DashboardController.php

```
<?php

namespace App\Http\Controllers;

use App\Models\Data_Pos;
use App\Models\Data_Regu;
use App\Models\Kebakaran;
use App\Models\Laporan_Masyarakat;
use App\Models\Penyelamatan;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;

class DashboardController extends
Controller
{
   public function
euclideanDistance($point1, $point2)
   {
      $sum = 0;
      for ($i = 0; $i < count($point1); $i++)
{
         $sum += pow($point1[$i] -
$point2[$i], 2);
      }
      return sqrt($sum);
   }

   function kMeansClustering($data, $k,
$maxIterations = 100)
   {
      // Inisialisasi pusat cluster secara acak
      $centers = array_slice($data, 0, $k);
      // dd($centers);

      $numPoints = count($data);
      $dimensions = count($data[0]);

      // Array untuk menyimpan cluster
setiap titik data
      $clusterAssignments = array_fill(0,
$numPoints, 0);
      // dd($clusterAssignments);

      // Array untuk menyimpan pusat
cluster baru sementara
      $newCenters = array_fill(0, $k,
array_fill(0, $dimensions, 0));
      // dd($newCenters);

      // Algoritma K-means
      for ($iteration = 0; $iteration <
$maxIterations; $iteration++) {
         $converged = true;

         // Hitung jarak Euclidean antara
setiap titik data dan pusat cluster
         for ($i = 0; $i < $numPoints; $i++)
{
            $minDistance = INF;
            $minIndex = 0;

            for ($j = 0; $j < $k; $j++) {
               $distance = $this-
>euclideanDistance($data[$i],
$centers[$j]);
               if ($distance < $minDistance)
{
                  $minDistance = $distance;
                  $minIndex = $j;
               }
            }

            // Periksa apakah cluster berubah
            if ($clusterAssignments[$i] !==
$minIndex) {
               $converged = false;
            }

            $clusterAssignments[$i] =
$minIndex;
         }

         // Jika sudah konvergen, hentikan
iterasi
         if ($converged) {
            break;
         }

         // echo json_encode($newCenters) .
"</br>";
         // Hitung ulang pusat cluster
berdasarkan rata-rata titik data dalam
setiap cluster
         if ($iteration > 0) {
            $counts = array_fill(0, $k, 1);
         } else {
            $counts = array_fill(0, $k, 0);
         }
```

```
            // echo json_encode($counts) .
"</br>";
            foreach ($clusterAssignments as
$index => $cluster) {
                for ($i = 0; $i < $dimensions;
$i++) {
                    // echo "$cluster -> $index ->
$i </br>";
                    $newCenters[$cluster][$i] +=
$data[$index][$i];
                }
                $counts[$cluster]++;
            }
            // echo
json_encode($clusterAssignments) .
"</br>";
            // echo json_encode($data) .
"</br>";
            // echo json_encode($newCenters) .
"</br>";
            // echo json_encode($counts) .
"</br>";

            for ($j = 0; $j < $k; $j++) {
                for ($i = 0; $i < $dimensions;
$i++) {
                    if ($counts[$j] != 0) {
                        $newCenters[$j][$i] /=
$counts[$j];
                    } else {
                        $newCenters[$j][$i] /= 1;
                    }
                }
            }
            // echo json_encode($newCenters) .
"</br></br>";


            $centers = $newCenters;
            // echo json_encode($centers) .
"</br></br>";
        }

        // dd($centers);

        // Mengembalikan hasil cluster
        $clusters = array_fill(0, $k, []);

        foreach ($clusterAssignments as
$index => $cluster) {
            $clusters[$cluster][] =
$data[$index];
        }

        return $clusters;
    }

    public function kmeans()
    {
        $data = [
            [-2.939554408355665,
104.80270628620647],
            [-3.0365007294434148,
104.77045248044105],
            [-3.00547183094745,
104.76529905182822],
            [-2.931161141880136,
104.78358789415701],
            [-2.9646410048462615,
104.7780731782559],
            [-2.9345418493658553,
104.79053026716937],
            [-3.0347493452634997,
104.75782338911986],
            [-3.009984425367705,
104.72304451236147],
            [-2.9504601047568806,
104.79717387881549],
            [-2.9617992779312017,
104.80952045367557],
            [-2.9837995685219245,
104.7702703700254],
            [-2.9680497781686697,
104.8165845094988],
            [-2.9664896524187885,
104.74185305767071],
            [-2.9801275680510977,
104.76569416584677],
            [-2.990119523626274,
104.7560060854446],

        ];

        $k = 3;

        // Panggil fungsi K-means clustering
        $result = $this-
>kMeansClustering($data, $k);

        // Tampilkan hasil cluster
```

```
        foreach ($result as $clusterIndex => $cluster) {
            echo "Cluster " . ($clusterIndex + 1) . ":\n";
            foreach ($cluster as $point) {
                echo "[" . $point[0] . ", " . $point[1] . "]\n";
            }
            echo "\n";
        }

        echo json_encode($data);
    }


    public function index()
    {
        //metode kebakaran
        $minta_data = Kebakaran::select('latitude', 'longitude')->get()->toArray();

        $data = [];
        foreach ($minta_data as $d) {
            $data[] = [$d['latitude'], $d['longitude']];
        }
        // dd($data);

        $markers = [];

        $k = 3;
        if (sizeof($data) <= $k) {
            $markers = $data;
        } else {
            $result = $this->kMeansClustering($data, $k);
            // dd($result);


            // Tampilkan hasil cluster
            foreach ($result as $clusterIndex => $cluster) {
                // echo "Cluster " . ($clusterIndex + 1) . ":\n";
                foreach ($cluster as $point) {

                    $markers[] = ["latitude" => $point[0], "longitude" => $point[1], "name" => "Kluster " . ($clusterIndex + 1)];
                    // echo "[" . $point[0] . ", " . $point[1] . "]\n";
                }
                // echo "\n";
            }
        }

        // dd($markers);


        // Panggil fungsi K-means clustering


        //metode penyelamatan//
        $minta_data = Penyelamatan::select('latitude', 'longitude')->get()->toArray();

        $data = [];

        foreach ($minta_data as $d) {
            $data[] = [$d['latitude'], $d['longitude']];
        }
        // dd($data);
        $markers3 = [];

        $k = 3;
        if (sizeof($data) <= $k) {
            $markers3 = $data;
        } else {

            // Panggil fungsi K-means clustering
            $result = $this->kMeansClustering($data, $k);
            // dd($result);


            // Tampilkan hasil cluster
            foreach ($result as $clusterIndex => $cluster) {
                // echo "Cluster " . ($clusterIndex + 1) . ":\n";
                foreach ($cluster as $point) {

                    $markers3[] = ["latitude" => $point[0], "longitude" => $point[1],
```

```
"name" => "Kluster " . ($clusterIndex + 1)];
                // echo "[" . $point[0] . ", " . $point[1] . "]\n";
            }
            // echo "\n";
        }
    }

    $markers2 = [];
    $data_masyarakat = Laporan_Masyarakat::whereNotNull('verified_oleh')->get();
    // dd($data_masyarakat);
    // Tampilkan hasil cluster
    foreach ($data_masyarakat as $d) {
        $markers2[] = ["latitude" => $d->latitude, "longitude" => $d->longitude, "name" => $d->id];
        // echo "[" . $point[0] . ", " . $point[1] . "]\n";

    }



    // dd($markers2);


    //data untuk barchart
    //kebakaran

    // $labels = [];
    $data_bar = [];
    $bulan = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12];



    $countsPerMonthKebakaran = Kebakaran::select(

DB::raw('DATE_FORMAT(tanggal, "%m") as month'),
        DB::raw('COUNT(*) as count')
    )
        ->groupBy('month')
        ->get();
    $data_kebakaran = [];
    for ($i = 0; $i < 12; $i++) {
        $data_kebakaran[] = 0;
    }
    foreach ($countsPerMonthKebakaran as $c) {
        // $labels[] = $c->month;
        $data_kebakaran[intval($c->month) - 1] = $c->count;
    }



    $countsPerMonthPenyelamatan = Penyelamatan::select(

DB::raw('DATE_FORMAT(tanggal, "%m") as month'),
        DB::raw('COUNT(*) as count')
    )
        ->groupBy('month')
        ->get();
    $data_penyelamatan = [];
    for ($i = 0; $i < 12; $i++) {
        $data_penyelamatan[] = 0;
    }
    foreach ($countsPerMonthPenyelamatan as $c) {
        // $labels[] = $c->month;
        $data_penyelamatan[intval($c->month) - 1] = $c->count;
    }

    $data_bar = [
        [
            "name" => "kebakaran",
            "data" => $data_kebakaran,
            "type" => "bar"

        ],
        [
            "name" => "penyelamatan",
            "data" => $data_penyelamatan,
            "type" => "bar"

        ]

    ];

    $kebakaran = Kebakaran::count();
    $penyelamatan = Penyelamatan::count();
    $pos = Data_Pos::count();
```

```
        $regu = Data_Regu::count();
        return view('dashboards.index',
compact('data_bar', 'markers', 'markers2',
'markers3', 'kebakaran', 'penyelamatan',
'pos', 'regu'));
    }
}
```

dashboards
index.blade.php

```
@extends('layouts.app')
@section('pageTitle', 'Dashboard')
@section('content')
    <main id="main" class="main">
        <div class="card">
            <div class="card-body">
                <div class="grey-bg container-
fluid">
                    <section id="minimal-
statistics">
                        <div class="row">
                            <div class="col-12 mt-3
mb-1">
                                <h4 class="text-
uppercase">DASHBOARD</h4>
                                <div class="col-md-12
text-center">
                                    <p>Selamat datang
di Dashboard Pemadam Kebakaran dan
Penyelamatan
                                        {{ Auth::user()-
>nama }}!</p>
                                </div>
                            </div>
                        </section>
                    </div>

                    <div class="row">
                        <div class="col-lg-3 col-sm-
6">
                            <div class="card-box bg-
red">
                                <div class="inner">
                                    <h3>{{ $kebakaran
}}</h3>
                                    <p>Kebakaran</p>
                                </div>
                                <div class="icon">
                                    <i class="fa-solid fa-
house-fire" aria-hidden="true"></i>
                                </div>
                                <a href="{{
route('kebakarans.index') }}" class="card-
box-footer">View More <i
                                        class="fa fa-arrow-
circle-right"></i></a>
                            </div>
                        </div>

                        <div class="col-lg-3 col-sm-
6">
                            <div class="card-box bg-
blue2">
                                <div class="inner">
                                    <h3>{{
$penyelamatan }}</h3>
                                    <p>Penyelamatan</p>
                                </div>
                                <div class="icon">
                                    <i class="fa-solid fa-
people-roof" aria-hidden="true"></i>
                                </div>
                                <a href="{{
route('penyelamatans.index') }}"
class="card-box-footer">View More <i
                                        class="fa fa-arrow-
circle-right"></i></a>
                            </div>
                        </div>
                        <div class="col-lg-3 col-sm-
6">
                            <div class="card-box bg-
red">
                                <div class="inner">
                                    <h3>{{ $pos }}</h3>
                                    <p>Pos</p>
                                </div>
                                <div class="icon">
                                    <i class="fa-solid fa-
location-dot" aria-hidden="true"></i>
                                </div>
                                <a href="{{
route('data_poss.index') }}" class="card-
box-footer">View More <i
                                        class="fa fa-arrow-
circle-right"></i></a>
                            </div>
                        </div>
```

```
            <div class="col-lg-3 col-sm-6">
                <div class="card-box bg-blue2">
                    <div class="inner">
                        <h3>{{ $regu }}</h3>
                        <p>Regu</p>
                    </div>
                    <div class="icon">
                        <i class="fa-solid fa-people-group"></i>
                    </div>
                    <a href="{{ route('data_regus.index') }}" class="card-box-footer">View More <i
                            class="fa fa-arrow-circle-right"></i></a>
                </div>
            </div>
        </div>
        <div class="row">
            <div class="col-12 mt-3 mb-1">
                {{-- <h4 class="text-uppercase">DASHBOARD</h4> --}}
                <div class="col-md-12 text-center">
                    <p>Dashboard ini kami dirancang dengan fokus pada kemudahan penggunaan dan fleksibilitas<br>
                        Bersama-sama,
                        kita akan terus menjaga kota Palembang tetap aman dan dilindungi dari bahaya kebakaran dan
                        bencana!
                        <br>
                        Pada Halaman Ini Kita Dapat Melihat Data Kebakaran dan Penyelamatan Yang Terjadi Di Kota
                        Palembang
                    </p>
                </div>
            </div>
            <div class="row">
                <div class="col-lg-6">
                    <div class="profile-img mt-5">
                        <h5 class="card-title">Bar Chart</h5>
                    <div class="card border-0 shadow p-3 mb-5 bg-white rounded">
                        <div class="card-content ml-2">

                            <!-- Bar Chart -->
                            <div id="barChart" style="min-height: 400px; min-width: 80%" class="echart"></div>
                            {{-- <div class="chart-container" style="position: relative; height:100vh; width:100vw"> --}}
                            {{-- <canvas id="barChart"></canvas> --}}
                            {{-- </div> --}}
                            <script>
                                var data_bar = <?php echo json_encode($data_bar); ?>;

console.log(data_bar);

document.addEventListener("DOMContentLoaded", () => {

echarts.init(document.querySelector("#barChart")).setOption({
                                    tooltip: {
                                        trigger: 'axis',

axisPointer: {
                                            type: 'shadow'
                                        }
                                    },
                                    legend: {},
                                    grid: {
                                        left: '3%',
                                        right: '4%',
                                        bottom: '3%',

containLabel: true
                                    },
                                    xAxis: {
```

```
                                    type:
'value',

boundaryGap: [0, 0.01]
                                },
                                xAxis: {
                                    type:
'category',
                                    data:
['Januari', 'Februari', 'Maret', 'April', 'Mei',
'Juni', 'Juli', 'Agustus',

'September', 'Oktober', 'November',
'Desember'
                                    ]

                                },
                                yAxis: {
                                    type:
'value'
                                },
                                series:
data_bar,

                            });
                        });
                    </script>
                    <!-- End Bar
Chart -->
                </div>
            </div>
        </div>
        <div class="col-lg-6">
            <div class="profile-img
mt-5">
                <h5 class="card-
title">Fire Fighter</h5>
                <div class="card
border-0 shadow p-3 mb-5 bg-white
rounded">
                    <div class="card-
content ml-2">
                        <p class="text-
justify">Platform Terpadu Untuk
Mengawasi, Mengelola, Dan

Mengkoordinasikan Operasi
                            Pemadam
Kebakaran Dan Penyelamatan Guna
Menjaga Keamanan Dan Keselamatan
Warga
                            Serta
Infrastruktur Kota Palembang. Dengan
Tampilan Yang Intuitif Dan Informasi
                            Real-Time,
Kami Siap Memberikan Pandangan
Menyeluruh Atas Kondisi Terkini,
                            Sumber Daya
Yang Tersedia, Serta Aktivitas Tim
Pemadam Kebakaran Dan
                            Penyelamatan
Untuk Meningkatkan Respons Dan
Efisiensi Dalam Mengatasi Situasi
                            Darurat.
Bersama-Sama, Mari Menjadikan
Palembang Lebih Aman Dan Tangguh
Dalam
                            Menghadapi
Tantangan Kebakaran Dan Bencana.</p>
                        </div>
                    </div>
                </div>
            </div>
            <div class="row justify-
content-between">
                <div class="col-xs-12 col-
sm-6 col-md-6">
                    <div class="profile-img
mt-5">
                        <h3>Lokasi Sering
Terjadi Penyelamatan</h3>
                        <div id="map3"
class="map3"></div>
                    </div>
                </div>
                <div class="col-xs-12 col-
sm-6 col-md-6">
                    <div class="profile-img
mt-5">
                        <h3>Lokasi Sering
Terjadi Kebakaran</h3>
                        <div id="map"
class="map"></div>
                    </div>
                </div>
            </div>
        </div>
```

```
    </div>
  </main>
  <script>
    function initMap(data) {
      var markers = data;
      console.log("map");
      console.log(markers);

      if (markers.length == 0) {
        var map = new ol.Map({
          target: 'map',
          layers: [
            new ol.layer.Tile({
              source: new
ol.source.OSM()
            })
          ],
          view: new ol.View({
            center:
ol.proj.fromLonLat([0, 0]),
            zoom: 4
          })
        });

      } else {
        var map = new ol.Map({
          target: 'map',
          layers: [
            new ol.layer.Tile({
              source: new
ol.source.OSM()
            })
          ],
          view: new ol.View({
            center:
ol.proj.fromLonLat([markers[0].longitude,
markers[0].latitude]),
            zoom: 4
          })
        });

      }

      console.log(map);

      // Add markers to the map
      markers.forEach(function(marker)
{
        var markerFeature = new
ol.Feature({
          geometry: new
ol.geom.Point(ol.proj.fromLonLat([marker
.longitude, marker.latitude])),
          name: marker.name
        });
        var markerStyle = new
ol.style.Style({
          image: new ol.style.Icon({
            anchor: [0.5, 32],
            anchorXUnits: 'fraction',
            anchorYUnits: 'pixels',
            src: '{{
asset('landing_page/icon/icongreen.png')
}}'
          })
        });
        if (marker.name == "Kluster 1")
{

        } else if (marker.name ==
"Kluster 2") {
          markerStyle = new
ol.style.Style({
            image: new ol.style.Icon({
              anchor: [0.5, 32],
              anchorXUnits: 'fraction',
              anchorYUnits: 'pixels',
              src: '{{
asset('landing_page/icon/iconred.png') }}'
            })
          });

        } else if (marker.name ==
"Kluster 3") {
          markerStyle = new
ol.style.Style({
            image: new ol.style.Icon({
              anchor: [0.5, 32],
              anchorXUnits: 'fraction',
              anchorYUnits: 'pixels',
              src: '{{
asset('landing_page/icon/iconblue.png') }}'
            })
          });

        }


markerFeature.setStyle(markerStyle);
```

```
            var vectorSource = new
ol.source.Vector({
                features: [markerFeature]
            });

            var vectorLayer = new
ol.layer.Vector({
                source: vectorSource
            });

            map.addLayer(vectorLayer);



            var radius = 200; // in meters

            var circle = new ol.Feature({
                geometry: new
ol.geom.Circle(ol.proj.fromLonLat([marke
r.longitude, marker.latitude]),
                    radius),
            });

            var circleLayer = new
ol.layer.Vector({
                source: new
ol.source.Vector({
                    features: [circle],
                }),
                style: new ol.style.Style({
                    stroke: new
ol.style.Stroke({
                        color: 'blue',
                        width: 2,
                    }),
                    fill: new ol.style.Fill({
                        color: 'rgba(0, 0, 255,
0.1)', // Transparent fill color
                    }),
                }),
            });

            map.addLayer(circleLayer);
        });


    }



    function initMap3(data) {
        var markers = data;
        console.log("map3");
            console.log(markers);

            if (markers.length == 0) {
                var map3 = new ol.Map({
                    target: 'map3',
                    layers: [
                        new ol.layer.Tile({
                            source: new
ol.source.OSM()
                        })
                    ],
                    view: new ol.View({
                        center:
ol.proj.fromLonLat([0, 0]),
                        zoom: 4
                    })
                });

            } else {
                var map3 = new ol.Map({
                    target: 'map3',
                    layers: [
                        new ol.layer.Tile({
                            source: new
ol.source.OSM()
                        })
                    ],
                    view: new ol.View({
                        center:
ol.proj.fromLonLat([markers[0].longitude,
markers[0].latitude]),
                        zoom: 4
                    })
                });

            }



            // Add markers to the map
            markers.forEach(function(marker)
{
                var markerFeature = new
ol.Feature({
                    geometry: new
ol.geom.Point(ol.proj.fromLonLat([marker
.longitude, marker.latitude])),
                    name: marker.name
                });
                var markerStyle = new
ol.style.Style({
```

```
                image: new ol.style.Icon({
                    anchor: [0.5, 32],
                    anchorXUnits: 'fraction',
                    anchorYUnits: 'pixels',
                    src: '{{
asset('landing_page/icon/icongreen.png')
}}'
                })
            });
            if (marker.name == "Kluster 1")
{

            } else if (marker.name ==
"Kluster 2") {
                markerStyle = new
ol.style.Style({
                    image: new ol.style.Icon({
                        anchor: [0.5, 32],
                        anchorXUnits: 'fraction',
                        anchorYUnits: 'pixels',
                        src: '{{
asset('landing_page/icon/iconred.png') }}'
                    })
                });

            } else if (marker.name ==
"Kluster 3") {
                markerStyle = new
ol.style.Style({
                    image: new ol.style.Icon({
                        anchor: [0.5, 32],
                        anchorXUnits: 'fraction',
                        anchorYUnits: 'pixels',
                        src: '{{
asset('landing_page/icon/iconblue.png') }}'
                    })
                });

            }


markerFeature.setStyle(markerStyle);

            var vectorSource = new
ol.source.Vector({
                features: [markerFeature]
            });

            var vectorLayer = new
ol.layer.Vector({
                source: vectorSource
            });

            map3.addLayer(vectorLayer);


            var radius = 200; // in meters

            var circle = new ol.Feature({
                geometry: new
ol.geom.Circle(ol.proj.fromLonLat([marke
r.longitude, marker.latitude]),
                    radius),
            });

            var circleLayer = new
ol.layer.Vector({
                source: new
ol.source.Vector({
                    features: [circle],
                }),
                style: new ol.style.Style({
                    stroke: new
ol.style.Stroke({
                        color: 'blue',
                        width: 2,
                    }),
                    fill: new ol.style.Fill({
                        color: 'rgba(0, 0, 255,
0.1)', // Transparent fill color
                    }),
                }),
            });

            map3.addLayer(circleLayer);
        });


    }


    // Load JSON data and initialize the
map
    json = <?php echo
json_encode($markers); ?>;
    json3 = <?php echo
json_encode($markers3); ?>;
    console.log(json);
    console.log(json3);
```

AuthController.php

```
<?php

namespace App\Http\Controllers\Auth;

use App\Http\Controllers\Controller;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;
use Illuminate\Support\Facades\Session;
use App\Models\User;
use Illuminate\Support\Facades\Hash;

class AuthController extends Controller
{
    /**
     * Write code on Method
     *
     * @return response()
     */
    public function index()
    {
        return view('auth.login');
    }

    /**
     * Write code on Method
     *
     * @return response()
     */
    public function registration()
    {
        return view('auth.registration');
    }

    /**
     * Write code on Method
     *
     * @return response()
     */
    public function postLogin(Request
$request)
    {
        $request->validate([
            'email' => 'required',
            'password' => 'required',
        ]);

        $credentials = $request->only(
            'email',
            'password'
        );
        if (Auth::attempt($credentials)) {
            return redirect()-
>intended('dashboard')
                ->withSuccess('You have
Successfully loggedin');
        }

        return redirect("login")-
>withSuccess('Oppes!
You have entered invalid credentials');
    }

    /**
     * Write code on Method
     *
     * @return response()
     */
    public function
postRegistration(Request $request)
    {
        // dd($request->all());
        $request->validate([
            'nip' => 'required',
            'nama' => 'required',
            'jenis_kelamin' => 'required',
            'email' =>
'required|email|unique:users',
            'password' => 'required|min:6',
            'foto' =>
'image|mimes:jpeg,png,jpg,gif,svg|max:20
48',
        ]);

        $input = $request->all();

        if ($image = $request->file('foto')) {
            $destinationPath = 'image/';
            $profileImage = date('YmdHis') .
"." . $image-
>getClientOriginalExtension();
            $image->move($destinationPath,
$profileImage);
            $input['foto'] = "$profileImage";
        }



        $check = $this->create($input);
```

```
        $check->assignRole("Masyarakat");

        return redirect("login")-
>withSuccess('Great! You have
Successfully loggedin');
    }

    /**
     * Write code on Method
     *
     * @return response()
     */
    public function dashboard()
    {
        if (Auth::check()) {
            return view('auth.dashboard');
        }

        return redirect("login")-
>withSuccess('Opps!You do not have
access');
    }

    /**
     * Write code on Method
     *
     * @return response()
     */
    public function create(array $data)
    {
        return User::create([
            'nip' => $data['nip'],
            'nama' => $data['nama'],
            'jenis_kelamin' =>
$data['jenis_kelamin'],
            'email' => $data['email'],
            'password' =>
Hash::make($data['password']),
            'foto' => $data['foto']
        ]);
    }

    /**
     * Write code on Method
     *
     * @return response()
     */
    public function logout()
    {
        Session::flush();
        Auth::logout();

        return Redirect('login');
    }
}
```

LandingPageController.php

```
<?php

namespace App\Http\Controllers;

use App\Models\Data_Pos;
use App\Models\Kebakaran;
use App\Models\Laporan_Masyarakat;
use App\Models\Penyelamatan;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;
use Mail;

class LandingPageController extends
Controller
{
    public function
euclideanDistance($point1, $point2)
    {
        $sum = 0;
        for ($i = 0; $i < count($point1); $i++)
{
            $sum += pow($point1[$i] -
$point2[$i], 2);
        }
        return sqrt($sum);
    }

    function kMeansClustering($data, $k,
$maxIterations = 100)
    {
        // Inisialisasi pusat cluster secara acak
        $centers = array_slice($data, 0, $k);

        $numPoints = count($data);
        $dimensions = count($data[0]);

        // Array untuk menyimpan cluster
setiap titik data
        $clusterAssignments = array_fill(0,
$numPoints, 0);
```

```
        // Array untuk menyimpan pusat cluster baru sementara
        $newCenters = array_fill(0, $k, array_fill(0, $dimensions, 0));

        // Algoritma K-means
        for ($iteration = 0; $iteration < $maxIterations; $iteration++) {
            $converged = true;

            // Hitung jarak Euclidean antara setiap titik data dan pusat cluster
            for ($i = 0; $i < $numPoints; $i++) {
                $minDistance = INF;
                $minIndex = 0;

                for ($j = 0; $j < $k; $j++) {
                    $distance = $this->euclideanDistance($data[$i], $centers[$j]);
                    if ($distance < $minDistance) {
                        $minDistance = $distance;
                        $minIndex = $j;
                    }
                }

                // Periksa apakah cluster berubah
                if ($clusterAssignments[$i] !== $minIndex) {
                    $converged = false;
                }

                $clusterAssignments[$i] = $minIndex;
            }

            // Jika sudah konvergen, hentikan iterasi
            if ($converged) {
                break;
            }

            // Hitung ulang pusat cluster berdasarkan rata-rata titik data dalam setiap cluster
            $counts = array_fill(0, $k, 0);
            foreach ($clusterAssignments as $index => $cluster) {
                for ($i = 0; $i < $dimensions; $i++) {
                    $newCenters[$cluster][$i] += $data[$index][$i];
                }
                $counts[$cluster]++;
            }

            for ($j = 0; $j < $k; $j++) {
                for ($i = 0; $i < $dimensions; $i++) {
                    $newCenters[$j][$i] /= $counts[$j];
                }
            }

            $centers = $newCenters;
        }

        // Mengembalikan hasil cluster
        $clusters = array_fill(0, $k, []);

        foreach ($clusterAssignments as $index => $cluster) {
            $clusters[$cluster][] = $data[$index];
        }

        return $clusters;
    }

    public function kmeans()
    {
        $data = [
            [-2.939554408355665, 104.80270628620647],
            [-3.0365007294434148, 104.77045248044105],
            [-3.00547183094745, 104.76529905182822],
            [-2.931161141880136, 104.78358789415701],
            [-2.9646410048462615, 104.7780731782559],
            [-2.9345418493658553, 104.79053026716937],
            [-3.0347493452634997, 104.75782338911986],
            [-3.009984425367705, 104.72304451236147],
```

```
        [-2.9504601047568806,
104.79717387881549],
        [-2.9617992779312017,
104.80952045367557],
        [-2.9837995685219245,
104.7702703700254],
        [-2.9680497781686697,
104.8165845094988],
        [-2.9664896524187885,
104.74185305767071],
        [-2.9801275680510977,
104.76569416584677],
        [-2.990119523626274,
104.7560060854446],

    ];

    $k = 3;

    // Panggil fungsi K-means clustering
    $result = $this-
>kMeansClustering($data, $k);

    // Tampilkan hasil cluster
    foreach ($result as $clusterIndex =>
$cluster) {
        echo "Cluster " . ($clusterIndex +
1) . ":\n";
        foreach ($cluster as $point) {
            echo "[" . $point[0] . ", " .
$point[1] . "]\n";
        }
        echo "\n";
    }

    echo json_encode($data);
  }

  public function index()
  {
    //   //metode kebakaran
    // $minta_data =
Kebakaran::select('latitude', 'longitude')-
>get()->toArray();

    // $data = [];
    // foreach ($minta_data as $d) {
    //     $data[] = [$d['latitude'],
$d['longitude']];
    // }
    // // dd($data);

    // $markers = [];

    // $k = 3;
    // if (sizeof($data) <= $k) {
    //     $markers = $data;
    // } else {
    //     $result = $this-
>kMeansClustering($data, $k);
    //     // dd($result);



    //     // Tampilkan hasil cluster
    //     foreach ($result as $clusterIndex
=> $cluster) {
    //         // echo "Cluster " .
($clusterIndex + 1) . ":\n";
    //         foreach ($cluster as $point) {

    //             $markers[] = ["latitude" =>
$point[0], "longitude" => $point[1],
"name" => "Kluster " . ($clusterIndex +
1)];
    //             // echo "[" . $point[0] . ", " .
$point[1] . "]\n";
    //         }
    //         // echo "\n";
    //     }
    // }

    // // dd($markers);



    // // Panggil fungsi K-means
clustering

    // //metode penyelamatan//
    // $minta_data =
Penyelamatan::select('latitude',
'longitude')->get()->toArray();

    // $data = [];
    // $markers3 = [];
    // foreach ($minta_data as $d) {
    //     $data[] = [$d['latitude'],
$d['longitude']];
    // }
    // // dd($data);
```

```
        // if (sizeof($data) < 4) {
        // } else {
        //     $k = 3;

        //     // Panggil fungsi K-means clustering
        //     $result = $this->kMeansClustering($data, $k);
        //     // dd($result);

        //     // Tampilkan hasil cluster
        //     foreach ($result as $clusterIndex => $cluster) {
        //         // echo "Cluster " . ($clusterIndex + 1) . ":\n";
        //         foreach ($cluster as $point) {

        //             $markers3[] = ["latitude" => $point[0], "longitude" => $point[1], "name" => "Kluster " . ($clusterIndex + 1)];
        //             // echo "[" . $point[0] . ", " . $point[1] . "]\n";
        //         }
        //         // echo "\n";
        //     }
        // }


        $markers2 = [];
        $data_masyarakat = Laporan_Masyarakat::whereNotNull('verified_oleh')->get();
        // dd($data_masyarakat);
        // Tampilkan hasil cluster
        foreach ($data_masyarakat as $d) {
            $markers2[] = ["latitude" => $d->latitude, "longitude" => $d->longitude, "name" => $d->id];
            // echo "[" . $point[0] . ", " . $point[1] . "]\n";
        }



        // dd($markers2);



        //data untuk barchart
        //kebakaran
        // $labels = [];
        $data_bar = [];
        $bulan = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12];

        $countsPerMonthKebakaran = Kebakaran::select(

DB::raw('DATE_FORMAT(tanggal, "%m") as month'),
            DB::raw('COUNT(*) as count')
        )
            ->groupBy('month')
            ->get();
        $data_kebakaran = [];
        for ($i = 0; $i < 12; $i++) {
            $data_kebakaran[] = 0;
        }
        foreach ($countsPerMonthKebakaran as $c) {
            // $labels[] = $c->month;
            $data_kebakaran[intval($c->month) - 1] = $c->count;
        }

        $countsPerMonthPenyelamatan = Penyelamatan::select(

DB::raw('DATE_FORMAT(tanggal, "%m") as month'),
            DB::raw('COUNT(*) as count')
        )
            ->groupBy('month')
            ->get();
        $data_penyelamatan = [];
        for ($i = 0; $i < 12; $i++) {
            $data_penyelamatan[] = 0;
        }
        foreach ($countsPerMonthPenyelamatan as $c) {
            // $labels[] = $c->month;
            $data_penyelamatan[intval($c->month) - 1] = $c->count;
        }

        $data_bar = [
            [
                "name" => "kebakaran",
                "data" => $data_kebakaran,
```

```
            "type" => "bar"

        ],
        [
            "name" => "penyelamatan",
            "data" => $data_penyelamatan,
            "type" => "bar"

        ]

    ];

    // dd($data_bar);
    $laporan_masyarakat =
Laporan_Masyarakat::all();
    $data_pos = Data_Pos::all();
    return view('landing_pages.index',
compact('markers2', 'data_bar', 'data_pos',
'laporan_masyarakat'));
  }

  public function create()
  {
    return view('landing_pages.create');
  }

  public function store(Request $request)
  {
    $request->validate([
        'nama' => 'required',
        'alamat' => 'required',
        'no_hp' => 'required',
        'email' => 'required',
        'jenis_kejadian' => 'required',
        'foto' =>
'required|image|mimes:jpeg,png,jpg,gif,svg
|max:2048',
        'foto_ktp' =>
'required|image|mimes:jpeg,png,jpg,gif,svg
|max:2048',
        'latitude' => 'required',
        'longitude' => 'required',
    ]);
    $input = $request->all();


    if ($image = $request->file('foto')) {
        $destinationPath = 'image/';
        $profileImage = date('YmdHis') .
"." . $image-
>getClientOriginalExtension();
        $image->move($destinationPath,
$profileImage);
        $input['foto'] = "$profileImage";
    }
    if ($image = $request-
>file('foto_ktp')) {
        $destinationPath = 'image/';
        $profileImage = "ktp_" .
date('YmdHis') . "." . $image-
>getClientOriginalExtension();
        $image->move($destinationPath,
$profileImage);
        $input['foto_ktp'] =
"$profileImage";
    }
    Laporan_Masyarakat::create($input);

    $details = [
        'title' => 'Mail from dinas pemadam
kebakaran',
        'body' => 'ada laporan masuk.'
    ];


Mail::to('notifikasikebakaran@gmail.com')
->send(new
\App\Mail\MyTestMail($details));



    return redirect()-
>route('landing_pages.index')
        ->with('success', 'Laporan
Masyarakat created successfully.');
  }
}
```

landing_pages
index.blade.php

```
@extends('landing_pages.app')
@section('content')

  <body class="body-color">
    <section id="text-header">
      <div class="container">
        <div class="row">
```

```
                <div class="col-md-12 text-
center">
                    <h2>Selamat Datang di
Halaman Profil Pemadam Kebakaran dan
Penyelamatan Kota Palembang!</h1>
                </div>
            </div>
        </div>
    </section>
    {{-- carousel --}}
    <div class="container">
        {{-- <div class="row">
        <div class="col-md-5 offset-md-
3"> --}}
        <div class="text-right">
            <div id="carouselExampleDark"
class="carousel carousel-dark slide" data-
bs-ride="carousel">
                <div class="carousel-
indicators">
                    <button type="button" data-
bs-target="#carouselExampleDark" data-
bs-slide-to="0" class="active"
                        aria-current="true" aria-
label="Slide 1"></button>
                    <button type="button" data-
bs-target="#carouselExampleDark" data-
bs-slide-to="1"
                        aria-label="Slide
2"></button>
                    <button type="button" data-
bs-target="#carouselExampleDark" data-
bs-slide-to="2"
                        aria-label="Slide
3"></button>
                </div>
                <div class="carousel-inner">
                    <div class="carousel-item
custom-carousel-item active" data-bs-
interval="10000">
                        <img src="{{
asset('landing_page/image/carousel1.jpeg')
}}" class="d-block w-100"
                            alt="...">
                        <div class="carousel-
caption text-light d-none d-md-block">
                            {{-- <h5>Kami Siap
Melayani Anda</h5> --}}
                            {{-- <p>Kami siap
melayani anda</p> --}}
                            {{-- <a href="{{
route('landing_pages.create') }}"
class="btn btn-danger">Lapor kejadian <i
                                class="bi bi-arrow-right-
circle"></i></a> --}}
                        </div>
                    </div>
                    <div class="carousel-item
custom-carousel-item " data-bs-
interval="2000">
                        <img src="{{
asset('landing_page/image/carousel2.jpeg')
}}" class="d-block w-100"
                            alt="...">
                        <div class="carousel-
caption text-light d-none d-md-block">
                            {{-- <h5>Second slide
label</h5>
                <p>Some representative
placeholder content for the second
slide.</p> --}}
                        </div>
                    </div>
                    <div class="carousel-item
custom-carousel-item ">
                        <img src="{{
asset('landing_page/image/carousel3.jpeg')
}}" class="d-block w-100"
                            alt="...">
                        <div class="carousel-
caption text-light d-none d-md-block">
                            {{-- <h5>Third slide
label</h5>
                <p>Some representative
placeholder content for the third slide.</p>
--}}
                        </div>
                    </div>
                </div>
                <button class="carousel-
control-prev" type="button" data-bs-
target="#carouselExampleDark"
                    data-bs-slide="prev">
                    <span class="carousel-
control-prev-icon text-light" aria-
hidden="true"></span>
                    <span class="visually-
hidden">Previous</span>
                </button>
```

```
                <button class="carousel-control-next" type="button" data-bs-target="#carouselExampleDark"
                    data-bs-slide="next">
                    <span class="carousel-control-next-icon" aria-hidden="true"></span>
                    <span class="visually-hidden">Next</span>
                </button>
            </div>
        </div>
    </div>
    </div>
    </div>
    {{-- end carousel --}}

    <div class="container mt-3">
        <div class="row">
            <div class="col-md-12 text-center">
                <h5>Dinas Pemadam Kebakaran dan Penyelamatan adalah unsur pelaksana pemerintah daerah yang diberi
                    tanggung jawab dalam melaksanakan tugas-tugas penanganan masalah kebakaran,
                    mulai dari pencegahan kebakaran, pemadam kebakaran, sampai dengan penyelamatan jiwa dari ancaman
                    kebakaran dan bencana alam lain.
                </h6>
            </div>
        </div>
    </div>

    <!-- Profile -->
    <section id="Profile">
        <div class="container">
            <div class="row justify-content-between">
                <div class="col-lg-5">
                    <div class="profile-img mt-5">
                        <h2>Lokasi Sedang Terjadi Kebakaran / Penyelamatan</h2>

                        <div id="map2" class="map2"></div>
                    <?php $i = 1; ?>
                    <table class="table table-bordered">
                        <thead>
                            <tr>
                                <th scope="col">No</th>
                                <th scope="col">Alamat</th>
                                <th scope="col">Kejadian</th>
                                <th scope="col">Keterangan</th>
                                <th scope="col">Tanggal</th>
                            </tr>
                        </thead>
                        <tbody>
                            @foreach ($laporan_masyarakat as $laporan_masyarakats)
                                @if ($laporan_masyarakats->verified_tanggal != null)
                                    <tr>
                                        <th scope="row">{{ $i++ }}</th>
                                        <td>{{ $laporan_masyarakats->alamat }}</td>
                                        <td>{{ $laporan_masyarakats->jenis_kejadian }}</td>
                                        <td>{{ $laporan_masyarakats->keterangan_kejadian }}</td>
                                        <td>{{ $laporan_masyarakats->verified_tanggal }}</td>
                                    </tr>
                                @endif
                            @endforeach
                        </tbody>
                    </table>
                </div>
            </div>
            <div class="col-lg-6">
                <div class="profile-img mt-6">
```

```
                    <div class="profile-img
mt-5">
                        <h2>Statistik
Keajadian</h2>
                        <div class="card
border-0 shadow p-3 mb-5 bg-white
rounded">
                            <div class="card-
content ml-2">

                                <!-- Bar Chart -->
                                <div
id="barChart" style="min-height: 400px;
min-width: %" class="echart"></div>
                                {{-- <div
class="chart-container" style="position:
relative; height:100vh; width:100vw"> --
}}
                                {{-- <canvas
id="barChart"></canvas> --}}
                                {{-- </div> --}}
                                <script>
                                    var data_bar =
<?php echo json_encode($data_bar); ?>;

console.log(data_bar);

document.addEventListener("DOMConten
tLoaded", () => {

echarts.init(document.querySelector("#bar
Chart")).setOption({
                                        tooltip: {
                                            trigger:
'axis',

axisPointer: {
                                                type:
'shadow'
                                            }
                                        },
                                        legend:
{},
                                        grid: {
                                            left:
'3%',
                                            right:
'4%',
                                            bottom:
'3%',
containLabel: true
                                        },
                                        xAxis: {
                                            type:
'value',

boundaryGap: [0, 0.01]
                                        },
                                        xAxis: {
                                            type:
'category',
                                            data:
['Januari', 'Februari', 'Maret', 'April', 'Mei',
'Juni', 'Juli', 'Agustus',

'September', 'Oktober', 'November',
'Desember'
                                            ]

                                        },
                                        yAxis: {
                                            type:
'value'
                                        },
                                        series:
data_bar,

                                    });
                                });
                            </script>

                            <!-- End Bar
Chart -->
                        </div>
                    </div>
                </div>
            </div>
        </div>
    </section>

    <div class="container mt-3">
        <div class="row">
            <div class="col-md-12 text-
center">
                <h5> Keamanan dan
keselamatan warga adalah prioritas utama
```

```
kami. Kami berkomitmen untuk memberikan
                pelayanan berkualitas tinggi, melalui upaya pencegahan, penanggulangan, dan penanganan darurat yang
                terkoordinasi dengan baik.
                <br><br>
                Jelajahi situs ini untuk mengetahui lebih lanjut tentang tim kami, keahlian kami, dan layanan yang
                kami sediakan. Jangan ragu untuk menghubungi kami jika Anda membutuhkan bantuan atau memiliki
                pertanyaan seputar keamanan.
                <br> <br>
                Terima kasih atas kepercayaan Anda pada Pemadam Kebakaran dan Penyelamatan Kota Palembang.
                Bersama-sama, mari kita jaga kota ini tetap aman dan damai.
              </h5>
            </div>
          </div>
        </div>
        <hr>
        <section id="kontak">
          <div class="container mt-1 table-container">
            <!-- Two columns layout -->
            <div class="row">
              <div class="col-lg-12 col-md-12">
                <div class="card border-0">
                  <div class="card-body text-center">
                    <!-- Table with Bootstrap classes -->
                    <h4 class="text-center">POS PEMADAM KEBAKARAN DAN PENYELAMATAN KOTA PALEMBANG</h4>
                    <table class="table">
                      <thead>
                        <tr>
                          <th width="150px">Nama Pos</th>
                          <th width="250px">Alamat</th>
                          <th width="80px">No Telepon</th>
                        </tr>
                      </thead>
                      <tbody>
                        <!-- Isi tabel disini -->
                        @foreach ($data_pos as $data_poss)
                          <tr>
                            <td>{{ $data_poss->pos }}</td>
                            <td>{{ $data_poss->alamat }}</td>
                            <td>{{ $data_poss->no_telepon }}</td>
                          </tr>
                        @endforeach
                        <!-- tambahkan baris lainnya sesuai kebutuhan -->
                      </tbody>
                    </table>
                  </div>
                </div>
              </div>
            </div>
          </div>
        </section>

    </body>
    <!-- End Profile -->

    <!-- Footer -->
    <footer id="Footer" class="footer">
      <div class="container">
        <div class="row mt-5">
          <div class="col-xl-6 col-lg-6 col-md-6">
            <div>
              <h6 class="mt-2 ">Dinas Pemadam Kebakaran Dan Penyelamatan Kota Palembang</h6>
              <p class="mt-2">Jl. Merdeka No.8, 19 Ilir, Kec. Bukit Kecil, Kota Palembang, Sumatera Selatan
                30113</p>
            </div>
          </div>
```

```
        </div>
      </div>
    </footer>

    <!-- End Footer -->
    <script
src="https://openlayers.org/en/v6.5.0/build
/ol.js"></script>
    <script>
      // Callback function to create map and
add markers


      function initMap2(data) {
        var markers = data;
        console.log(markers);

        if (markers.length == 0) {
          var map2 = new ol.Map({
            target: 'map2',
            layers: [
              new ol.layer.Tile({
                source: new
ol.source.OSM()
              })
            ],
            view: new ol.View({
              center:
ol.proj.fromLonLat([0, 0]),
              zoom: 4
            })
          });

        } else {
          var map2 = new ol.Map({
            target: 'map2',
            layers: [
              new ol.layer.Tile({
                source: new
ol.source.OSM()
              })
            ],
            view: new ol.View({
              center:
ol.proj.fromLonLat([markers[0].longitude,
markers[0].latitude]),
              zoom: 4
            })
          });
        }

        // Add markers to the map
        markers.forEach(function(marker)
{
          var markerFeature = new
ol.Feature({
            geometry: new
ol.geom.Point(ol.proj.fromLonLat([marker
.longitude, marker.latitude])),
            name: marker.name
          });
          var markerStyle = new
ol.style.Style({
            image: new ol.style.Icon({
              anchor: [0.5, 32],
              anchorXUnits: 'fraction',
              anchorYUnits: 'pixels',
              src: '{{
asset('landing_page/icon/iconred.png') }}'
            })
          });
          if (marker.name == "Kluster 1")
{

          } else if (marker.name ==
"Kluster 2") {
            markerStyle = new
ol.style.Style({
              image: new ol.style.Icon({
                anchor: [0.5, 46],
                anchorXUnits: 'fraction',
                anchorYUnits: 'pixels',
                src: '{{
asset('landing_page/icon/iconred.png') }}'
              })
            });

          } else if (marker.name ==
"Kluster 3") {
            markerStyle = new
ol.style.Style({
              image: new ol.style.Icon({
                anchor: [0.5, 46],
                anchorXUnits: 'fraction',
                anchorYUnits: 'pixels',
                src: '{{
asset('landing_page/icon/iconred.png') }}'
              })
```

```
            });

        }

markerFeature.setStyle(markerStyle);

        var vectorSource = new
ol.source.Vector({
            features: [markerFeature]
        });

        var vectorLayer = new
ol.layer.Vector({
            source: vectorSource
        });

        map2.addLayer(vectorLayer);

        var radius = 200; // in meters

        var circle = new ol.Feature({
            geometry: new
ol.geom.Circle(ol.proj.fromLonLat([marke
r.longitude, marker.latitude]),
                radius),
        });

        var circleLayer = new
ol.layer.Vector({
            source: new
ol.source.Vector({
                features: [circle],
            }),
            style: new ol.style.Style({
                stroke: new
ol.style.Stroke({
                    color: 'blue',
                    width: 2,
                }),
                fill: new ol.style.Fill({
                    color: 'rgba(0, 0, 255,
0.1)', // Transparent fill color
                }),
            }),
        });

        map2.addLayer(circleLayer);
    });

}
```

```
    // Load JSON data and initialize the
map
    json2 = <?php echo
json_encode($markers2); ?>;
    console.log(json2);

    initMap2(json2);
  </script>
@endsection
```