BAB V
IMPLEMENTASI SISTEM

Pada bab V akan menjelaskan tentang desain program dan koding program. Berikut ini tampilan-tampilan halaman yang ada di dalam program yaitu tampilan dari sisi pengguna saja.

5.1 Spesifikasi Perangkat Keras

Perangkat keras yang digunakan dalam pembuatan aplikasi adalah:

1. Intel® Core™ i7-8750H 2.20GHz - 4.00GHz.
2. Memory DDR4 16 GB.
3. Hardisk 1 TB, SSD 240 GB.
4. Monitor 15,2".

5.2 Spesifikasi Perangkat Lunak

1. Android Studio.
2. Sublime Text Editor.
3. Snipping Tool.
4. Navicat.
5. XAMPP.
6. UC Browser.

5.3. Batasan Implementasi Sistem

Tahapan implementasi pada sistem ini merupakan kelanjutan dari tahapan perancangan sistem yang telah di uraikan sebelumnya pada bab IV, pada bab ini juga akan diuraikan implementasi proses sistem dari perancangan sistem yang telah dibuat sebelumnya.

5.4. Implementasi Sistem

Implementasi pada sistem ini memiliki beberapa proses yang sudah di uraikan pada bab VI, dan terbagi menjadi berberapa proses, dan form. Koding dari implementasi sistem ini cantumkan pada bab ini.

5.5 Implementasi Program

Berikut ini koding dari file PHP yang digunakan untuk mengolah data dari/ke database dan android.

1. Dijkstra.php

   File ini adalah *library* untuk perhitungan Dijkstra.

```
<?php
class Dijkstra {

   var $visited = array();
   var $distance = array();
   var $previousNode = array();
   var $startnode =null;
   var $map = array();
   var $infiniteDistance = 0;
   var $numberOfNodes = 0;
   var $bestPath = 0;
   var $matrixWidth = 0;

   function Dijkstra(&$ourMap, $infiniteDistance) {
      $this -> infiniteDistance = $infiniteDistance;
      $this -> map = &$ourMap;
      $this -> numberOfNodes = count($ourMap);
      $this -> bestPath = 0;
   }

   function findShortestPath($start,$to) {
      $this -> startnode = $start;
      for ($i=0;$i<$this -> numberOfNodes;$i++) {
         if ($i == $this -> startnode) {
            $this -> visited[$i] = true;
            $this -> distance[$i] = 0;
         } else {
            $this -> visited[$i] = false;
```

```
            $this -> distance[$i] = isset($this -> map[$this -> startnode][$i])
               ? $this -> map[$this -> startnode][$i]
               : $this -> infiniteDistance;
         }
         $this -> previousNode[$i] = $this -> startnode;
      }

      $maxTries = $this -> numberOfNodes;
      $tries = 0;
      while (in_array(false,$this -> visited,true) && $tries <= $maxTries) {
         $this -> bestPath = $this->findBestPath($this->distance,array_keys($this ->
visited,false,true));
         if($to !== null && $this -> bestPath === $to) {
            break;
         }
         $this -> updateDistanceAndPrevious($this -> bestPath);
         $this -> visited[$this -> bestPath] = true;
         $tries++;
      }
   }

   function findBestPath($ourDistance, $ourNodesLeft) {
      $bestPath = $this -> infiniteDistance;
      $bestNode = 0;
      for ($i = 0,$m=count($ourNodesLeft); $i < $m; $i++) {
         if($ourDistance[$ourNodesLeft[$i]] < $bestPath) {
            $bestPath = $ourDistance[$ourNodesLeft[$i]];
            $bestNode = $ourNodesLeft[$i];
         }
      }
      return $bestNode;
   }

   function updateDistanceAndPrevious($obp) {
      for ($i=0;$i<$this -> numberOfNodes;$i++) {
         if(     (isset($this->map[$obp][$i]))
            &&    (!($this->map[$obp][$i] == $this->infiniteDistance) || ($this-
>map[$obp][$i] == 0 ))
            &&    (($this->distance[$obp] + $this->map[$obp][$i]) < $this ->
distance[$i])
         )
         {
```

```
                $this -> distance[$i] = $this -> distance[$obp] + $this ->
map[$obp][$i];
                $this -> previousNode[$i] = $obp;
            }
        }
    }

    function printMap(&$map) {
        $placeholder = ' %' . strlen($this -> infiniteDistance) .'d';
        $foo = '';
        for($i=0,$im=count($map);$i<$im;$i++) {
            for ($k=0,$m=$im;$k<$m;$k++) {
                $foo.= sprintf($placeholder, isset($map[$i][$k]) ? $map[$i][$k] : $this ->
infiniteDistance);
            }
            $foo.= "\n";
        }
        return $foo;
    }

    function getResults($to) {
        $ourShortestPath = array();
        $foo = '';
        for ($i = 0; $i < $this -> numberOfNodes; $i++) {
            if($to !== null && $to !== $i) {
                continue;
            }
            $ourShortestPath[$i] = array();
            $endNode = null;
            $currNode = $i;
            $ourShortestPath[$i][] = $i;
            while ($endNode === null || $endNode != $this -> startnode) {
                $ourShortestPath[$i][] = $this -> previousNode[$currNode];
                $endNode = $this -> previousNode[$currNode];
                $currNode = $this -> previousNode[$currNode];
            }
            $ourShortestPath[$i] = array_reverse($ourShortestPath[$i]);
            if ($to === null || $to === $i) {
            if($this -> distance[$i] >= $this -> infiniteDistance) {
                $foo .= sprintf("tidak ada rute dari %d ke %d. \n",$this -> startnode,$i);
            } else {
                $foo .= sprintf('Dari %d => ke  %d = %d tujuan [%d]: ikuti rute ini %s' ,
                        $this -> startnode,$i,$this -> distance[$i],
```

```
                count($ourShortestPath[$i]),
                implode('-',$ourShortestPath[$i]));
        }
        //$foo .= str_repeat('-',20) . "<br>";
          if ($to === $i) {
            break;
          }
        }
      }
      return $foo;
    }

    public function Distance($to){
      $ourShortestPath = array();
      $foo = '';
      for ($i = 0; $i < $this -> numberOfNodes; $i++) {
        if($to !== null && $to !== $i) {
          continue;
        }
        $ourShortestPath[$i] = array();
        $endNode = null;
        $currNode = $i;
        $ourShortestPath[$i][] = $i;
        while ($endNode === null || $endNode != $this -> startnode) {
          $ourShortestPath[$i][] = $this -> previousNode[$currNode];
          $endNode = $this -> previousNode[$currNode];
          $currNode = $this -> previousNode[$currNode];
        }
        $ourShortestPath[$i] = array_reverse($ourShortestPath[$i]);
        if ($to === null || $to === $i) {
          if($this -> distance[$i] >= $this -> infiniteDistance) {
            $foo .= sprintf("tidak ada rute dari %d ke %d. \n",$this ->
startnode,$i);
          } else {
            $foo .= sprintf($this -> distance[$i]);

          }
          //$foo .= str_repeat('-',20) . "<br>";
          if ($to === $i) {
            break;
          }
        }
      }
```

```
        return $foo;
    }
} // end class
?>
```

2. CalPath.php

file ini berfungsi untuk melakukan perhitungan algoritma Dijkstra.

```
<?php

include("dijkstra.php");
include 'koneksi.php';
define('I',1000);

// Size of the matrix
$matrixWidth = 100;

// $points is an array in the following format: (router1,router2,distance-between-
them)
$points = array(
      array("2","3",2.18),
      array("1","2",0.262),
      array("196","545",0.491),
      array("1","673",0.424),
      array("374","673",0.782),
      array("374","679",0.237),
      array("4","679",0.764),
      array("3","4",0.356),
      array("2","14",0.912),
      array("5","14",1.56),
      array("5","6",0.96),
      array("6","1",1.22),
      array("7","6",2.07),
      array("300","548",0.682),
      array("7","4",1.16),
      array("375","6",1.78),
      array("375","676",0.568),
      array("13","676",0.739),
      array("13","12",2.09),
      array("11","12",2.56),
      array("10","11",0.876),
      array("9","10",0.315),
```

```
    array("8","9",0.747),
    array("8","14",1.5),
    array("13","375",0.705),
    array("15","13",0.336),
    array("15","16",0.172),
    array("16","17",0.63),
    array("17","18",0.81),
    array("19","18",1.08),
    array("20","19",1.51),
    array("20","21",4.04),
    array("22","21",5.58),
   array("331","333",1.21),
);

$ourMap = array();

$node = array(
        array("1","1",-7.4510355,112.7153937),
        array("2","2",-7.4514982,112.7177167),
        array("3","3",-7.470978,112.716024),
        array("4","4",-7.4700738,112.7129341),
        array("5","5",-7.4397689,112.7055075),
        array("6","6",-7.4488116,112.704548),
        array("7","7",-7.4673223,112.7024881),
        array("8","8",-7.4303499,112.7219627),
        array("9","9",-7.4276689,112.7151392),
        array("10","10",-7.4284349,112.7123926),
        array("11","11",-7.424307,112.705097),
        array("12","12",-7.4226101,112.681381),
        array("13","13",-7.4415701,112.6830458),
        array("14","14",-7.443493,112.7194594),
        array("15","15",-7.4446496,112.6821487),
        array("16","16",-7.4458921,112.6812932),
        array("17","17",-7.4456069,112.6756352),
        array("18","18",-7.4525927,112.6735742),
        array("19","19",-7.4485927,112.6649912),
        array("20","20",-7.4461746,112.6515165),
        array("21","21",-7.4363684,112.6163131),
        array("22","22",-7.4106436,112.5784512),
        array("23","23",-7.4359035,112.573393),
        array("24","24",-7.4458302,112.570686),
        array("25","25",-7.4205527,112.6727387),
        array("26","26",-7.4183463,112.6578257),
```

```
        array("377","Pondok Pesantren Al-Fattah",-8.1197459,111.1488778),
        array("378","Pondok Tremas",-8.1169764,111.1439715),
        array("379","Pondok Pesantren Minhajul Muna",-
8.1113208,111.4267462),
        array("380","Arrisalah International program modern boarding sc",-
7.981996,111.433093),
        array("381","Ponpes Al Islam",-7.9309817,111.5096289),
        array("382","Pondok Modern Darussalam, Gontor",-
7.9280884,111.498257),
        array("383","Pondok Pesantren Sulamul Huda",-7.930932,111.518488),
        array("384","PPTQ ALHASAN",-7.8544659,111.4875342),
        array("385","Pondok Pesantren Wali Songo Ngabar",-
7.9173939,111.4741037),
        array("386","Pesantren Putri Al-Mawaddah",-7.9507503,111.5070052),
        array("387","Pondok Pesantren Darul Istiqomah",-
7.9507692,111.3990937),
        array("388","Pondok Pesantren Putra Al Iman Ponorogo (Islamic B",-
7.8443895,111.4018155),
        array("389","Pondok Pesantren Darul Falah Sukorejo",-
7.8352607,111.4274024),
        array("390","Darul Huda Mayak",-7.8679468,111.4894059),
        array("391","Pondok Pesantren Hudatul Muna Dua",-
7.8705556,111.45269),
        array("392","PP. Ittihadul Ummah",-7.8559714,111.4677562),
        array("393","Pondok Pesantren Kyai Haji Hasyim Asy'ari",-
7.8712889,111.4635242),
        array("394","Pondok Pesantren Putra Al Iman Ponorogo (Islamic B",-
7.8443895,111.4018155),
        array("395","Yayasan Pondok Pesantren dan Pendidikan An-Nuur",-
7.792625,111.469774),
        array("681","MTS As Syafi'iyah Pogalan",-8.0946233,111.7375708),
);

// Read in the points and push them into the map

for ($i=0; $i < count($points); $i++) {
   $x = $points[$i][0];
   $y = $points[$i][1];
   $c = $points[$i][2];
   $ourMap[$x][$y] = $c;
   $ourMap[$y][$x] = $c;
}
```

```
// ensure that the distance from a node to itself is always zero
// Purists may want to edit this bit out.

for ($i=0; $i < $matrixWidth; $i++) {
   for ($k=0; $k < $matrixWidth; $k++) {
      if ($i == $k) $ourMap[$i][$k] = 0;
   }
}

// initialize the algorithm class
$dijkstra = new Dijkstra($ourMap, I,$matrixWidth);

// $dijkstra->findShortestPath(0,13); to find only path from field 0 to field 13...

$query1 = mysqli_query($con, "SELECT * FROM temp_table");

while ($isi = $query1->fetch_object()) {
     $lat_asal   = $isi->lat_temp;
     $lon_asal   = $isi->lon_temp;
     $toClass    = $isi->tujuan_temp;
}
/*$lat_asal   = -7.321183;
$lon_asal   = 112.731321;
$toClass    = 513;*/

//$dijkstra->findShortestPath($fromClass, $toClass);

// Display the results

//echo '<pre>';
//echo "the map looks like:\n\n";
//echo $dijkstra -> printMap($ourMap);
//echo "rute terpendek dari  ".$fromClass." ke ".$toClass." adalah  :<br>";
//$hasil = $dijkstra -> getResults((int)$toClass);

for ($i=0; $i < count($node); $i++) {
   $id          = $node[$i][0];
   $nama        = $node[$i][1];
   $lat_tujuan  = $node[$i][2];
   $lon_tujuan  = $node[$i][3];

   $rad    = 0.0174532925;
   $r      = 6371;
```

```
    $lat1   = $lat_asal * $rad;
    $lon1   = $lon_asal * $rad;
    $lat2   = $lat_tujuan * $rad;
    $lon2   = $lon_tujuan * $rad;
    $x      = ($lon2 - $lon1) * (cos(($lat1 + $lat2) / 2));
    $y      = ($lat2 - $lat1);
    $h      = ((sqrt($x * $x + $y * $y) * $r));
    $hasil_jarak[] = number_format($h, 3, ".", ",");
}


//mencari index jarak terdekat
$index = array_search(min($hasil_jarak), $hasil_jarak, true);
$index_min = $index + 1;

$query = mysqli_query($con, "SELECT * FROM node WHERE id_node =
'$index_min'");
while ($isi = $query->fetch_object()) {
      $id_min = $isi->id_node;
      $nama_min = $isi->nama_node;
      $lat_min = $isi->latitude;
      $lon_min = $isi->longitude;
}
//die(var_dump($lon_min));

    //   echo $nama_min;
    //   echo $lat_min;
    //   echo $lon_min;

mysqli_query($con, "UPDATE awal SET nama_node='$nama_min',
lat_node='$lat_min', lon_node='$lon_min' WHERE id_node='awal'");

// if ($abc){
//     echo "sukses";
// } else {
//     echo "gagal";
// }

$dijkstra->findShortestPath($id_min, $toClass);

// Display the results

//echo '<pre>';
```

```
//echo "the map looks like:\n\n";
//echo $dijkstra -> printMap($ourMap);
$distance = $dijkstra->Distance((int)$toClass);
"rute terpendek dari  ".$id_min." ke ".$toClass." adalah  :";
$hasil1 = $dijkstra -> getResults((int)$toClass);

$awal       = substr($hasil1, 0, 21);
$akhir      = substr($hasil1, 21);
$c          = explode(" ", $hasil1);

if (strlen($c[7]) ==  1) {
     $hasil1 = $awal.'00'.$akhir;
} elseif (strlen($c[7]) == 2) {
     $hasil1 = $awal.'0'.$akhir;
} elseif (strlen($c[7]) == 3){
     $hasil1;
}

mysqli_query($con, "UPDATE jarak_node SET jarak = '$distance' WHERE
id_jarak = 'jarak'");
//echo '</pre>';
$node       = $c[13];
$a          = explode("-", $node);
$response["node"] = array();
mysqli_query($con, "DELETE FROM node_jalur");

for ( $i = 0; $i < count($a); $i++ ) {
     $id         = $a[$i];
     $query_node = mysqli_query($con, "SELECT latitude,longitude FROM node
WHERE id_node='$id'");
     while ($isi = $query_node->fetch_object()) {
          $lat_node = $isi->latitude;
          $lon_node = $isi->longitude;
     }

     $query_jarak = mysqli_query($con, "SELECT * FROM temp_table WHERE
id_temp='jarak'");
     while ($isi = $query_jarak->fetch_object()) {
          $lat = $isi->lat_temp;
          $lon = $isi->lon_temp;
     }
```

```
    mysqli_query($con, "INSERT INTO
node_jalur(node,latitude_jalur,longitude_jalur)
VALUES('$a[$i]','$lat_node','$lon_node')");

    $b["hasil"]       = $a[$i];
    $b["latitude"]    = $lat_node;
    $b["longitude"]   = $lon_node;
    $b["jarak"]       = $distance;
    array_push($response["node"], $b);
}

$response["success"] = 1;

echo json_encode($response);

?>
```

3. Read_ponpes.php

File ini berfungsi untuk membaca data ponpes yang berada di database.

```
<?php

/*
 * Berikut adalah kelas untuk membaca data ponpes
 */

// array for JSON response
$response = array();

// include koneksi class
include 'koneksi.php';

$sql = "SELECT * FROM node WHERE id_node >= 377";
$ponpes_result = mysqli_query ($con, $sql); //run the query

 // check for empty result
 if (mysqli_num_rows($ponpes_result) > 0) {

  $response["ponpes"] = array();
  $no = 1;
```

```
  while ($row = mysqli_fetch_array($ponpes_result)) {
   // temp user array
   $ponpes = array();
   $ponpes["no"]        = $no++;
   $ponpes["nama"]      = $row["nama_node"];
   $ponpes["alamat"]    = 'null';
   $ponpes["latitude"]  = $row["latitude"];
   $ponpes["longitude"] = $row["longitude"];

   // push single puasa into final response array
   array_push($response["ponpes"], $ponpes);
  }
  // success
  $response["success"] = 1;

  // echoing JSON response
  echo json_encode($response);
 } else {
  $response["success"] = 0;
  $response["message"] = "Tidak ada data ponpes";

  // echo no users JSON
  echo json_encode($response);
 }
?>
  // echoing JSON response
  echo json_encode($response);
 } else {
  $response["success"] = 0;
  $response["message"] = "Tidak ada data ponpes";

  // echo no users JSON
  echo json_encode($response);
 }
?>
```

4. Node.php

   File ini digunakan untuk membaca dat node yang berada di database.

```
<?php
        include("koneksi.php");
```

```
        if(isset($_GET['mode'])){

                //cek nilai dari mode
                if($_GET['mode'] == "getDataNode"){

                                $result=mysqli_query($con, "SELECT * FROM
node WHERE id_node >= 377 AND id_node != 681 ORDER BY nama_node
ASC");
                                $rows=array();
                                while($row = mysqli_fetch_assoc($result)){
                                        $rows[]=$row;
                                }

                                if(mysqli_num_rows($result) > 0){
                                        //convert response to JSON format
                                        $data ="{values:
".json_encode($rows)."}";
                                        echo $data;
                                }else{
                                        echo "nodata";
                                }
                }
        }

?>
```

5. Awal_akhir.php

File ini berfungsi untuk membaca node awal dan node akhir dari database.

```
<?php

/*
 * Berikut adalah kelas untuk membaca data pondok
 */

// array for JSON response
$response = array();

// include koneksi class
include 'koneksi.php';

$awal_result = mysqli_query ($con, "SELECT * FROM awal");
```

```
// check for empty result
if (mysqli_num_rows($awal_result) > 0) {

 $response["node_awal"] = array();
 $no = 1;

 while ($row = mysqli_fetch_array($awal_result)) {
  // temp user array
  $point = array();
  $point["id_node"]   = $row["id_node"];
  $point["nama_node"] = $row["nama_node"];
  $point["lat_node"]  = $row["lat_node"];
  $point["lon_node"]  = $row["lon_node"];

  // push single puasa into final response array
  array_push($response["node_awal"], $point);
 }

 // success
 $response["success"] = 1;

 // echoing JSON response
 echo json_encode($response);
} else {
 $response["success"] = 0;
 $response["message"] = "Tidak ada data pondok";

 // echo no users JSON
 echo json_encode($response);
 }
?>
```

6. Update_awal.php

File ini berfungsi untuk mengupdate node awal, dan menampilkannya ke android.

```
<?php
include 'koneksi.php';
$node = array(
        array("1","Node 1",-7.058686,112.7952738),
        array("2","Node 2",-7.1139906,113.321909),
        array("3","Node 3",-7.1245747,113.3843829),
```

```
        array("4","Node 4",-7.2172278,113.3958764),
        array("5","Node 5",-7.1609344,113.4867306),
        ……..
        array("379","MOJOKERTO_1",-7.465081,112.432054),
        array("380","MOJOSARI",-7.520808,112.562845),
        array("381","JOMBANG_1",-7.537751,112.233111),
        array("382","JOMBANG_2",-7.537617,112.232962),
        array("383","JOMBANG_3",-7.537741,112.234511)
    );

$query1 = mysqli_query($con, "SELECT * FROM temp_table");

while ($isi = $query1->fetch_object()) {
    $lat_asal   = $isi->lat_temp;
    $lon_asal   = $isi->lon_temp;
    $toClass    = $isi->tujuan_temp;
}

for ($i=0; $i < count($node); $i++) {
   $id          = $node[$i][0];
   $nama        = $node[$i][1];
   $lat_tujuan  = $node[$i][2];
   $lon_tujuan  = $node[$i][3];

   $rad    = 0.0174532925;
   $r      = 6371;
   $lat1   = $lat_asal * $rad;
   $lon1   = $lon_asal * $rad;
   $lat2   = $lat_tujuan * $rad;
   $lon2   = $lon_tujuan * $rad;
   $x      = ($lon2 - $lon1) * (cos(($lat1 + $lat2) / 2));
   $y      = ($lat2 - $lat1);
   $h      = ((sqrt($x * $x + $y * $y) * $r));
   $hasil_jarak[] = number_format($h, 3, ".", ",");
}

//mencari index jarak terdekat
$index = array_search(min($hasil_jarak), $hasil_jarak);
$index_min = $index + 1;

$query = mysqli_query($con, "SELECT * FROM node WHERE id_node =
'$index_min'");
while ($isi = $query->fetch_object()) {
```

```
    $id_min = $isi->id_node;
    $nama_min = $isi->nama_node;
    $lat_min = $isi->latitude;
    $lon_min = $isi->longitude;
}

$z = mysqli_query($con, "UPDATE awal SET nama_node='$nama_min',
lat_node='$lat_min', lon_node='$lon_min' WHERE id_node='awal'");

//$response['hasil'] = array();
if ($z) {
    $response['success'] = 1;
    echo json_encode($response);
} else {
    $response["success"] = 0;
    $response["message"] = "Tidak ada data pondok";
    echo json_encode($response);
}
?>
```

7. Input.php

File ini berfungsi untuk menginput node awal dan node akhir ke database.

```
<?php


/*
 * Berikut adalah kelas untuk meng-update data mahasiswa
 */
include 'koneksi.php';

// array for JSON response
$response = array();

// check for required fields
if (isset($_POST['tujuan']) && isset($_POST['latitude']) &&
isset($_POST['longitude'])) {

    //$fromClass = $_POST['asal'];
    $toClass = $_POST['tujuan'];
    $lat_asal = $_POST['latitude'];
    $lon_asal = $_POST['longitude'];
```

```
    $node_akhir = mysqli_query($con, "SELECT * FROM node WHERE id_node
= '$toClass'");
    while ($isi = $node_akhir->fetch_object()) {
            $nama_node  = $isi->nama_node;
            $lat        = $isi->latitude;
            $lon        = $isi->longitude;
      }

    mysqli_query($con, "UPDATE akhir SET nama_node='$nama_node',
lat_node='$lat', lon_node='$lon' WHERE id_node='akhir'");

    $query = mysqli_query($con, "UPDATE temp_table SET id_temp='jarak',
lat_temp='$lat_asal', lon_temp='$lon_asal', tujuan_temp='$toClass' WHERE
id_temp='jarak'");

    // check if row inserted or not
    if ($query) {
        // successfully updated
        $response["success"] = 1;
        $response["message"] = "inputan diterima";

        // echoing JSON response
        echo json_encode($response);
    } else {
        $response["success"] = 0;
        $response["message"] = "inputan tidak diterima";

        // echoing JSON response
        echo json_encode($response);
    }
} else {
    // required field is missing
    $response["success"] = 0;
    $response["message"] = "isi semua form isian";

    // echoing JSON response
    echo json_encode($response);
}
?>
```

8. Create_jw.php

File ini berfungsi untuk mengambil jarak dari *direction* API, dan memasukkan ke database.

```
<?php
include 'koneksi.php';
$lat_awal = $_POST['lat_awal'];
$lon_awal = $_POST['lon_awal'];
$lat_akhir = $_POST['lat_akhir'];
$lon_akhir = $_POST['lon_akhir'];
// $lat_awal = -7;
// $lon_awal = 112;
// $lat_akhir = -6;
// $lon_akhir = 112;

$url =
file_get_contents('https://maps.googleapis.com/maps/api/directions/json
?origin='.$lat_awal.','.$lon_awal.'&destination='.$lat_akhir.','.$lon_akhir
.'&avoid=tolls&mode=driving&key=AIzaSyBTVcGt2fegGE6taEgiwhr
QL7QWU5dgJC0');

//$insert = mysqli_query($con, "INSERT INTO
jarak_waktu(jarak,waktu) VALUES('$jarak','$waktu')");

$response["jarwak"] = array();
//if ($url != null) {
        $data = json_decode($url, true);

        $jarak = $data['routes'][0]['legs'][0]['distance']['text'];
        $waktu = $data['routes'][0]['legs'][0]['duration']['text'];

        $jarwak = array();
        $jarwak["jarak"]    = $jarak;
        $jarwak["waktu"]    = $waktu;

        // push single puasa into final response array
        array_push($response["jarwak"], $jarwak);
        // successfully inserted into database
   $response["success"] = 1;

   // echoing JSON response
   echo json_encode($response);
// } else {
//     // failed to insert row
```

```
        //    $response["success"] = 0;
        //    $response["message"] = "Oops! An error occurred.";

        //    // echoing JSON response
        //    echo json_encode($response);
        // }
?>
```

9. Read_jarak.php

   File ini berfungsi untuk membaca jarak dari database, kemudian menampilkannya ke android.

```
<?php

/*
 * Berikut adalah kelas untuk membaca data pondok
 */

// array for JSON response
$response = array();

// include koneksi class
include 'koneksi.php';

$jarak_result = mysqli_query ($con, "SELECT * FROM jarak_node WHERE
id_jarak = 'jarak'"); //run the query

 // check for empty result
 if (mysqli_num_rows($jarak_result) > 0) {

  $response["jarak"] = array();

  while ($row = mysqli_fetch_array($jarak_result)) {
   // temp user array
   $pondok = array();

   $waktu = $row["jarak"] / 40;

   $time = number_format($waktu, 2, ',', '.');

   $pondok["jarak"] = $row["jarak"];
```

```
    $pondok["waktu"] = $time;

    // push single puasa into final response array
    array_push($response["jarak"], $pondok);
   }
   // success
   $response["success"] = 1;

   // echoing JSON response
   echo json_encode($response);
  } else {
   $response["success"] = 0;
   $response["message"] = "Tidak ada data pondok";

   // echo no users JSON
   echo json_encode($response);
  }
?>
```

10. Polyline.php

File ini berfungsi untuk menampilkan node yang dilalui hasil pergitungan Dijkstra ke android.

```
<?php

/*
 * Berikut adalah kelas untuk membaca data pondok
 */

// array for JSON response
$response = array();

// include koneksi class
include 'koneksi.php';

$jarak_result = mysqli_query ($con, "SELECT * FROM node_jalur"); //run the
query

 // check for empty result
 if (mysqli_num_rows($jarak_result) > 0) {

  $response["polyline"] = array();
```

```
 while ($row = mysqli_fetch_array($jarak_result)) {
  // temp user array
  $pondok = array();
  $pondok["node"]    = $row["node"];
  $pondok["latitude"]    = $row["latitude_jalur"];
  $pondok["longitude"]    = $row["longitude_jalur"];

  // push single puasa into final response array
  array_push($response["polyline"], $pondok);
 }
 // success
 $response["success"] = 1;

 // echoing JSON response
 echo json_encode($response);
} else {
 $response["success"] = 0;
 $response["message"] = "Tidak ada data pondok";

 // echo no users JSON
 echo json_encode($response);
}
?>
```

11. Koneksi.php

File ini berfungsi untuk menghubungkan file PHP dengan database.

```
<?php
        define('DB_USER', "root"); // User database
        define('DB_PASSWORD', ""); // Password database
        define('DB_DATABASE', "ta_sindy"); // Nama database
        define('DB_SERVER', "localhost"); // Server database


        // Koneksi ke database
        $con = mysqli_connect(DB_SERVER, DB_USER, DB_PASSWORD,
DB_DATABASE) or die('Unable to connect');
?>
```