**Data Warehouse (B)**

Pertukaran *File* *Bot to Bot* pada Aplikasi Telegram dengan Menggunakan Fitur *Channel*



I GUSTI BAGUS NGURAH SATYA WIBAWA

NIM. 1404505071

Asistensi pada tanggal

20 Desember 2016

TEKNOLOGI INFORMASI

FAKULTAS TEKNIK

UNIVERSITAS UDAYANA

2016

**Pertukaran File Bot to Bot pada Aplikasi Telegram dengan Menggunakan Fitur Channel**

1. Aplikasi Telegram

Telegram adalah aplikasi bertukar pesan yang memungkinkan pengguna untuk mengirimkan pesan pesan rahasia yang dienkripsi *end-to-end* sebagai keamanan tambahan. Dengan Telegram Anda juga dapat berbagi lebih dari sekedar gambar dan vidio, tapi Telegram juga memungkinkan Anda mentransfer dokumen atau mengirim lokasi Anda saat ini ke teman dengan mudah. Telegram merupakan aplikasi terbaik dari semua, cepat, ringan, tidak ada iklan dan benar-benar gratis. (sumber: www.centerklik.com)

Telegram mempunyai *user* *interface* yang bersih dan mempunyai berbagai fitur, dan karena itu Telegram sangat banyak digunakan orang di seluruh dunia saat ini. Tidak seperti aplikasi *chat* lainnya, Telegram adalah berbasis *cloud*, yang berarti Anda dapat dengan mulus memindahkan percakapan Anda antara *smartphone*, *tablet*, *web* dan bahkan di *desktop* Anda. (sumber: www.centerklik.com)

Salah satu kelebihan dari aplikasi ini, Telegram membebaskan para pengembang aplikasi untuk membuat aplikasi Bot. Bot adalah program komputer yang melakukan pekerjaan tertentu secara otomatis. Telegram menyediakan 2 jenis Bot yaitu Bot API dan Bot MTProto. Bot API menggunakan tatap muka HTTP *request*, sedangkaan Bot MTProto menggunakan tatap muka protokol yang dibuat langsung oleh Telegram pada sistem operasi Unix.

Pengembangan Bot kali ini, penulis menggunakan Bot API dengan dengan bahasa pemrograman Python yang sudah ditambahkan librarypyTelegramBotAPI.

1. Program Pertukaran *File Bot to Bot*

Permasalah yang harus di selesaikan penulis adalah mengirim *file* dari satu Bot ke Bot lainnya. Namun hal yang menjadi kendala bagi penulis adalah kebijakan dari Telegram yang tidak menyediakan komunikasi antar Bot pada API Bot. Untuk itu, penulis membuat cara baru bertukar data dengan memanfaatkan fasilitas *channel* dari Telegram. Fitur *channel* hampir mirip dengan fitur yang dimiliki BlackBerry Messenger. Meski berbeda tipis dengan *group*, *channel* dapat difungsikan untuk kegunaan yang lebih luas, misalnya untuk brand atau perusahaan. Di sana mereka dapat memperoleh anggota yang tak terbatas, menyebarkan informasi terbaru dan menjangkau anggota dengan cepat dan gratis.

Langkah pertama adalah membuat Bot pada Bot Father Telegram. Bot yang dibuat berjumlah 2 Bot dengan nama *username* Bot masing-masing adalah KiBarakBot (kibarak\_bot) dan KiBarak2Bot (BarakBot). Saat membuat Bot, penulis diberi *token* untuk bisa menggunakan Bot tersebut yang di *generate* langsung oleh Telegram.

|  |
| --- |
|  |

**Gambar 1** Pembuatan 2 Bot

Selanjutnya dibuat *channel* baru untuk uji coba dengan nama PaumBot dan kemudian memasukan kedua Bot yang telah dibuat tadi sebagai *administrator* pada *channel*.

|  |
| --- |
|  |

**Gambar 2** Administrator pada PaumBot

Setelah penulis mendapatkan 2 *token* dari 2 Bot yang telah dibuat dan sudah menjadikan Bot tersebut menjadi *administrator* pada *channel* PaumBot, penulis memasukkan 2 token Bot secara terpisah kedalam aplikasi pertukaran file dengan nama bot\_file.py pada direktori *server* A dan direktori *server* B yang sudah dibuat sebelumnya.

|  |
| --- |
| import mysql.connector as mc  import requests  import telebot  TOKEN = ‘<TOKEN ANDA DISINI>’  bot = telebot.TeleBot(TOKEN)  database = {"host": "<HOST>", "user": "<USER DB>", "password": "<PASSWORD DB>", "db": "<NAMA DB>"}  def isExist(update\_id):  conn = mc.MySQLConnection(\*\*database)  c = conn.cursor()  query = '''SELECT \* FROM `in` WHERE `update\_id`='%s' ''' % update\_id  c.execute(query)  count = 0  for row in c.fetchall(): count += 1  if (count > 0):  return True  else:  return False  def sendFileMsg(chat\_id, fname):  doc = open(fname, 'rb')  send = bot.send\_document(chat\_id, doc)  print(send)  def sendMsg(chat\_id, msg):  send = bot.send\_message(chat\_id, msg)  print(send)  def replyMsg():  conn = mc.MySQLConnection(\*\*database)  c = conn.cursor()  query = '''SELECT \* FROM `out` WHERE sttus ='ready' '''  c.execute(query)  for row in c.fetchall():  sendMsg(row[1], row[2])  query = '''UPDATE `out` SET sttus='sent' WHERE id = %i'''  % row[0]  c.execute(query)  conn.commit()  def downloadFile():  conn = mc.MySQLConnection(\*\*database)  c = conn.cursor()  query = "SELECT \* FROM `in` WHERE `attacment`='true' AND  `download`='false'"  c.execute(query)  for row in c.fetchall():  file\_info = bot.get\_file(row[5])  r = requests.get('https://api.telegram.org/file/bot{0}/{1}'.format(TOKEN, file\_info.file\_path))  with open('download/' + row[6], 'wb') as f:  for chunk in r.iter\_content(chunk\_size=1024):  if chunk:  f.write(chunk)  query = '''UPDATE `in` SET `downloaded`='true' WHERE id  = %i''' % row[0]  c.execute(query)  conn.commit()  def receiveMsg():  conn = mc.MySQLConnection(\*\*database)  c = conn.cursor()  msgs = bot.get\_updates()  for msg in msgs:  chat\_text = ""  file\_id = ""  file\_name = ""  mime\_type = ""  channel\_post = msg.channel\_post  if (channel\_post != None):  update\_id = msg.update\_id  doc = channel\_post.document  chat\_id = channel\_post.chat.id  if (doc == None):  attachment = 'false'  chat\_text = channel\_post.text  else:  attachment = 'true'  file\_id = doc.file\_id  file\_name = doc.file\_name  mime\_type = doc.mime\_type  if (not isExist(update\_id)):  query = '''INSERT INTO `in`(  update\_id, chat\_id, chat\_text,  attachment, file\_id, file\_name,  mime\_type)  VALUES (%i,%i,'%s','%s','%s','%s','%s')  ''' % (  update\_id, chat\_id, chat\_text,  attachment, file\_id, file\_name,  mime\_type)  c.execute(query)  conn.commit() |

**Kode Program 1** Program bot\_file.py

Kode Program 1 menunjukan pada bagian atas program bot\_file.py terdapat sintaks untuk mengimpor *library* yang dibutuhkan seperti *telebot*, *request* dan *time*. Pada bagian tersebut tersebut juga terdapat inisialisasi variabel *token* yang digunakan untuk membuat objek Bot, dan pengaturan *database* yang akan digunakan. Program ini terdiri dari 6 fungsi yang akan dipanggil dari program Python yang berbeda.

**Tabel 1** Penjelasan fungsi pada bot\_file.py

|  |  |  |  |
| --- | --- | --- | --- |
| No | Nama Fungsi | Kegunaan | Dipanggil pada |
| 1 | isExist(update\_id) | Untuk mengecek apakah data dengan file\_id yang dimasukan pada basis data sudah ada atau tidak. | Fungsi receiveMsg() |
| 2 | sendFileMsg(chat\_id, fname) | Untuk mengirim file yang namanya dimasukan kedalam fungsi dan ada berada pada folder yang sama. | Program bot\_file\_send.py |
| 3 | sendMsg(chat\_id, msg) | Untuk mengirim pesan tertulis. | Program bot\_file\_send.py,  Fungsi replyMsg() |
| 4 | receiveMsg() | Untuk menerima pesan masuk yang berisi konten file atau teks yang kemudian dimasukan kedalam database. | Program bot\_file\_receive.py |
| 5 | downloadFile() | Untuk mendownload file yang datanya sudah dimasukan kedalam basis data. | Program bot\_file\_download.py |
| 6 | replyMsg() | Untuk mengirim pesan balasan berdasarkan pesan yang dikirimkan oleh pengguna yang lain. | Program bot\_file\_reply.py |

Fungsi-fungsi yang telah dijelaskan pada Tabel 1 akan dipanggil dan digunakan pada program Python yang berbeda, gunanya untuk mempercepat proses sesuai dengan kegunaan yang dibutuhkan.

|  |
| --- |
| from bot\_file import sendFileMsg, sendMsg  chat\_id = -1001060483879  #1. fungsi untuk mengirim file  sendFileMsg(chat\_id, 'bot\_file.py')  #2. Fungsi untuk mengirim pesan teks  sendMsg(chat\_id, '/count') |

**Kode Program 2** Program bot\_file\_send.py

Program bot\_fille\_send.py menjalankan fungsi sendFileMsg untuk mengrim file dan fungsi sendMsg untuk mengirim teks. Jika ingin mengirim *file* saja, maka aktifkan fungsi sendFileMsg dan memberikan tanda komentar (tanda #) pada fungsi sendMsg. Begitu juga sebaliknya.

|  |
| --- |
| import time  from bot\_file import receiveMsg  while(True):  receiveMsg()  time.sleep(2) |

**Kode Program 3** Program bot\_file\_receive.py

Program bot\_file\_receive.py menjalankan receiveMsg yang berfungsi untuk memasukan data-data yang didapat dari pesan yang dikirim pada *channel*. Fungsi receiveMsg akan mem-*polling* secara terus menerus dengan menggunakan *looping*.

|  |
| --- |
| import time  from bot\_file import downloadFile  while(True):  downloadFile()  time.sleep(2) |

**Kode Program 4** Program bot\_file\_download.py

Program bot\_file\_download.py menjalankan fungsi downloadFile untuk mengunduh *file* yang datanya ada dalam *database* dengan status belum di unduh.

|  |
| --- |
| import time  from bot\_file import replyMsg  while(True):  replyMsg()  time.sleep(2) |

**Kode Program 5** Program bot\_file\_reply.py

Pesan dengan sintaks tertentu dalam *database* memicu *triger* yang sudah dibuat untuk menulis pesan balasan yang dimasukan kedalam tabel out. Pesan balasan ini nantinya akan dikirim oleh fungsi replyMsg yang dijalankan pada program bot\_file\_reply.py.

|  |
| --- |
| DELIMITER $$  USE `telegram\_msg`$$  DROP TRIGGER /\*!50032 IF EXISTS \*/ `msg\_handler`$$  CREATE  /\*!50017 DEFINER = 'root'@'localhost' \*/  TRIGGER `msg\_handler` AFTER INSERT ON `in`  FOR EACH ROW BEGIN  DECLARE v\_count INT;  IF(new.chat\_text = "/count") THEN  SET v\_count := 0;  SELECT COUNT(id) INTO v\_count FROM `in` WHERE attachment='true';  INSERT INTO `out`(chat\_id,chat\_text) VALUES(new.chat\_id,CONCAT("You have ", CONVERT(v\_count, CHAR)," attachment message(s) in database"));  END IF;    END;  $$  DELIMITER ; |

**Kode Program 6** Trigger msg\_handler

Triger msg\_handler akan berjalan ketika ada pesan teks yang masuk dengan nilai ‘/count’. Setelah mendapat pemicu tersebut, *triger* akan mencari jumlah pesan yang memiliki *attachment* yang sudah tercatat di dalam *database*. Setelah mendapat nilai jumlah tersebut, *triger* akan membuat pesan balasan yang dimasukan ke dalam tabel out. Pesan ini nantinya dikirim oleh program bot\_file\_reply.py.

1. Uji Coba Program

Untuk mengetahui kesuaian program dengan fungsinya, diperlukan suatu uji coba. Uji coba mengambil kasus *server* B mengirim *file* ke *server* A dan menjalankan sintaks ‘/count’. Program bot\_file\_receive.py, bot\_file\_download.py, bot\_file\_reply.py dijalankan pada *server* A seperti yang ditunjukan pada Gambar 3.

|  |
| --- |
|  |

**Gambar 3** Menjalankan program untuk menerima pesan dan file

Gambar 3 memperlihatkan IDE Pycharm sudah menjalankan program bot\_file\_receive.py, bot\_file\_download.py, bot\_file\_reply.py pada server A.

|  |
| --- |
| (a) |
| (b) |

**Gambar 4** Isi pada tabel in dan out

Gambar 4 memperlihatkan isi dari tabel in dan out pada *database* server A masih kosong.

|  |
| --- |
|  |

**Gambar 5** Server B mengirimkan pesan

Server B mengirimkan pesan ‘Oi’, pesan ‘/count’ kemudian dilanjutkan dengan mengirimkan file dengan nama ‘bot\_file.py’, dan mengirimkan lagi pesan ‘/count’. Hasilnya ditunjukan pada Gambar 6.

|  |
| --- |
|  |

**Gambar 6** Tampilan pada percakapan kedua bot

Gambar 6 tidak terlihat jelas siapa yang menjadi pengirim pesan. Hal ini diakibatkan oleh kebijakan Telegram yang menyelaraskan nama dari administrator dengan nama Channel menjadi sama. Terlihat pada Gambar 6, balasan dari server A adalah ‘You have 0 attachment message(s) in database’ dan ‘You have 1 attachment message(s) in database’. Terjadi perubahan jumlah attachment message pada database.

|  |
| --- |
|  |

**Gambar 7** Data pada tabel in

Gambar 7 menunjukan pesan yang masuk kedalam database. Terlihat ada 1 attachment dile yang tercatat pada database dengan nama ‘bot\_file.py’.

|  |
| --- |
|  |

**Gambar 8** Data pada tabel out

Dikarenakan server B mengirim pesan ‘/count’ sebanyak 2 kali, maka pada tabel out terdapat 2 pesan yang dibuat.

|  |
| --- |
|  |

**Gambar 9** File yang dikirim telah terunduh

Gambar 9 menunjukan file bot\_file.py sudah terunduh dan disimpan di dalam folder download pada server A.