Lampiran 3 Kodingan Program

Source Code Metode ARIMA.

```
<?php

namespace App;

class TimeSeries
{
    // Method untuk menghitung rata-rata dari time series
     public function mean($data)
     {
          return array_sum($data) / count($data);
      }

      // Method untuk menghitung ACF
      public function calculateACF($data)
      {
          $n = count($data);
          $mean = $this->mean($data);
          $acf = [];

          for ($lag = 0; $lag < $n; $lag++) {
              $numerator = 0;
              $denominator = 0;

              for ($t = 0; $t < $n - $lag; $t++) {
                  $numerator += ($data[$t] - $mean) *
                    ($data[$t + $lag] - $mean);
              }

              for ($t = 0; $t < $n; $t++) {
                  $denominator += ($data[$t] - $mean) **
2;
              }

              $acf[$lag] = $numerator / $denominator;
         }

          return $acf;
      }

      // Method untuk menghitung Standard ACF (SACF)
      public function calculateSACF($acf)
      {
```

```
41          $n = count($acf);
42          $sacf = [];
43
44          for ($k = 1; $k < $n; $k++) {
45              $sum_rj2 = 0;
46              for ($j = 1; $j <= $k - 1; $j++) {
47                  $sum_rj2 += $acf[$j] ** 2;
48              }
49
50              $sacf[$k] = sqrt((1 + 2 * $sum_rj2) / ($n -
                      $k));
51          }
52
53          return $sacf;
54      }
55
56      // Method untuk menghitung T-statistic ACF (TACF)
57      public function calculateTACF($acf, $sacf)
58      {
59          $n = count($acf);
60          $tacf = [];
61
62          for ($k = 1; $k < $n; $k++) {
63              $tacf[$k] = $acf[$k] / $sacf[$k];
64          }
65
66          return $tacf;
67      }
68  }
```

Source Code Metode Double Exponential Smoothing.

```
1  <?php
2
3  namespace App;
4
5
6  class DoubleExponentialSmoothingService
7  {
8          public  function  predict($data,  $alpha,  $beta,
$n_preds)
9      {
10         $n = count($data);
11         $level = [$data[0]];
12         $trend = [$data[1] - $data[0]];
13         $result = [$data[0]];
14
15         for ($i = 1; $i < $n; $i++) {
16            $level[] = $alpha * $data[$i] + (1 - $alpha) *
                   ($level[$i - 1] + $trend[$i - 1]);
17             $trend[] = $beta * ($level[$i] - $level[$i -
1])          + (1 - $beta) * $trend[$i - 1];
18            $result[] = $level[$i] + $trend[$i];
19         }
20
21         for ($i = 0; $i < $n_preds; $i++) {
22                $result[] = $level[$n - 1] + ($i + 1) *
$trend[$n- 1];
23         }
24
25         return $result;
26      }
27  }
```