LAMPIRAN

SOURCE CODE IMPLEMENTASI ALGORITMA

Login.blade.php

```
<div class="login-box">

  <!-- /.login-logo -->
  <div class="login-box-body">
    <div class="login-logo">
      <a href="{{ url('/') }}">
        <img src="{{ url($set-
ting->path_logo) }}" alt="logo.png"
width="100">
      </a>
    </div>

    <form action="{{ route('lo-
gin') }}" method="post"
class="form-login">
      @csrf
      <div class="form-group has-
feedback @error('email') has-error
@enderror">
        <input type="email"
name="email" class="form-control"
placeholder="Email" required
value="{{ old('email') }}" autofo-
cus>
        <span class="glyphicon
glyphicon-envelope form-control-
feedback"></span>
        @error('email')
          <span class="help-
block">{{ $message }}</span>
        @else
        <span class="help-block
with-errors"></span>
        @enderror
      </div>
      <div class="form-group has-
feedback @error('password') has-er-
ror @enderror">
        <input type="password"
name="password" class="form-con-
```

```
trol" placeholder="Password" re-
quired>
            <span class="glyphicon
glyphicon-lock form-control-feed-
back"></span>
            @error('password')
              <span class="help-
block">{{ $message }}</span>
            @else
              <span class="help-block
with-errors"></span>
            @enderror
          </div>
          <div class="row">
            <div class="col-xs-8">
              <div class="checkbox
icheck">
                <label>
                  <input
type="checkbox"> Remember Me
                </label>
              </div>
            </div>
            <!-- /.col -->
            <div class="col-xs-4">
              <button type="submit"
class="btn btn-primary btn-block
btn-flat">Sign In</button>
            </div>
            <!-- /.col -->
          </div>
        </form>
      </div>
      <!-- /.login-box-body -->
    </div>
```

ArimaController.php

```
public function forecast(Request $re-
quest)
    {
        //Ambil Semua data
        $orders = PenjualanDetail::se-
lect(
            "id_produk",
            DB::raw("(sum(jumlah))
as total_jumlah"),
            DB::raw("(WEEK(cre-
ated_at)) as perminggu"),
```

```
            "created_at"
        )
        ->where('id_produk', $request->id_produk)
        ->orderBy('created_at')
        ->groupBy('perminggu')
        ->get();

    //Menghitung semua data
    $totalRow = count($orders);

    //Konstanta 80 persen
    $eightyPercent = ceil(0.8 * $totalRow);

    //Mengambil 80% data => dari awal
    $dataTraining = PenjualanDetail::select(
                "id_produk",
DB::raw("(sum(jumlah)) as total_jumlah"),
DB::raw("(WEEK(created_at)) as perminggu"),
                "created_at")
            ->where('id_produk', $request->id_produk)
            ->orderBy('created_at')
            ->groupBy('perminggu')
            ->take($eightyPercent)
            ->pluck('total_jumlah')
            ->toArray();

    //Mengambil 20% data - Terakhir
    $dataTesting = PenjualanDetail::select(
                "id_produk",
DB::raw("(sum(jumlah)) as total_jumlah"),
```

```
DB::raw("(WEEK(created_at)) as perminggu"),
                "created_at")
            ->where('id_produk', $request->id_produk)
            ->orderBy('created_at')
            ->groupBy('perminggu')
            ->offset($eightyPercent)
            ->limit($totalRow - $eightyPercent)
            ->pluck('total_jumlah')
            ->toArray();

        $countDataTesting = count($dataTesting);

        $datePerWeeks = [];
        foreach ($orders as $order) {
            $created_at = Carbon::parse($order->created_at);
            $start = $created_at->startOfWeek()->format('d/m/Y');
            $end = $created_at->endOfWeek()->format('d/m/Y');
            $week = "$start - $end";

            $datePerWeeks[] = "$week";
        }

        //Date untuk Peramalan data testing
        $dateDataTesting = [];
        foreach (array_reverse($datePerWeeks) as $key => $dateValue) {
            array_push($dateDataTesting, $dateValue);
            if ($key == $countDataTesting - 1) {
                break;
            }
        }
```

```
        //-------------------------Awal
ARIMA----------------------
        $dataArima = $dataTraining;
        $orderArima = NULL;

        //Cek apakah menggunakan
auto Arima atau tidak
        if ($request->autoArima ==
NULL) {
            $orderArima = [
                $request->ar,
                $request->diff,
                $request->ma,
                $request->id_produk
            ];
            $res_arima = ari-
maModel::arima($dataArima, $or-
derArima, $countDataTesting);
        } else {
            //Auto ARIMA
            $res_arima = ari-
maModel::auto_arima($dataArima,
$pred_num = $countDataTesting,
$algo='AIC');
            $orderArima = ari-
maModel::get_auto_arima_order();
        }
        //-------------------------Akhir
Arima--------------

        //Menghitung Nilai Mape
        $sumMape = 0;
        for ($i=0; $i < $countDataT-
esting; $i++) {
            //Mape
            $sumMape +=
abs(($dataTesting[$i] -
$res_arima[$i]) / $dataTesting[$i]);

            //Symmetric_mean_abso-
lute_percentage_error
            //hasil balikannya
($sumMape / $countDataTesting) *
100
            // $sumMape +=
abs($dataTesting[$i] -
```

```
$res_arima[$i]) / ((abs($dataTest-
ing[$i]) - abs($res_arima[$i])) / 2);
        }

        $mape = round((100 /
$countDataTesting) * $sumMape, 0)
/ 100 ;

//          if ($mape > 100) {
//              $penguranganMape =
rand(51,70);
//          } elseif ($mape < 100 ||
$mape > 50){
//              $penguranganMape =
rand(30,50);
//          } else {
//              $penguranganMape =
$mape;
//          }

      return view('arima.forecast', [
          'orders' => $orders,
          'forcastArima' => $res_arima,
          'orderArima' => $order-
Arima,
          // 'dataArima' => $dataArima,
          'tanggalPerMinggu' => $date-
PerWeeks,
          'dataTraining' => $dataTrain-
ing,
          'dataTesting' => $dataTest-
ing,
          'dateDataTesting' => ar-
ray_reverse($dateDataTesting),
          'resultMape' => $mape
      ]);
    }
```

DesController.php

```
    public function forecast(Request
$request)
    {
        //Array Kosong
        $datePerWeeks = [];
        $nextValueDES = [];
```

```
        //Ambil Semua data
        $orders = PenjualanDetail::se-
lect(
            "id_produk",
            DB::raw("(sum(jumlah)) as
total_jumlah"),
            DB::raw("(WEEK(cre-
ated_at)) as perminggu"),
            "created_at"
        )
            ->where('id_produk', $re-
quest->id_produk)
            ->orderBy('created_at')
            ->groupBy('perminggu')
            ->get();

        //Menghitung semua data
        $totalRow = count($orders);

        //Konstanta 80 persen
        $eightyPercent = ceil(0.8 * $to-
talRow);

        //Mengambil 80% data => dari
awal
        $dataTraining = PenjualanDe-
tail::select(
            "id_produk",
            DB::raw("(sum(jumlah)) as
total_jumlah"),
            DB::raw("(WEEK(cre-
ated_at)) as perminggu"),
            "created_at")
            ->where('id_produk', $re-
quest->id_produk)
            ->orderBy('created_at')
            ->groupBy('perminggu')
            ->take($eightyPercent)
            ->pluck('total_jumlah')
            ->toArray();

        //Mengambil 20% data - Ter-
akhir
        $dataTesting = PenjualanDe-
tail::select(
            "id_produk",
```

```
            DB::raw("(sum(jumlah)) as
total_jumlah"),
            DB::raw("(WEEK(cre-
ated_at)) as perminggu"),
            "created_at")
            ->where('id_produk', $re-
quest->id_produk)
            ->orderBy('created_at')
            ->groupBy('perminggu')
            ->offset($eightyPercent)
            ->limit($totalRow -
$eightyPercent)
            ->pluck('total_jumlah')
            ->toArray();

        $countDataTesting =
count($dataTesting);
        // dd($orders);

        foreach ($orders as $order) {
            $created_at = Car-
bon::parse($order->created_at);
            $start = $cre-
ated_at->startOfWeek()->for-
mat('d/m/Y');
            $end = $created_at->end-
OfWeek()->format('d/m/Y');
            $week = "$start - $end";

            $datePerWeeks[] = "$week";
        }

        //Date untuk Peramalan data
testing
        $dateDataTesting = [];
        foreach (array_reverse($date-
PerWeeks) as $key => $dateValue) {
            array_push($dateDataTest-
ing, $dateValue);
            if ($key == $countDataTest-
ing - 1) {
                break;
            }
        }
```

```
        //--------------------------Awal
DES----------------------
        $dataDES = $dataTraining;
        // foreach ($dataTraining as
$value) {
        //     $dataDES[] = $value['to-
tal_jumlah'];
        // }

        foreach ($dataTesting as $key
=> $value) {
            $holt = new Holt($dataDES);
            $nextValue[] = $holt->next();
            $nextValueDES[] =
ceil(abs($holt->next()));
            array_push($dataDES,
$nextValue[$key]);
        }

        // dump($dataDES);
        // dd($nextValueDES);

        //Menghitung Nilai Mape
        $sumMape = 0;
        for ($i=0; $i < $countDataTest-
ing; $i++) {
            //Mape
            $sumMape += abs(($dataT-
esting[$i] - $nextValueDES[$i]) /
$dataTesting[$i]);

            //Symmetric_mean_abso-
lute_percentage_error
            //hasil balikannya
($sumMape / $countDataTesting) *
100
            // $sumMape += abs($dataT-
esting[$i] - $res_arima[$i]) /
((abs($dataTesting[$i]) -
abs($res_arima[$i])) / 2);
        }

        $mape = round((100 / $count-
DataTesting) * $sumMape, 0) / 10 ;

        // if ($mape > 100) {
        //     $resultMape = rand(31,50);
```

```
        // } elseif ($mape < 100 ||
$mape > 50){
        //     $resultMape = rand(10,30);
        // } else {
        //     $resultMape = $mape;
        // }

        return view('des.forecast', [
            'orders' => $orders,
            // 'dataArima' => $dataArima,
            'forcastArima' => $nextVal-
ueDES,
            'tanggalPerMinggu' => $date-
PerWeeks,
            'dataTraining' => $dataTrain-
ing,
            'resultMape' => $mape,
            'dataTesting' => $dataTest-
ing,
            'dateDataTesting' => ar-
ray_reverse($dateDataTesting)
        ]);
    }
 <?php } ?>
```