BAB IV

HASIL DAN PEMBAHASAN

4.1. ANALISA KEBUTUHAN

Kebutuhan yang diperlukan untuk penelitian ini adalah sebagai berikut:

1. Kebutuhan Hardware

   Satu unit Notebook dengan spesifikasi sebagai berikut:

   Processor: Intel Core i7-10750H

   RAM: 8 GB DDR4

   Storage: 512 GB SSD

2. Kebutuhan Software

   Software yang dibutuhkan untuk penelitian ini adalah

   a. Sistem Operasi Windows 10

   b. *Google* Crome

   c. *Google Spreadsheet*

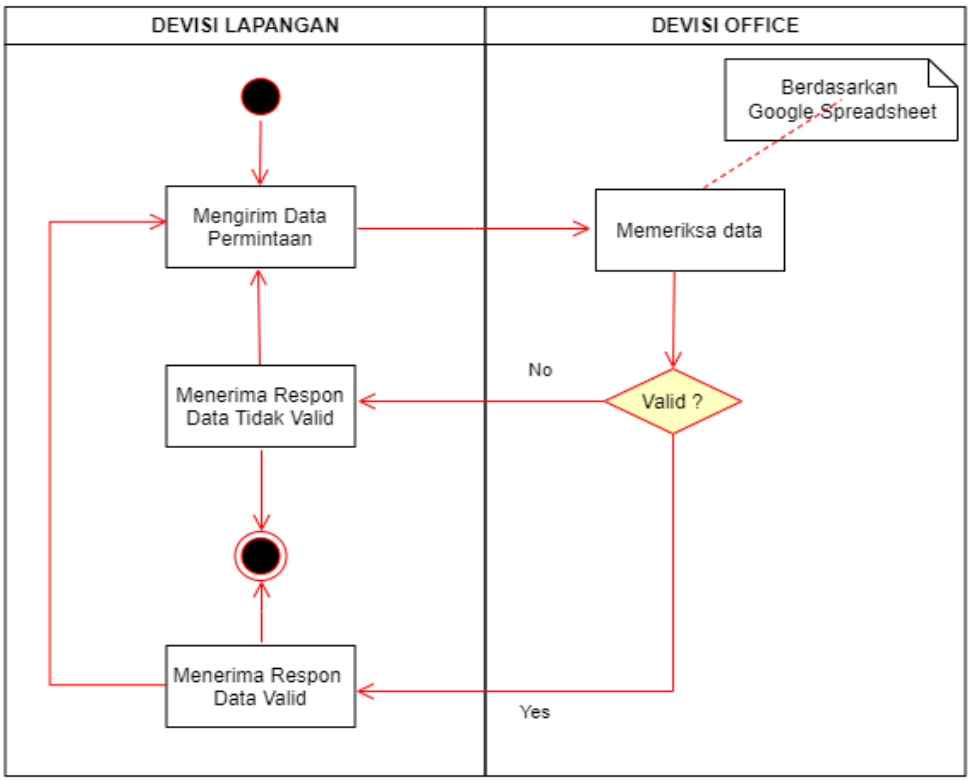
   d. Telegram

4.2. ALUR KERJA SISTEM

Alur kerja sistem adalah tahapan yang menggambarkan suatu sistem bekerja, pada penelitian ini alur kerja sistem dibagi menjadi dua yaitu sebagai berikut:

1. Alur Sistem Lama

Alur sistem ini menggambarkan alur sistem yang digunakan saat sistem baru belum di gunakan. Adapun entitas yang terlibat dapat digambarkan pada activity diagram di bawah ini



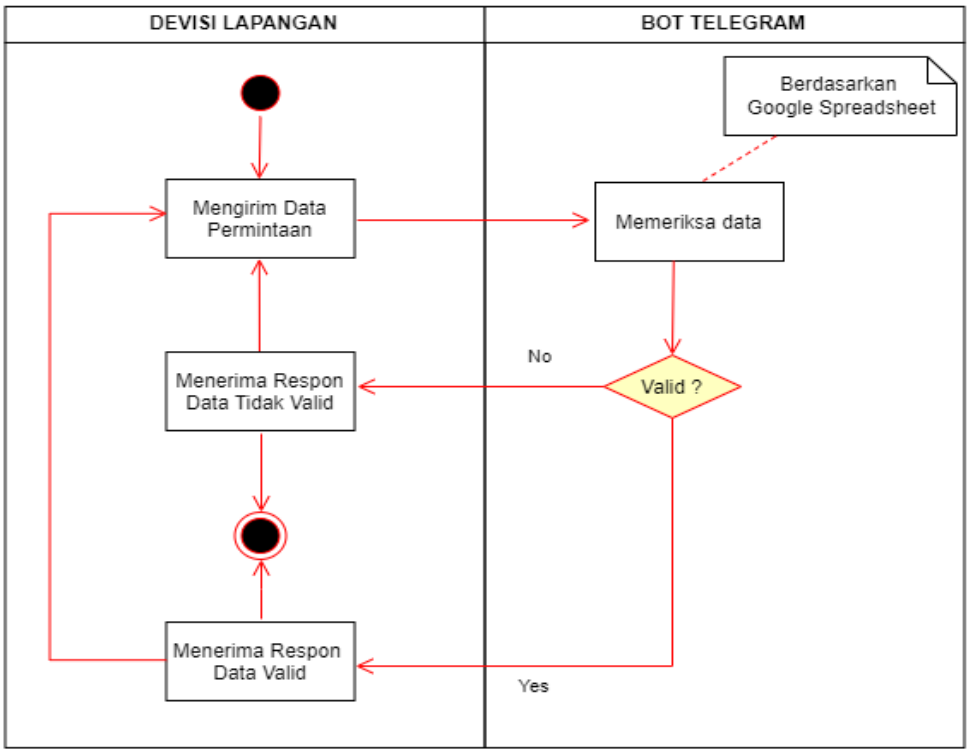
Gambar 4.1 *Activity Diagram* Sistem Lama

- Activity Diagram di mulai dari Devisi Lapangan mengirim data ke Devisi *Office*
- Devisi *Office* memeriksa data berdasarkan *database Google Spreadsheet* dan mengirimkan respon ke Devisi Lapangan, jika data tersebut valid maka respon yang dikirm ke Devisi lapangan berupa data yang diminta, jika data tersebut tidak valid maka respon yang di kirim adalah respon tidak valid atau data tidak ada di *database*

- Devisi Lapangan menerima respon dari Devisi *Office*.

2. Alur Sistem Baru

Alur sistem baru menggambarkan alur sistem yang diusulkan pada penelitian ini, berikut ini gambaran dari alur sistem yang baru.

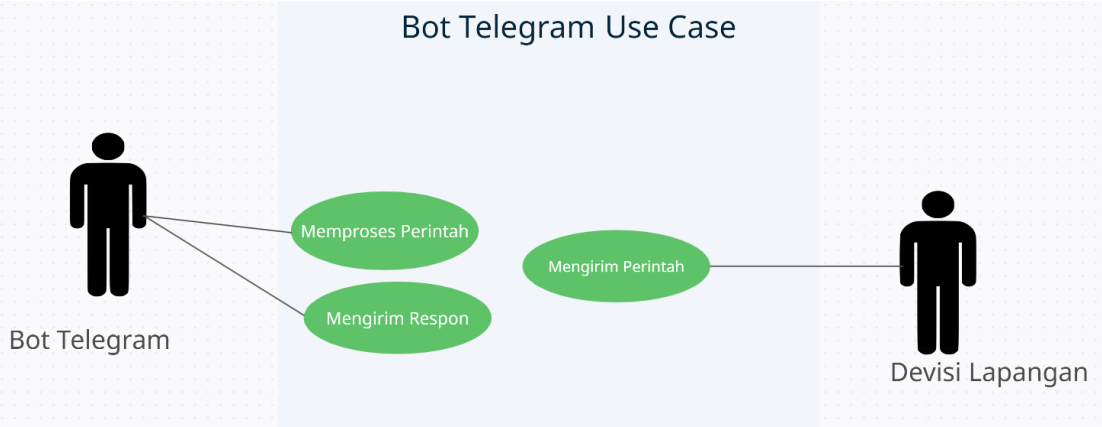


Gambar 4.2 *Activity Diagram* Sistem Baru

- Activity Diagram di mulai dari Devisi Lapangan mengirim data ke *Bot* Telegram
- Dengan Algoritma yang ditanam *Bot* Telegram memeriksa data berdasarkan *database Google Spreadsheet* dan mengirimkan respon ke Devisi Lapangan, jika data tersebut valid maka respon yang dikirm berupa data yang diminta, jika data tersebut tidak valid maka respon yang di kirim adalah respon tidak valid atau data tidak ada di *database*

- Devisi Lapangan menerima respon dari *Bot* Telegram.

Alur kerja sistem antara sistem lama dengan sistem baru tidak banyak perubahan atau bisa dikatakan sama, yang membedakan adalah entitas dari sistem tersebut yaitu Devisi *Office* di ganti dengan *Bot* Telegram. Berikut ini adalah *Use Case Diagram* dari sistem yang baru



Gambar 4.3 *Use Case Diagram* Sistem Baru

Pada Gambar 4.3 di perlihatkan *Use Case Diagram* dari sisem baru yang terdiri dari 2 aktor yaitu

- Devisi Lapangan melakukan pengiriman perintah dari telegram
- Bot Telegram melakukan pemrosesan terahadap data yang dikirim oleh Devisi lapangan dan melakukan pengiriman respon terhadap perintah tersebut

4.3. ALGORITMA SISTEM

Dalam penelitian ini terdapat algoritma yang ditanam di sistem. Algoritma ini di tanam pada *bot* telegram dan *google spreadsheet* yang akan di tampilkan sebagai berikut:

1. Algoritma Pada *Bot* Telegram

   Algoritma pada *bot* telegram berfungsi untuk mengirimkan perintah yang telah di masukkan ke *Google Spreadsheet*. Ini berupa link dan API yang di arahkan dari *bot* Telegram ke *Google Spreadsheet*.

2. Algoritma pada *Google Spreadsheet*

   Algoritma pada *google spreadsheet* dibagi menjadi beberapa kategori sebagai berikut:

   a. Algoritma pada *Google* Apps *Script*

      Google Apps *Script* adalah *platform* skrip yang dikembangkan oleh *Google* untuk pengembangan aplikasi ringan di *platform Google Workspace*. Bahasa skrip yang di gunakan adalah bahasa pemograman Java. Berikut ini gambaran dari algoritma yang ditanam pada *Google* Apps *Script*

      - Fungsi doPost(e)

        Fungsi ini dibuat untuk mengcapture, menganalisa dan membalas permintaan dari *user* melalui *bot*. Fungsi ini bisa dikatakan fungsi utama atau *Main Fuction*. Fungsi ini dapat dibagi menjadi beberapa formula IF sebagai berikut:

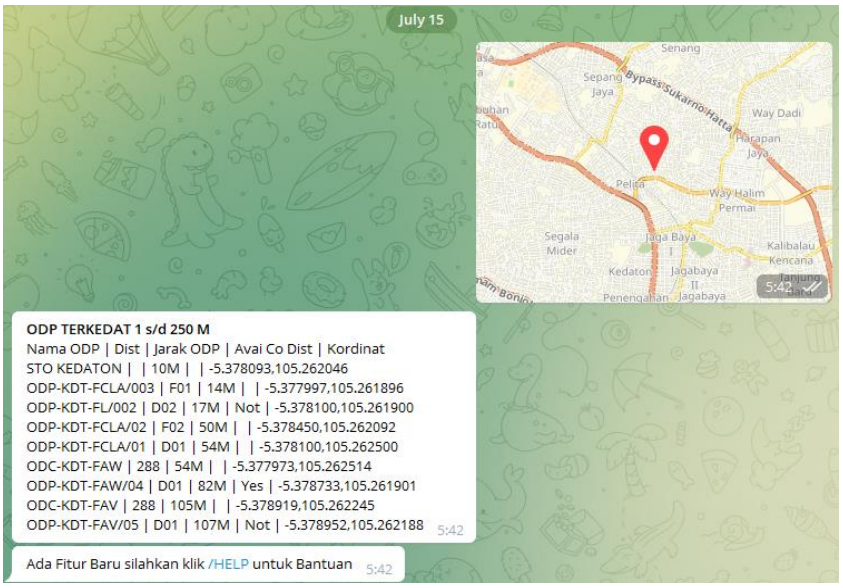
        1) Jika yang di input adalah titik kordinat dengan metode “Send My Current Location” dan tidak mengirim teks. Formula ini berguna untuk memfilter perintah yang khusus dengan metode tersebut, kemudian data di tangkap dan di arahkan ke Sheet Find Near pada cell B2, dengan algoritma yang sudah di tanam pada Sheet Find

Near akan mengkalkulasi data yang di input dan menampilkan data yang di inginkan jika memang data tersebut ada. Kemudian *user* yang mengirim permintaan akan di catat pada Sheet All Visitor. Berikut ini adalah *script* dari formula ini.

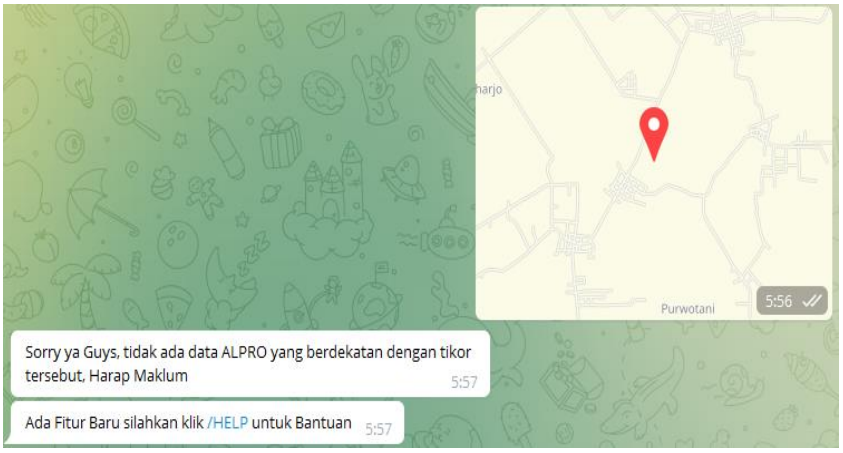
```
  if (updates.message.text==null) {
    var cordinate = geostring.latitude + "," + geostring.longitude;
    //kirm ke telegram
    var sheetx = SpreadsheetApp.openById(SheetID).getSheetByName('Find Near')
;


    //input to search
    sheetx.getRange(2, 2).setValue(cordinate);
    var xyxz = sheetx.getRange(2, 17).getValue();

    if (xyxz == "1") {
      sendText(updates.message.chat.id, notif);
      sendText(updates.message.chat.id, bantuan);
      }

    else if (xyxz >= 10) {
      //perulangan
      for (var row = 2; row <= 10; row++) {
        variables += sheetx.getRange(row, 11).getValue() + " | " + sheetx.
getRange(row, 13).getValue() + " | " + sheetx.getRange(row, 14).getValue().
toFixed(0) + "M | " + sheetx.getRange(row, 21).getValue() + " | " + sheetx.
getRange(row, 12).getValue() + "\n";
        }




      //Send Message
      sendText(updates.message.chat.id, label + variables);
      sendText(updates.message.chat.id, bantuan);

      //input to sheet Visitor
      var sheet2 = SpreadsheetApp.openById(SheetID).getSheetByName('All
Visitor');
      var last = sheet2.getRange('A1').getDataRegion().getLastRow();
      sheet2.getRange(last + 1, 1).setValue(new Date());
      sheet2.getRange(last + 1, 2).setValue(updates.message.chat.id);
      sheet2.getRange(last + 1, 3).setValue(updates.message.chat.first_name
+ ' ' + updates.message.chat.last_name);
      sheet2.getRange(last + 1, 4).setValue(updates.message.chat.username);
      sheet2.getRange(last + 1, 5).setValue(cordinate);
      }

    else {
```

Gambar 4.4 *Script* IF dengan metode *Send My Current Location*

Adapun contoh respon yang ditampilkan dari formula ini adalah sebagai berikut:



Gambar 4.5 Respon *valid* dengan metode *Send My Current Location*



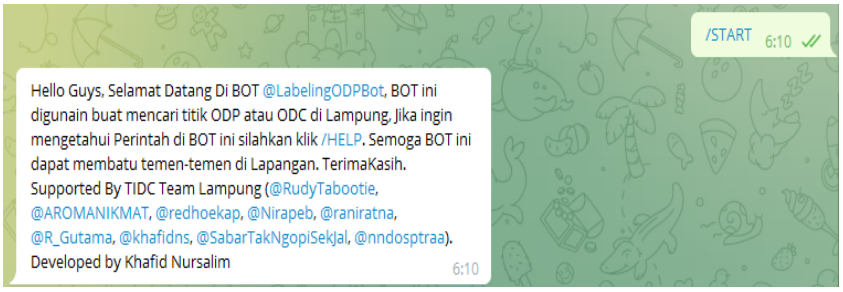
Gambar 4.6 Respon *not valid* dengan metode *Send My Current Location*

2) Jika yang di input adalah karakter /START, formula ini berguna untuk menampilkan deskripsi dari *Bot* Telegram, berikut ini adalah *script* yang terdapat pada formula ini

```
else if (updates.message.text.toUpperCase() == "/START") {
  //kirim ke bot telegram ketika ada perintah /start
  sendText(updates.message.chat.id, "Hello Guys, Selamat Datang Di BOT
@LabelingODPBot, BOT ini digunain buat mencari titik ODP atau ODC di Lampung,
Jika ingin mengetahui Perintah di BOT ini silahkan klik /HELP. Semoga BOT ini
dapat membatu temen-temen di Lapangan. TerimaKasih. Supported By TIDC Team
Lampung (@RudyTabootie, @AROMANIKMAT, @redhoekap, @Nirapeb, @raniratna,
@R_Gutama, @khafidns, @SabarTakNgopiSekJal, @nndosptraa). Developed by Khafid
Nursalim");
  }
```

Gambar 4.7 *Script* IF dengan karakter /START

Adapun contoh respon yang ditampilkan dari formula ini adalah sebagai berikut:



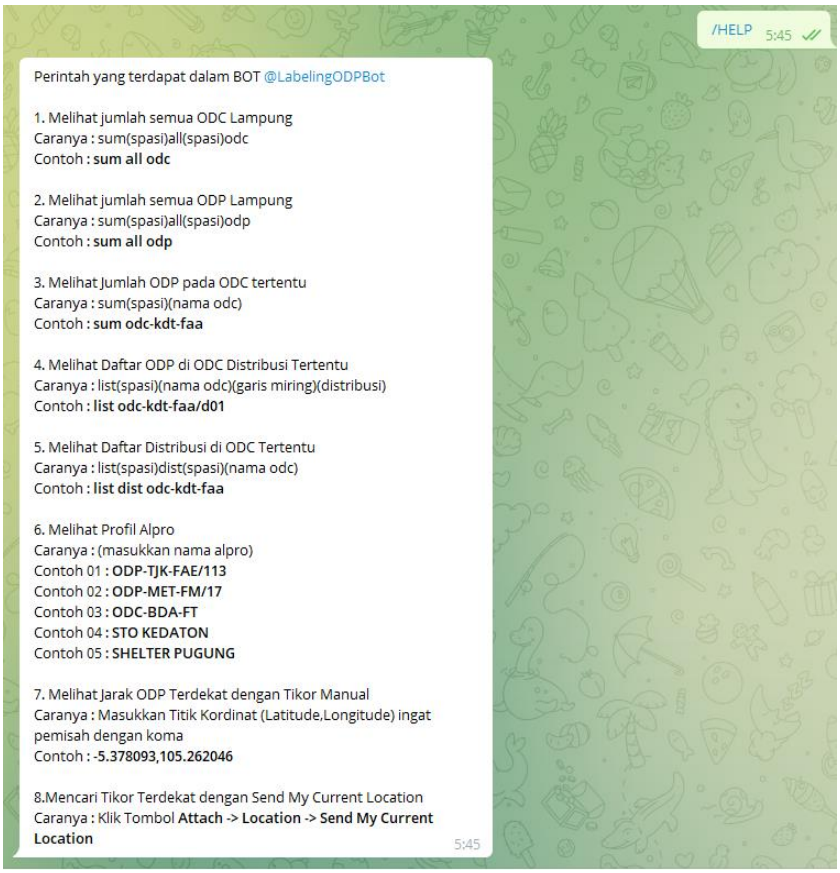
Gambar 4.8 Respon *Bot* dengan karakter /START

3) Jika yang di input adalah karakter /HELP, formula ini berguna untuk menampilkan prosedur penggunaan aplikasi. Berikut ini *script* yang terdapat dalam formula ini:

```
else if (updates.message.text.toUpperCase() == "/HELP") {
  //kirim ke bot telegram ketika ada perintah /start
  var helps = "Perintah yang terdapat dalam BOT @LabelingODPBot\n\n1. Melihat
jumlah semua ODC Lampung\nCaranya : sum(spasi)all(spasi)odc\nContoh : <b>sum all
odc</b>\n\n2. Melihat jumlah semua ODP Lampung\nCaranya : sum(spasi)all(spasi)
odp\nContoh : <b>sum all odp</b>\n\n3. Melihat Jumlah ODP pada ODC
tertentu\nCaranya : sum(spasi)(nama odc)\nContoh : <b>sum odc-kdt-faa</b>\n\n4.
Melihat Daftar ODP di ODC Distribusi Tertentu\nCaranya : list(spasi)(nama odc)
(garis miring)(distribusi)\nContoh : <b>list odc-kdt-faa/d01</b>\n\n5. Melihat
Daftar Distribusi di ODC Tertentu\nCaranya : list(spasi)dist(spasi)(nama odc)
\nContoh : <b>list dist odc-kdt-faa</b>\n\n6. Melihat Profil Alpro\nCaranya :
(masukkan nama alpro)\nContoh 01 : <b>ODP-TJK-FAE/113</b>\nContoh 02 :
<b>ODP-MET-FM/17</b>\nContoh 03 : <b>ODC-BDA-FT</b>\nContoh 04 : <b>STO KEDATON</
b>\nContoh 05 : <b>SHELTER PUGUNG</b>\n\n7. Melihat Jarak ODP Terdekat dengan
Tikor Manual\nCaranya : Masukkan Titik Kordinat (Latitude,Longitude) ingat
pemisah dengan koma\nContoh : <b>-5.378093,105.262046</b> \n\n8.Mencari Tikor
Terdekat dengan Send My Current Location\nCaranya : Klik Tombol <b>Attach ->
Location -> Send My Current Location</b>";
  sendText(updates.message.chat.id, helps)
}
```

Gambar 4.9 *Script* IF dengan karakter /HELP

Adapun contoh respon yang ditampilkan dari formula ini adalah sebagai berikut:



Gambar 4.10 Respon *Bot* dengan karakter /HELP

4) Jika yang di input adalah karakter SUM ALL ODP, formula ini berguna untuk menampilkan jumlah seluruh ODP yang tersimpan dalam *database*. Berikut ini *script* yang terdapat dalam formula ini:

```
else if (updates.message.text.toUpperCase() == "SUM ALL ODP") {
  //kirim ke bot telegram ketika ada perintah /start
  var sheetxx = SpreadsheetApp.openById(SheetID).getSheetByName('Jumlah ODP|ODC');
  var jml = "Jumlah ODP seluruh Lampung adalah "+sheetxx.getRange(1, 8).getValue();
  sendText(updates.message.chat.id, jml);
  sendText(updates.message.chat.id, bantuan);

  //input visitor
  var sheet2 = SpreadsheetApp.openById(SheetID).getSheetByName('All Visitor');
  var last = sheet2.getRange('A1').getDataRegion().getLastRow();
  sheet2.getRange(last + 1, 1).setValue(new Date());
  sheet2.getRange(last + 1, 2).setValue(updates.message.chat.id);
  sheet2.getRange(last + 1, 3).setValue(updates.message.chat.first_name + ' ' + updates.
message.chat.last_name);
  sheet2.getRange(last + 1, 4).setValue(updates.message.chat.username);
  sheet2.getRange(last + 1, 5).setValue(updates.message.text);

  }
```

Gambar 4.11 *Script* IF dengan karakter SUM ALL ODP

Adapun contoh respon yang ditampilkan dari formula ini adalah sebagai berikut:

Gambar 4.12 Respon *Bot* dengan karakter SUM ALL ODP

5) Jika yang di input adalah karakter SUM ALL ODC, formula ini berguna untuk menampilkan jumlah seluruh ODC yang tersimpan dalam *database*. Berikut ini *script* yang terdapat dalam formula ini:

```
else if (updates.message.text.toUpperCase() == "SUM ALL ODC") {
  //kirim ke bot telegram ketika ada perintah /start
  var sheetxx = SpreadsheetApp.openById(SheetID).getSheetByName('Jumlah ODP|ODC');
  var jml = "Jumlah ODC seluruh Lampung adalah "+sheetxx.getRange(1, 9).getValue();
  sendText(updates.message.chat.id, jml);
  sendText(updates.message.chat.id, bantuan);

  //input visitor
  var sheet2 = SpreadsheetApp.openById(SheetID).getSheetByName('All Visitor');
  var last = sheet2.getRange('A1').getDataRegion().getLastRow();
  sheet2.getRange(last + 1, 1).setValue(new Date());
  sheet2.getRange(last + 1, 2).setValue(updates.message.chat.id);
  sheet2.getRange(last + 1, 3).setValue(updates.message.chat.first_name + ' ' + updates.
message.chat.last_name);
  sheet2.getRange(last + 1, 4).setValue(updates.message.chat.username);
  sheet2.getRange(last + 1, 5).setValue(updates.message.text);

  }
```

Gambar 4.13 *Script* IF dengan karakter SUM ALL ODC

Adapun contoh respon yang ditampilkan dari formula ini adalah sebagai berikut:

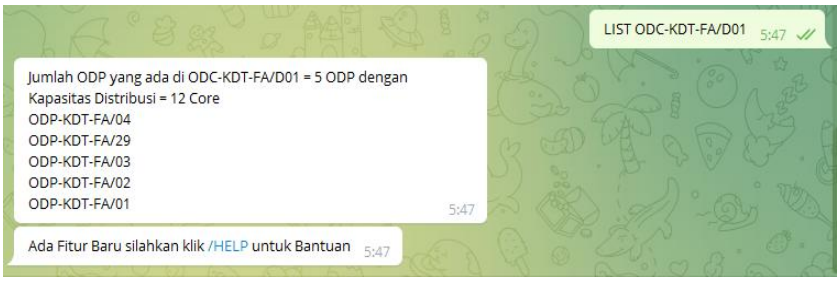
Gambar 4.14 Respon *Bot* dengan karakter SUM ALL ODC

6) Jika yang di input adalah karakter LIST ODC

```
else if (updates.message.text.toUpperCase().slice(0, 8) == "LIST ODC") {
  //kirim ke bot telegram ketika ada perintah ODC

  var sheetxx = SpreadsheetApp.openById(SheetID).getSheetByName('List ODP Dist');
  sheetxx.getRange(1, 2).setValue(updates.message.text.toUpperCase().slice(5, 20));
  var veri = sheetxx.getRange(1, 4).getValue();

  //input visitor
  var sheet2 = SpreadsheetApp.openById(SheetID).getSheetByName('All Visitor');
  var last = sheet2.getRange('A1').getDataRegion().getLastRow();
  sheet2.getRange(last + 1, 1).setValue(new Date());
  sheet2.getRange(last + 1, 2).setValue(updates.message.chat.id);
  sheet2.getRange(last + 1, 3).setValue(updates.message.chat.first_name + ' ' + updates.
message.chat.last_name);
  sheet2.getRange(last + 1, 4).setValue(updates.message.chat.username);
  sheet2.getRange(last + 1, 5).setValue(updates.message.text);

  if(veri == "2"){
    sendText(updates.message.chat.id, updates.message.text.toUpperCase().slice(5, 20)+"
Tidak ada di sistem" );
    sendText(updates.message.chat.id, bantuan);
  }
  else {
    var www = "Jumlah ODP yang ada di " + updates.message.text.toUpperCase().slice(5,
20) + " = " + sheetxx.getRange(1, 5).getValue()+" ODP dengan Kapasitas Distribusi = "+
sheetxx.getRange(1, 9).getValue()+" Core\n";
    for (var row = 3; row <= veri; row++) {
      variables += sheetxx.getRange(row, 1).getValue()+"\n";
      }
    sendText(updates.message.chat.id, www + variables);
    sendText(updates.message.chat.id, bantuan);
  }
}
```

Gambar 4.15 *Script* IF dengan karakter LIST ODC

Adapun contoh respon yang ditampilkan dari formula ini adalah sebagai berikut:



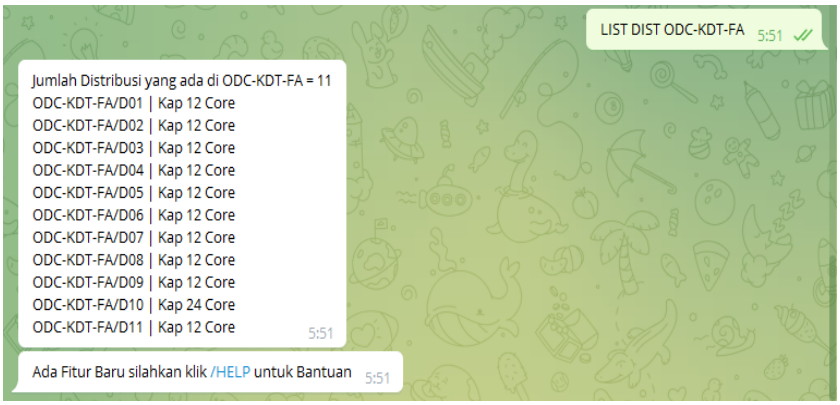
Gambar 4.16 Respon *Bot* dengan karakter LIST ODC

7) Jika yang di input adalah karakter LIST DIS

```
else if (updates.message.text.toUpperCase().slice(0, 8) == "LIST DIS") {
  //kirim ke bot telegram ketika ada perintah ODC

  var sheetxx = SpreadsheetApp.openById(SheetID).getSheetByName('Dist Search');
  sheetxx.getRange(1, 2).setValue(updates.message.text.toUpperCase().slice(10, 21)+"/");
  var veri = sheetxx.getRange(1, 4).getValue();

  //input visitor
  var sheet2 = SpreadsheetApp.openById(SheetID).getSheetByName('All Visitor');
  var last = sheet2.getRange('A1').getDataRegion().getLastRow();
  sheet2.getRange(last + 1, 1).setValue(new Date());
  sheet2.getRange(last + 1, 2).setValue(updates.message.chat.id);
  sheet2.getRange(last + 1, 3).setValue(updates.message.chat.first_name + ' ' + updates.
message.chat.last_name);
  sheet2.getRange(last + 1, 4).setValue(updates.message.chat.username);
  sheet2.getRange(last + 1, 5).setValue(updates.message.text);

  if(veri == "0"){
    sendText(updates.message.chat.id, updates.message.text.toUpperCase().slice(10, 21)+"
Tidak ada di sistem" );
    sendText(updates.message.chat.id, bantuan);
  }
  else {
    var www = "Jumlah Distribusi yang ada di " + updates.message.text.toUpperCase().slice
(10, 21) + " = " + sheetxx.getRange(1, 5).getValue()+"\n";
    for (var row = 2; row <= veri; row++) {
      variables += sheetxx.getRange(row, 1).getValue()+" | Kap "+ sheetxx.getRange(row,
2).getValue()+ " Core\n";
      }
    sendText(updates.message.chat.id, www + variables);
    sendText(updates.message.chat.id, bantuan);
  }
}
```

Gambar 4.17 *Script* IF dengan karakter LIST DIS

Adapun contoh respon yang ditampilkan dari formula ini adalah sebagai berikut:



Gambar 4.18 Respon *Bot* dengan karakter LIST DIS

- Fungsi getRows

  Fungsi ini dibuat untuk mengambil data dari *Google* Spereadsheet dimasukkan kedalam penyimpanan sementara guna untuk mengkalkulasi dan mendapatkan informasi tentang data-data yang ada.

```
function getRows() {
    //alocate range gsheet
    var rangeName = 'Sheet1!A2:Q';
    var rows = Sheets.Spreadsheets.Values.get(SheetID, rangeName).values;
    return rows;
}
```

Gambar 4.19 *Script* fungsi getRows

- Fungsi searchOdc

  Fungsi ini dibuat untuk mencari data ODC berdasarkan fungsi getRows yang telah mempunyai data yang di simpan dalam tempat penyimpanan sementara. Data yang diambil adalah data Titik Kordinat, Kapasitas ODC, Tgl Golive, Nama Project dan Type Project. Berikut ini *script* dari Fungsi searchOdc

```
function searchOdc(idOdp) {
    //variable find data with uppercase
    var data = idOdp.toUpperCase();
    var dataOdp = getRows();

    //find data
    for (var row = 0; row < dataOdp.length; row++) {
        if (dataOdp[row][0] == data) {
            kordinat = dataOdp[row][1];
            return "<u><b>PROFIL ODC</b></u>\n<b>Nama ODC : </b>" + dataOdp[row][0] + "\n" +
                "<b>Titik Kordinat : </b>" + dataOdp[row][1] + "\n" +
                "<b>Kapasitas ODC : </b>" + dataOdp[row][2] + "\n" +
                "<b>Tgl Golive : </b>" + dataOdp[row][3] + "\n" +
                "<b>Nama Project : </b>" + dataOdp[row][5] + "\n" +
                "<b>Type Project : </b>" + dataOdp[row][4] ;



        }
    }
    return "Sorry ya Guys, data gak ketemu, kemungkinan data belum di Input atau kalian
typo, silahkan cek lagi penulisannya, Terima Kasih";
}
```

Gambar 4.20 *Script* fungsi searchOdc

- Fungsi searchOdpById1

  Fungsi ini dibuat untuk mencari data ODP berdasarkan fungsi getRows yang telah mempunyai data yang di simpan dalam tempat penyimpanan sementara. Data yang diambil adalah data Nama ODP, Distribusi, Titik Kordinat, Tgl Golive, Nama Project Type Project, *Core* Distribusi, *Core Feeder*, IP GPON, Modul GPON, Port GPON, Lokasi GPON, Merek GPON, ID SW dan Mitra Pembangun

```
function searchOdpById1(idOdp) {
    //variable find data with uppercase
    var data = idOdp.toUpperCase();
    var dataOdp = getRows();

    //find data
    for (var row = 0; row < dataOdp.length; row++) {
        if (dataOdp[row][0] == data) {
            kordinat = dataOdp[row][1];
            return "<u><b>PROFIL ODP</b></u>\n<b>Nama ODP : </b>" + dataOdp[row][0] + "\n" +
                "<b>Distribusi : </b>" + dataOdp[row][2] + "\n" +
                "<b>Titik Kordinat : </b>" + dataOdp[row][1] + "\n" +
                "<b>Tgl Golive : </b>" + dataOdp[row][3] + "\n" +
                "<b>Nama Project : </b>" + dataOdp[row][5] + "\n" +
                "<b>Type Project : </b>" + dataOdp[row][4] + "\n" +
                "<b>Core Distribusi : </b>" + dataOdp[row][8] + "\n" +
                "<b>Core Feeder : </b>" + dataOdp[row][9] + "\n" +
                "<b>IP GPON : </b>" + dataOdp[row][10] + "\n" +
                "<b>Modul GPON : </b>" + dataOdp[row][11] + "\n" +
                "<b>Port GPON : </b>" + dataOdp[row][12] + "\n" +
                "<b>Lokasi GPON : </b>" + dataOdp[row][16] + "\n" +
                "<b>Merek GPON : </b>" + dataOdp[row][15] + "\n" +
                "<b>ID SW : </b>" + dataOdp[row][14] + "\n" +
                "<b>Mitra Pembangun : </b>" + dataOdp[row][13];


        }
    }
    return "Sorry ya Guys, data gak ketemu, kemungkinan data belum di Input atau kalian
typo, silahkan cek lagi penulisannya, Terima Kasih";
}
```

Gambar 4.21 *Script* fungsi searchOdpById1

- Fungsi SentText

  Fungsi ini dibuat sebagai fungsi untuk mengirimkan data ke *bot* Telegram

```
function sendText(chatid, text, replymarkup) {
    var data = {
        method: "post",
        payload: {
            method: "sendMessage",
            chat_id: String(chatid),
            text: text,
            parse_mode: "HTML",
            reply_markup: JSON.stringify(replymarkup)
        }
    };
    UrlFetchApp.fetch('https://api.telegram.org/bot' + token + '/', data);
}
```

Gambar 4.22 *Script* fungsi sendText

b. Algoritma pada Sheet Sheet1

Algoritma yang di tanam pada Sheet Sheet1 adala berupa formula excel yang bertujuan untuk mendapatkan informasi yang lebih berguna dengan inputan yang sedikit. Berikut ini formula yang ditanam

- Pada *field* CEK DUPLIKASI

```
=countif($A:$A,A2)
```

Field ini berfungsi untuk memeriksa karakter yang duplikasi berdasarkan *field* ODP/ODC/STO. Jika terdapat duplikasi *field* ini akan menghitung jumlah kata yang berulang, jika tidak terdapat duplikasi *field* ini akan menunjukkan angka 1.

- Pada *field* STO

```
=if(A2="","", MID(A2,FIND("-",A2) +
1,FIND(CHAR(160),SUBSTITUTE(A2,"-",CHAR(160),2)) - 1 -
(FIND("-",A2))))
```

Field ini berfungsi untuk menampilkan STO yang terdapat pada *field* ODP/ODC/STO dengan memeriksa terlebih dahulu isi dari *field* ODP/ODC/STO, jika tidak terdapat isi dalam filed, *field* ini tidak menunjukkan isi, jika terdapat isi *field* ini akan meneruskan algoritmanya, yaitu mengambil karakter tengah yang diapit oleh karakter “-“

- Pada *field* MEREK OLT

```
=IF(K2="","", VLOOKUP(K2,'Info Olt'!$A$2:$D$113,4,0))
```

Field ini berfungsi untuk menampilkan merek olt berdasarkan *field* IP yang disinkronkan dengan sheet Info Olt.

- Pada *field* POWER OLT

```
=IF(K2="","",VLOOKUP(K2,'Info Olt'!$A$2:$B$113,2,FALSE))
```

Field ini berfungsi untuk menampilkan STO sebagai sumber dari odp berdasarkan *field* IP yang disinkronkan dengan sheet Info Olt.

- Pada *field* FORMULASI

```
=if(mid(A2,11,1)="/","ODC-
"&MID(A2,5,6)&"/"&C2,if(mid(A2,12,1)="/","ODC-
"&MID(A2,5,7)&"/"&C2,""))
```

Field ini berfungsi untuk menampilkan catuan ODC dan Distribusi berdasarkan *field* ODP/ODC/STO, jika tidak terdapat isi dalam filed, *field* ini tidak menunjukkan isi, jika terdapat isi *field* ini akan meneruskan algoritmanya, yaitu menambah karakter “ODC-“ dan mengambil karakter setelah karakter “/” dan ditambah dengan karakter yang ada di *field* “DIS”

- Pada *field* LATITUDE & LONGITUDE.

```
=IF(B2="","", SPLIT(TRIM(B2),","))
```

Field ini berfungsi untuk menampilkan dan memisahkan kordinat berdasarkan *field* “TIKOR” dengan tanda koma “,” sebagai pemisah.

- Pada *field* CATUAN ODC

```
=if(mid(A2,11,1)="/","ODC-
"&MID(A2,5,6),if(mid(A2,12,1)="/","ODC-"&MID(A2,5,7),""))
```

Field ini berfungsi untuk menampilkan ODC sebagai sumber ODP dengan menambahkan karakter “ODC-“ dan mengambil karakter setelah karakter “-“ pertama pada *field* ODP/ODC/STO.

c. Algoritma pada Sheet All Visitor

Algoritma yang di tanam pada Sheet All Visitor ini bertujuan untuk melihat jumlah pengguna keseluruhan dari *bot* telegram yang terdapat pada penelitian ini. Berikut ini formula yang di tanam

- Pada seluruh *field* VERIFY

```
=IF(LEFT(E2,1)="-",1, countif(Sheet1!$A:$A,E2))
```

Field ini berfungsi untuk memeriksa perintah yang dapat di layani oleh sistem, jika perintah tersebut dapat di layani maka akan menampilkan 1, tetapi jika tidak dapat di layani akan menampilkan 0.

- Pada *field* TOTAL REQUEST

```
=COUNTA(E:E)-1
```

Field ini berfungsi untuk menghitung seluruh permintaan yang di terima oleh *bot* dengan menghitung *field* “*User*name” yang terisi.

- Pada *field* SUCCESS REQUEST

```
=SUM(G:G)
```

Field ini berfungsi untuk mehitung jumlah permintaan yang dapat di layani dengan menjumlahkan *field* “Verify”

- Pada *field* TOTAL VISITOR

```
=COUNTUNIQUE(D:D)
```

Field ini berfungsi untuk menghitung jumlah pengguna yang mengakses *bot* ini. Rumus dari *field* ini adalah menghitung pengguna yang unik pada *field* "*User*name"

- Pada *field* NO, VISITOR, COUNT

  ```
  =SORT(M2:N336,N2:N336,false)
  ```

  Field ini berfungsi untuk melihat *user* yang sering menggunakan *bot*. rumus dari *field* ini adalah mengurutkan *user*name berdasarkan jumlah permintaan terbanyak.

d. Algoritma pada Sheet One Day History

Algoritma yang di tanam pada Sheet One Day History ini betujuan untuk melihat jumlah pengguna per hari yang melakukan permintaan pada *Bot* telegram. Berikut ini formula yang di tanam

- Pada *Field* TODAY

  ```
  =TODAY()
  ```

  Field ini berfungsi untuk menampilkan tanggal hari ini.

- Filed yang lain

  Untuk *field* yang lain berfungsi sama dengan Algoritma yang terdapat pada Sheet All Visitor, perbedaaanya terdapat pada perhitungan. Jika pada Sheet All Visitor perhitungannya adalah total keseluruhan sedangkan pada Sheet One Day History perhitungannya adalah perhari.

e. Algoritma pada Sheet Find Near

Algorima yang ditanam pada Sheet Find Near tergolong kompleks karena terdapat beberapa algoritma. Algoritma yang ditaman pada Sheet ini bertujuan untuk melihat jarak ODP atau alpro (alat produksi) terdekat dari titik kordinat yang di masukkan. Berikut ini formula yang di tanam

- Pada *field* ODP, KORDINAT, DISTRIBUSI

```
=FILTER(Sheet1!A:C,(Sheet1!T:T>=C5)*(Sheet1!T:T<=C4),(Sheet1!U
:U>=D5)*(Sheet1!U:U<=D4))
```

Field ini berfungsi untuk menampilkan daftar yang mengandung kordinat berdasarkan *field* input max dan *field* input min

- Pada *field* DISTANCE

```
=IF(I2="","", 2 * 6371000 * ASIN(SQRT((SIN((I2*(3.14159/180)-
$C$2*(3.14159/180))/2))^2+COS(I2*(3.14159/180))*COS($C$2*(3.14
159/180))*SIN(((J2*(3.14159/180)-$D$2*(3.14159/180))/2))^2)))
```

Field ini berfungsi untuk mengkalkulasi jarak antara tikor yang dimasukkan dengan list yang ditampilkan

- *Field* ODP (SORT)

```
=if(F2="","", sort(E2:H, H2:H, TRUE))
```

Field ini berfungsi untuk mengurutkan berdasarkan jarak terdekat

f. Algoritma pada Sheet Skema ODP

Algorima yang ditanam pada Sheet Find Skema ODP tergolong kompleks karena terdapat beberapa algoritma. Algoritma yang ditaman pada Sheet ini bertujuan untuk melihat layout dari suatu distribusi

tertentu berikut kapasiatas distribusi dan jumlah odp yang terdapat dalamnya. Berikut ini formula yang di tanam

- Pada *field* CORE FEE USED

  ```
  =IF(B2="","", ROUNDUP(SUMIF($L$2:$N$1864,B2,$M$2:$M$1864)/4,0))
  ```

  Field ini berfungsi untuk menghitung jumlah *core* yang digunakan dalam suatu *feeder* berdasarkan pebulatan dari kalkulasi *field CORE* USED dibagi menjadi 4

- Pada *field* CORE FEE IDLE

  ```
  =IF(C2="","",C2-(D2+F2))
  ```

  Field ini berfungsi untuk menghitung jumlah *core* yang tersedia dalam suatu *feeder* berdasarkan kalkulasi *field* KAP FEE diambil dengan *field CORE* FEE USED

- Pada *field* KET FEEDER

  ```
  =IF(E2="","",IF(E2=0,"HABIS",IF(E2<=3,"KRITIS","IDLE")))
  ```

  Field ini berfungsi untuk melihat status apakah suatu *feeder* masih dapat digunakan atau dapat di pakai untuk pembangunan, parameter yang ada berupa keterangan HABIS jika *core* habis, KRITIS jika *core* tersisa kurang dari 3 dan IDLE jika *core* lebih dari 3.

- Pada *field* NAMA DISTRIBUSI

  ```
  =IF(B2="","","DS-"&MID(A2,5,3)&"-FE-"&MID(B2,8,10)&"/"&RIGHT
  (A2,2))
  ```

  Field ini berfungsi untuk memberi nama pada suatu distribusi tertentu berdasarkan *field* NAMA *FEEDER* dan *field* ODC DAN DIST

- Pada *field* CORE USED

```
=COUNTA(T2:AQ2)
```

Field ini berfungsi untuk mengkalkulasi jumlah ODP yang terdapat pada Distribusi tertentu

- Pada *field* CORE IDLE

```
=IF(K2="","",K2-(M2+O2))
```

Field ini berfungsi untuk mengkalkulasi jumlah *core* distribusi yang tersisa dengan berdasarkan *field CORE* USED diambil *field* KAP DIST

- Pada *field* KET DIS

```
=IF(N2="","",IF(N2=0,"HABIS",IF(N2<=3,"KRITIS","IDLE")))
```

Field ini berfungsi untuk melihat status apakah suatu distribusi masih dapat digunakan atau dapat di pakai untuk pembangunan, parameter yang ada berupa keterangan HABIS jika *core* habis, KRITIS jika *core* tersisa kurang dari 3 dan IDLE jika *core* lebih dari 3.

- Pada *field* ODP1, ODP2 dst

```
=TRANSPOSE(filter(Sheet1!S:S,Sheet1!R:R=A2))
```

Field ini berfungsi untuk menampilkan daftar ODP pada Distribusi Tertentu secara Horizontal berdasarkan *database* yang telah di Input pada Sheet Sheet1

g. Algoritma pada Sheet Jumlah ODP|ODC

Algoritma yang di tanam pada Sheet Jumlah ODP|ODC ini betujuan untuk melihat jumlah ODP yang terdapat pada ODC tertentu dan untuk

melihat jumlah keseluruhan ODP yang terdapat dalam *database*. Berikut ini formula yang di tanam

- Pada *field* CATUAN ODC

  ```
  =unique(filter(Sheet1!Z2:Z,Sheet1!Z2:Z<>""))
  ```

  Field ini berfungsi untuk menampilkan daftar ODC berdasarkan Sheet Sheet1

- Pada *field* JUMLAH ODP

  ```
  =IF(A2="","", COUNTIF(Sheet1!Z:Z,A2))
  ```

  Field ini berfungsi untuk menampilkan jumlah ODC berdasarkan *field* CATUAN ODC

- Pada cell H1

  ```
  =sum(B:B)
  ```

  Cell ini berfungsi untuk mengkalkulasi jumlah keseluruhan odp yang terdapat dalam *database*

h. Algoritma pada Sheet List ODP Dist

Algoritma yang di tanam pada Sheet List ODP Dist ini betujuan untuk melihat daftar ODP yang terdapat pada distribusi tertentu. Berikut ini formula yang di tanam

- Pada *field* NAMA ODP

  ```
  =IFERROR(if(B1="","", FILTER(Sheet1!A:A,Sheet1!R:R=B1)),"")
  ```

  Field ini berfungsi untuk menampilkan daftar ODP tertentu berdasarakan keyword yang terdapat pada cell B2 yang menyesuaikan dengan Sheet Sheet1

- Pada *field* CORE IDLE

  ```
  =VLOOKUP(B1,'Skema ODP'!$A:$N,14,FALSE)
  ```

  Field ini berfungsi untuk menampilkan *core* idle berdasarkan keyword yang terdapat pada cell B2 yang menyesuaikan dengan Sheet Skema ODP

- Pada *field* CORE BROKEN

  ```
  =VLOOKUP(B1,'Skema ODP'!$A:$O,15,FALSE)
  ```

  Field ini berfungsi untuk menampilkan *core* broken berdasarkan keyword yang terdapat pada cell B2 yang menyesuaikan dengan Sheet Skema ODP

- Pada *field* CAP CORE

  ```
  =VLOOKUP(B1,'Skema ODP'!$A:$K,11,FALSE)
  ```

  Field ini berfungsi untuk menampilkan kapasitas *core* berdasarkan keyword yang terdapat pada cell B2 yang menyesuaikan dengan Sheet Skema ODP

4.4. ANALISIS PERBANDINGAN SISTEM LAMA DENGAN SISTEM BARU

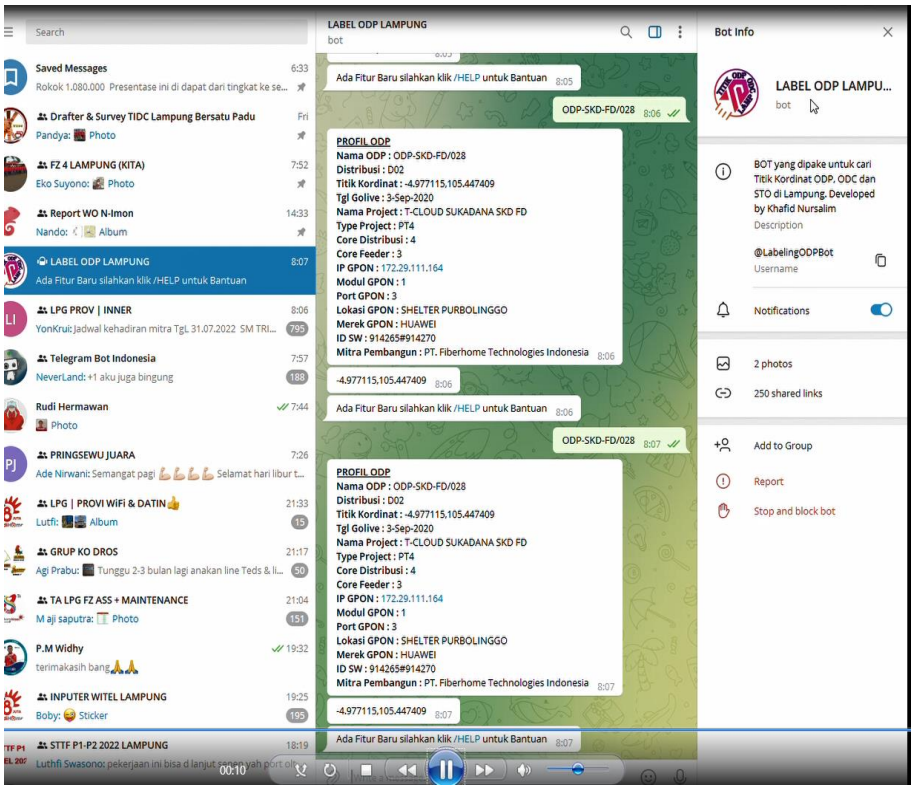
Analisis perbandingan sistem ini bertujuan untuk melihat tingkat kemanfaatan dari sistem baru dengan sistem lama. Adapun perbandingan sistem lama dengan sistem yang baru dapat dilihat pada tabel berikut:

Tabel 4.1 Perbandingan Sistem

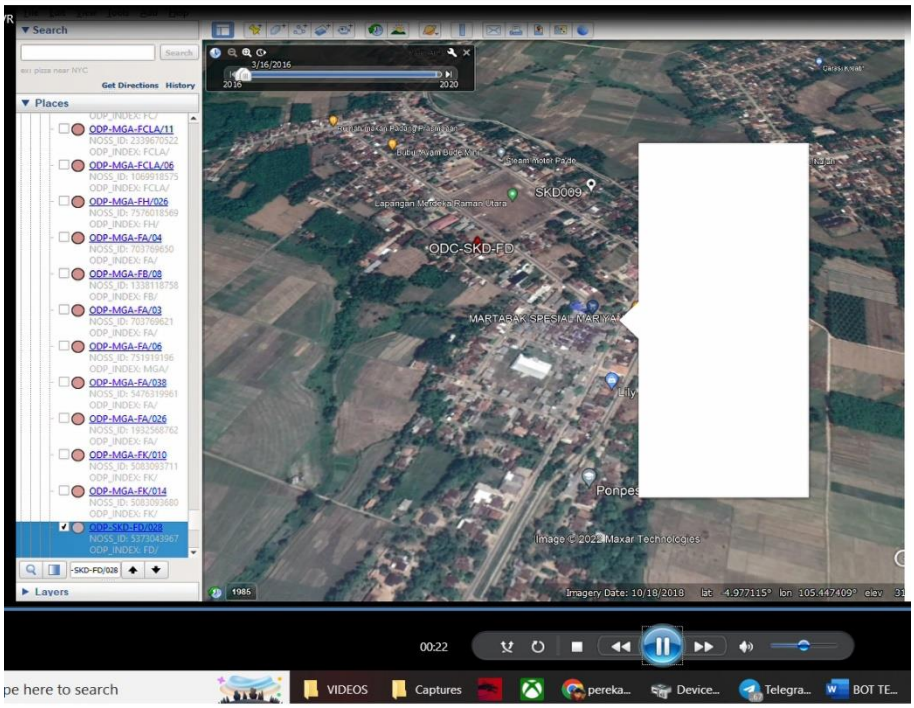
NO	INDIKATOR	SISTEM LAMA	SISTEM BARU
1	Lama Waktu Respon	22 detik	10 detik
2	Waktu *Standby*	8 Jam	24 Jam
3	Kelengkapan Data	50 %	75 %
4	Sensitifitas Terahadap Perintah	60 %	90%
5	Otomatisasi Sistem	100 %	50%

1. Lama Waktu Respon

   Pada indikator ini dilakukan percobaan yang dapat di simulasikan yaitu dengan melihat respon sistem terhadap perintah. Sistem lama memiliki respon 23 detik sedangkan sistem baru memiliki respon 10 detik. Sehingga untuk indikator ini sistem baru lebih unggul dibandingkan sistem yang lama dimana semakin kecil waktu respon akan semakin baik.



Gambar 4.23 Waktu Respon Sistem Baru



Gambar 4.24 Waktu Respon Sistem Lama

2. Waktu *Standby*

   Indikator ini menandakan waktu sistem untuk dapat *standby* atau idle dalam satu hari. Pada sistem lama waktu *standby* dilihat dari jam kerjan karyawan yaitu dari jam 08.00 s/d 17.00 dipotong waktu istirahat selama 1 jam, sedangkan untuk sistem yang baru waktu *standby* sistem adalah 24 jam, sehingga pada indikator ini sistem baru lebih unggul dibandingkan sistem yang lama dimana waktu *standby* sistem yang lebih lama akan semakin baik.

3. Kelengkapan Data

   Indikator ini mengindikasikan kelengkapan data sistem yang ditampilkan ke pada *user*. Kelengkapan data sistem lama sangat terbatas yaitu 50% dilihat dari informasi yang di tampilkan dalam sistem. Adapun data yang di tampilkan berupa nama odp, index odp, dan tikor odp. Sedangkan dalam sistem yang baru data yang di tampilkan berupa nama odp, distribusi, kordinat, tanggal golive, type project, *core* distribusi, *core feeder*, ip gpon, modul gpon, port gpon, lokasi gpon, merek gpon, ID SW, dan Mitra Pembangun.

4. Sensitifitas Terahadap Perintah

   Indikator ini menandakan tingkat kesensitifitasan sistem terhadap perintah atau order yang dimasukkan. Pada sistem yang lama kesensitifitasannya dapat ditolerir sebesar 60% karena yang membaca dan menterjemahkan adalah manusia sebagai contoh permintaan dari *user* tidak harus standar yang

penting dapat di logika dengan manusia sedangkan tingkat ke sensitifan dari sistem baru sebesar 90 % didapat dari permintaan dari *user* harus standar tetapi sistem dapat mentolerir huruf capital dan non capital. Sehingga untuk indikator ini sistem lama lebih unggul dibandingkan sistem yang baru dimana semakin kecil presentase indikator ini akan semakin baik.

5. Otomatisasi sistem

   Indikator ini menandakan tingkat keterlibatan manusia dalam sistem. Pada sistem lama campur tangan manusia sangat dominan sekitar 100 % dibandingkan dengan sistem yang baru, sistem yang baru campur tangan manusia sekitar 50% karena dibantu dengan penggunaan algoritma yang di tanam pada sistem