median_filtered_image.jpg**) di dalam folder **storage/app/public**.

8. Dua fungsi, `download_average` dan `download_median`, dibuat untuk membuka file hasil, yang mensimulasikan proses pengunduhan file.
9. Menggunakan `plt.subplots()` untuk membuat tata letak 3 kolom
10. `plt.show()` menampilkan antarmuka hasil, di mana tiap gambar diberi judul yang sesuai.

10. Membuat file `landing.blade.php` sebagai halaman `introduction/pengenalan` aplikasi

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Cleanex - Landing Page</title>
  <link rel="icon" href="{{ asset('images/logo.png') }}" type="image/png">

  <!-- Bootstrap CSS -->
  <link
href="https://cdn.jsdelivrivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css"
rel="stylesheet">

  <!-- Custom CSS -->
  <link rel="stylesheet" href="{{ asset('css/style.css') }}">

  <!-- Google Fonts -->
  <link
href="https://fonts.googleapis.com/css2?family=Poppins:wght@400;600&display=swap"
rel="stylesheet">
</head>
<body>
  <!-- Navbar -->
  <nav class="navbar">
    <div class="container">
      <a class="navbar-brand" href="#">
        Cleanex
      </a>
    </div>
  </nav>
```

```

<!-- Section 1 -->
<section class="section-one py-5">
  <div class="container">
    <div class="row align-items-center">
      <!-- Left Column (Image + Text) -->
      <div class="col-md-6 text-center column-1"> <!-- Added class column-1 -->
        

        <h1 class="image-text text-center mt-3 w-700">Remove Noise</h1>
        <h5 class="image-text mt-3 w-700">100% Automatically and Free</h5>
      </div>

      <!-- Right Column (Upload Box) -->
      <div class="col-md-6 column-2"> <!-- Added class column-2 -->
        <div class="upload-box text-center">
          <div class="upload-container">
            <a href="/image-upload" class="btn btn-primary custom-btn btn-lg
mb-3">Upload Image</a>

            <p class="text-muted">Upload image noise and see the magic</p>
          </div>

          <div class="example-images mt-4">
            <p>No image? Try one of these:</p>
            <div class="d-flex justify-content-center gap-2">
              
              
              
              
            </div>
          </div>

          <div class="terms mt-3">
            <p>By uploading an image or URL you agree to our <a
href="#">Terms of Service</a>. To learn more about how we handle personal data,
check our <a href="#">Privacy Policy</a>.</p>
          </div>
        </div>
      </div>
    </div>
  </div>

```

```

        </div>
    </div>
</div>
</section>
<!-- Section 2 -->
<section class="section-two py-5">
    <div class="container">
        <div class="row align-items-center">
            <!-- Left Column (Text) -->
            <div class="col-md-6">
                <h2 class="text-left">Seputar Domain Spasial</h2>
                <p class="mt-4">
                    Domain spasial merupakan salah satu konsep penting dalam
pengolahan citra yang merujuk pada representasi citra dalam bentuk koordinat
spasialnya. Dalam domain ini, pengolahan citra dilakukan dengan memanipulasi nilai
piksel citra secara langsung. Metode ini sering digunakan untuk teknik seperti
pemfilteran, penghalusan, dan deteksi tepi.

                    Dengan menggunakan transformasi domain spasial, kita dapat
meningkatkan kualitas citra dan mengekstrak fitur-fitur penting yang ada di
dalamnya. Pendekatan ini sangat efektif dalam berbagai aplikasi, termasuk pengenalan
wajah, analisis medis, dan pemantauan lingkungan.
                </p>
            </div>

            <!-- Right Column (Image) -->
            <div class="col-md-6 text-center">
                
            </div>
        </div>
    </div>
</section>
<!-- Section 3 -->
<section class="section-two py-5">
    <div class="container">
        <div class="row align-items-center">
            <!-- Left Column (Text) -->
            <div class="col-md-6 text-center">
                <img src="{{ asset('images/domainspasial.png') }}" alt="Domain
Spasial" class="img-fluid" style="border-radius: 10px; max-width: 100%; height:

```



```

auto;"/>

</div>

<!-- Right Column (Image) -->
<div class="col-md-6">
    <h2 class="text-left">Filter Median atau Filter Rata-Rata</h2>
    <p class="mt-4">
        Filter Median: Filter median menggantikan nilai setiap piksel
        dengan nilai median dari piksel-piksel di sekitarnya, efektif untuk menghilangkan
        noise tipe garam dan merica sambil mempertahankan tepi. Filter ini tahan terhadap
        outlier, sehingga ideal untuk aplikasi dalam praproses citra dan visi komputer.</p>
        <p>Filter Rata-rata: Filter rata-rata menggantikan nilai setiap
        piksel dengan rata-rata nilai piksel di sekitarnya, sehingga menghaluskan citra dan
        mengurangi noise Gaussian. Meskipun efektif untuk penghalusan dan pengurangan noise,
        filter ini dapat juga mengaburkan tepi dan detail halus.
    </p>
</div>
</div>
</section>
<section class="section-two py-5">
    <h2 class="text-center mb-5">Just Picture It</h2>
    <div class="row align-items-center">
        <div class="col-md-4 text-center">
            
            <p class="text-center">Image Noise</p>
        </div>
        <div class="col-md-4 text-center">
            
            <p class="text-center">Average Filter Image</p>
        </div>
        <div class="col-md-4 text-center">
            
            <p class="text-center">Median Filter Image</p>
        </div>
    </div>
</section>
<footer style="text-align: center; padding: 20px; background-color: #dcede1;">
    <p>&copy; 2024 Kelompok 1 | all content is reserved.</p>

```

```

</footer>

    <!-- Bootstrap JS (optional) -->
    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"><
/script>
</body>
</html>

```

11. Membuat image-upload.blade.php dan isi program berikut, untuk membuat tampilan halaman upload gambar

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Clearnex - image upload</title>
    <link rel="icon" href="{{ asset('images/logo.png') }}" type="image/png">

    <!-- Bootstrap CSS -->
    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css"
rel="stylesheet">

    <!-- Custom CSS -->
    <link rel="stylesheet" href="{{ asset('css/style.css') }}">

    <!-- Google Fonts -->
    <link
href="https://fonts.googleapis.com/css2?family=Poppins:wght@400;600&display=swap"
rel="stylesheet">
</head>
<body>
    <!-- Navbar -->
    <nav class="navbar">
        <div class="container">
            <a class="navbar-brand" href="#">

```

```

        Cleanex
    </a>
</div>
</nav>

<div class="container content-box text-center">
    <h1 class="mb-4">Upload any image with noise<br> and make it clear</h1>
    <form id="uploadForm" action="/process-image" method="POST"
enctype="multipart/form-data">
        @csrf
        <div class="mb-3">
            <input type="file" name="image" id="image" class="form-control"
required>
        </div>
        <button type="submit" class="btn btn-primary custom-btn">Proses
Gambar</button>
    </form>
</div>

<!-- Modal for showing processed image -->
<div class="modal fade" id="imageModal" tabindex="-1"
aria-labelledby="imageModalLabel" aria-hidden="true">
    <div class="modal-dialog modal-dialog-centered modal-lg">
        <div class="modal-content">
            <div class="modal-header">
                <h5 class="modal-title" id="imageModalLabel">Processed
Images</h5>
                <button type="button" class="btn-close" data-bs-dismiss="modal"
aria-label="Close"></button>
            </div>
            <div class="modal-body">
                <div class="row">
                    <div class="col-md-6">
                        <h6>Median Filter Result</h6>
                        <img id="medianImage" src="" alt="Median Filter Result"
class="img-fluid">
                    </div>
                    <div class="col-md-6">
                        <h6>Average Filter Result</h6>
                        <img id="averageImage" src="" alt="Average Filter Result"

```

```
class="img-fluid">
    </div>
</div>
</div>
</div>
</div>
</div>

<!-- Bootstrap JS -->
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js">
</script>
</body>
</html>
```

Penjelasan Kode Program

1. **Struktur Dasar:** Halaman HTML untuk mengunggah dan memproses gambar dengan pengurangan noise, menggunakan HTML5.
2. **Bagian `<head>`:**
 - **Meta Tags:** Mengatur bahasa, encoding karakter, dan viewport untuk tampilan responsif.
 - **Bootstrap CSS:** Menyertakan Bootstrap untuk tata letak responsif.
 - **CSS Kustom:** Menambahkan gaya khusus dari `style.css`.
 - **Google Fonts:** Menggunakan font *Poppins* untuk tampilan teks.
3. **Navbar:**
 - Menampilkan logo dan teks "Cleanex" sebagai nama aplikasi pada navbar sederhana.
4. **Konten Utama:**
 - **Judul dan Instruksi:** Mengajak pengguna untuk mengunggah gambar dengan noise untuk dibersihkan.
 - **Formulir Unggah:**
 - Metode **POST** mengirimkan data gambar ke endpoint `/process-image`.
 - Memerlukan input file (gambar) yang wajib diisi dan tombol "Proses Gambar".
5. **Modal untuk Hasil Gambar:**
 - Menampilkan gambar yang diproses setelah diunggah.
 - Dua bagian untuk hasil gambar: **Median Filter** dan **Average Filter**.
6. **Bootstrap JS:**
 - Menyertakan pustaka JavaScript Bootstrap untuk mengaktifkan komponen dinamis, seperti modal.

12. Buat file style.css untuk melakukan styling agar tampilan lebih menarik

13. Buat route pada web.php

```
<?php

use Illuminate\Support\Facades\Route;
use App\Http\Controllers\ImageUploadController;
use App\Http\Controllers\ImageProcessingController;

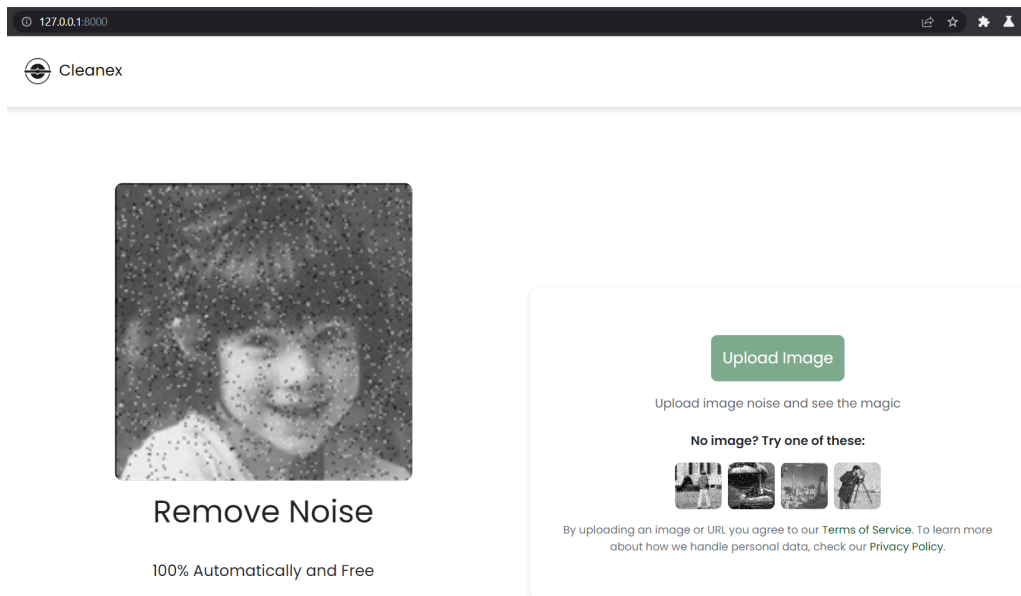
Route::get('/image-upload', [ImageProcessingController::class, 'index']);
Route::post('/process-image', [ImageProcessingController::class, 'processImage']);
Route::get('/', function () {
    return view('landing'); // Mengarahkan ke file Blade 'landing.blade.php'
});
```

14. Jalan perintah “Php Artisan Route:cache”

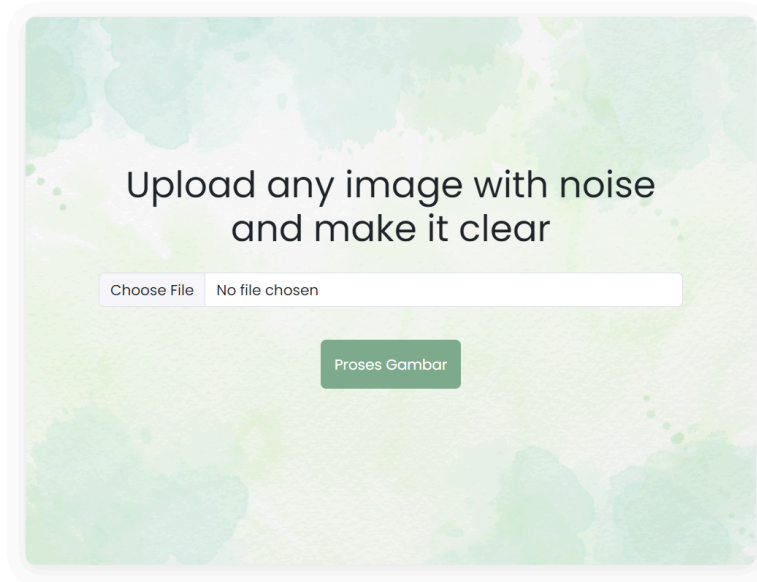
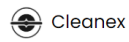
```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  [icon] zsh + v
katarinalaurentia@katarinas-air clearnex-tugas-pc % php artisan route:cache
```

B. Panduan Penggunaan aplikasi

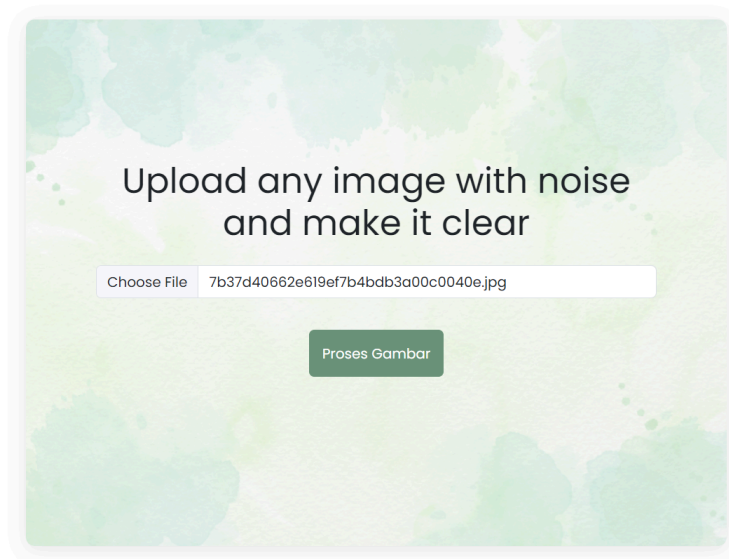
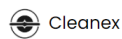
1. Akses <http://127.0.0.1:8000/> untuk mengakses landing page >> klik Upload image



2. Pada halaman <http://127.0.0.1:8000/image-upload>, pilih gambar dengan noise yang akan di clear noise

A screenshot of a web application interface for image upload. The background is a light green watercolor texture. At the top, the text 'Upload any image with noise and make it clear' is centered. Below this, there is a file selection area with a 'Choose File' button and a text box containing 'No file chosen'. At the bottom, there is a green button labeled 'Proses Gambar'.

3. Klik button “Process Image”, untuk memproses gambar

A screenshot of the same web application interface as before, but now the file selection text box contains a file name: '7b37d40662e619ef7b4bdb3a00c0040e.jpg'. The 'Proses Gambar' button remains at the bottom.

4. Berikut gambar yang sudah berhasil diproses dengan filter median dan filter rata-rata



Original Image with Noise



Average Filtered Image



Median Filtered Image

