

**MULTI CHANNEL ACCESS  
DIALOGFLOW**



OLEH:

I Wayan Yoga Sukrasena (1705551011)

Tanggal Asistensi	Keterangan
3 Oktober 2019	Membuat Chatbot PATEN di Telegram
9 Oktober 2019	Membuat Chatbot PATEN di Telegram

**PROGRAM STUDI TEKNOLOGI INFORMASI  
FAKULTAS TEKNIK  
UNIVERSITAS UDAYANA  
2019**

## DAFTAR ISI

DAFTAR ISI.....	2
BAB I PENDAHULUAN.....	3
1.1    Sumber Jurnal.....	3
1.2    Latar Belakang .....	3
1.1    Tujuan Penulisan .....	3
1.2    Batasan Penulisan.....	3
BAB II LANDASAN TEORI.....	5
2.1    Chatbot .....	5
2.2    Dialogflow.....	5
2.3    Algoritma.....	8
2.4    Telegram.....	8
2.5    Webhook .....	9
2.6    Python.....	9
2.7    Heroku .....	10
BAB III PEMBAHASAN JURNAL.....	12
BAB IV PEMBUATAN CHATBOT.....	15
4.1    Rancangan Chatbot.....	15
4.1.1    Pembuatan Agent .....	21
4.1.2    Pembuatan Intent.....	22
4.1.3    Pembuatan Entitas .....	26
4.2    Tahap Fulfillment .....	28
4.2.1    Pembuatan Data pada MySQL.....	28
4.2.2    Pembuatan Fulfillment .....	29
4.2.3    Percobaan Bot .....	37
BAB V PENUTUP.....	46
5.1    Kesimpulan.....	46
5.2    Saran .....	46

# **BAB I**

## **PENDAHULUAN**

### **1.1 Sumber Jurnal**

Penulisan laporan Multi Channel Access yang berkaitan dengan Dialogflow ini menggunakan Jurnal sebagai objek pembahasan dan implementasi dari sistem Chatbot. Berikut merupakan data sumber jurnal yang digunakan.

Judul Jurnal : Ontology based Chatbot (For E-commerce Website)

Penulis : Anusha Vegesna , Pranjal Jain, Dhruv Porwal

### **1.2 Latar Belakang**

Chatbot adalah layanan obrolan virtual dengan kecerdasan buatan atau *Artificial Intelligence* (AI) yang dapat digunakan untuk mensimulasikan percakapan manusia baik secara teks ataupun suara. Chatbot bisa dibangun sesuai dengan topik yang sudah dimodelkan dalam basis pengetahuan. Chatbot menjadi lebih populer di grup bisnis saat ini karena Chatbot dapat mengurangi biaya layanan pelanggan dan menangani beberapa pengguna sekaligus, namun untuk menyelesaikan banyak tugas, diperlukan Chatbot yang memiliki sifat efisien. Mengatasi hal tersebut, maka dibuat sebuah Chatbot yang efisien dan menghemat waktu dalam memesan makanan dan *booking* tempat pada sebuah restoran yang dibangun menggunakan Dialogflow.

### **1.1 Tujuan Penulisan**

Tujuan dari penulisan laporan Dialogflow adalah sebagai berikut.

- a. Untuk mengetahui apa yang dimaksud dengan *Chatbot*
- b. Untuk mengetahui integrasi yang dimiliki oleh Dialogflow dalam pembuatan *chatbot*.
- c. Mengkaji penggunaan Dialogflow pada *chatbot* dari jurnal terkait.

### **1.2 Batasan Penulisan**

Batasan dari penulisan laporan Dialogflow adalah sebagai berikut.

- a. Implementasi yang dilakukan hanya pada penggunaan *Fulfillment* dan *Integration*.

## **BAB II**

### **LANDASAN TEORI**

#### **2.1 Chatbot**

Chatbot adalah layanan obrolan virtual dengan kecerdasan buatan atau *Artificial Intelligence* (AI) yang dapat digunakan untuk mensimulasikan percakapan manusia baik secara teks ataupun suara. Chatbot bisa dibangun sesuai dengan topik yang sudah dimodelkan dalam basis pengetahuan.

Chatbot terdiri dari tiga kombinasi, di antaranya adalah: Pertama yaitu *user interface*, dimana ini sendiri adalah jembatan antara Chatbot dan *user* dalam berinteraksi. Selanjutnya *Artificial Intelligence*, AI inilah yang membuat Chatbot mengerti dan memahami setiap interaksi yang terjadi dengan *user*. Ketiga adalah integrasi, integrasi yang dimaksud ini adalah kemampuan untuk menggabungkan dengan sistem lainnya dimana akan menambah kekayaan fitur atau informasi tambahan yang terdapat di dalam suatu Chatbot.

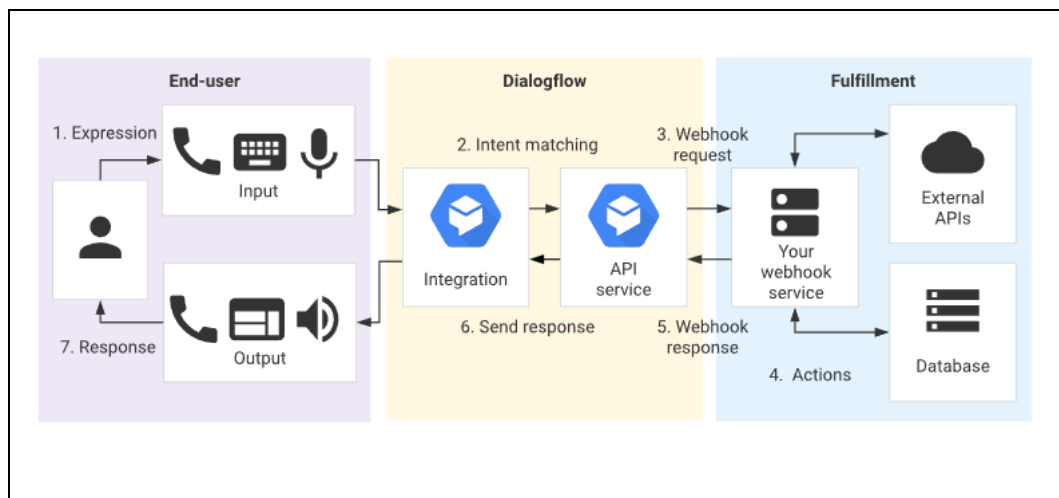
#### **2.2 Dialogflow**

API.AI yang dulunya adalah Speaktioit adalah pengembang teknologi interaksi computer dan manusia. Fokus teknologi yang digunakan adalah dalam bidang NLP (*Natural Language Processing*). Perusahaan ini telah mengembangkan beberapa produk yang memiliki fitur utama melakukan tugas tanya jawab dengan bahasa alami antara manusia dan komputer. Bulan September 2014 Speaktioit memperkenalkan sebuah *platform* yang diberi nama API.AI. Platform inilah yang banyak digunakan oleh pengembang pihak ketiga untuk membuat sebuah *chatbot*, selain itu API.AI menyediakan kepada pengembang fitur-fitur lainnya yaitu sebuah SDK (*Software Development Kit*) yang memiliki fungsi dapat menambahkan pengenalan suara, pemahaman bahasa alami dan konversi *text-to-speech* untuk aplikasi Android, dan IOS. API.AI juga memiliki halaman *website* yang digunakan untuk melakukan pengujian skenario yang dapat digunakan para *developer* untuk menguji skenario yang telah dirancang sebelumnya. Bulan September tahun 2016

Google membeli perusahaan ini dan memberikan nama Dialogflow. Google menggunakan layanan ini untuk asisten *virtual* Google.

Dialogflow merupakan *platform* yang menyediakan NLP (*Natural Language Processing*) dan NLU (*Natural Language Understanding*) dalam pembuatan Chatbot, dimana dengan adanya hal tersebut dapat memudahkan memfasilitasi atau penanganan percakapan manusia, dengan kata lain untuk melakukan interaksi antara manusia dengan sistem komputer yang berbasis pada percakapan teks maupun suara.

NLP merupakan kemampuan sebuah sistem NLP dalam memiliki pengetahuan bahasa alami baik dari susunan kalimat, arti dari kata tersebut dan maksud dari sebuah kalimat. NLU adalah merupakan sub bidang dari NLP, dimana fokus tujuan dari NLU adalah untuk melakukan pemahaman terhadap suatu kalimat dan melakukan analisis semantik.



**Gambar 2.1** Cara kerja platform Dialogflow (API.AI)

Gambar 2.1 merupakan cara kerja di dari *platform* Dialogflow. Pertama adalah *user* yang mengirimkan *request* berupa teks ataupun suara ke dalam *platform* Dialogflow, kemudian permintaan tersebut akan diproses di dalam *Intent* untuk memetakan permintaan tersebut dan tindakan apa yang harus dilakukan. Selanjutnya apabila terdapat informasi tambahan yang akan diberikan sebagai respon, *platform* ini akan mengirimkan *fulfillment*, ini akan mendapatkan informasi

dari sumber daya luar. Selanjutnya *user* akan mendapatkan jawaban dari pertanyaan tersebut. Pengertian mengenai komponen-komponen penting pada Dialogflow adalah seperti berikut.

- a. *Agents* mempunyai tugas untuk mengelola seluruh alur percakapan, di dalam *Agents* terdapat sekumpulan *Intents*, dan *Entities*.
- b. *Intents* merupakan sebuah tempat untuk memetakan pertanyaan apa yang dikirim oleh pengguna dan tindakan apa yang harus dilakukan. Tujuan dari *Intents* adalah mendefinisikan tata bahasa percakapan dan tugas apa yang harus dilakukan saat pengguna menggunakan frasa tertentu.
- c. *Entities* adalah sebuah alat yang sangat kuat untuk mengidentifikasi nilai yang berguna dari sebuah masukan bahasa alami dalam hal ini adalah pertanyaan yang dikirim oleh user. Dialogflow memiliki 3 jenis *Entities*, yaitu *System Entities*, *Developer Entities*, dan *User Entities*. *System Entities* merupakan entitas yang sudah dibenamkan di dalam sistem, artinya entitas tersebut sudah tersedia, *Developer Entities* merupakan entitas yang dibuat dan didefinisikan oleh developer, *User Entities* adalah yang didefinisikan untuk setiap *user* pada setiap permintaan.
- d. *Context* mewakili keadaan dari permintaan pengguna. Dengan menggunakan *contexts*, Dialogflow dapat mengontrol alur percakapan dengan pengguna. Mengkonfigurasi *contexts* diidentifikasi dengan nama *string*. Ketika suatu *intent* cocok, setiap *contexts output* yang dikonfigurasi untuk *intent* itu menjadi aktif. Saat *contexts* apa pun aktif, Dialogflow hanya akan cocok dengan *intents* yang dikonfigurasi dengan *contexts input* yang cocok dengan *contexts* yang sedang aktif.
- e. *Events* memungkinkan untuk melakukan *request intent* berdasarkan sesuatu yang telah dikomunikasikan oleh pengguna. Dialogflow mendukung *events* dari beberapa *platform* berdasarkan tindakan yang dilakukan pengguna pada *platform* tersebut.
- f. *Fulfillment* adalah kode yang digunakan sebagai *webhook* yang memungkinkan *Agent Dialogflow* memanggil logika bisnis berdasarkan *intent-by-intent*. Selama percakapan, *fulfillment* memungkinkan untuk

menggunakan informasi yang diekstraksi oleh NLP *Dialogflow* untuk menghasilkan respon dinamis atau memicu tindakan di *back-end*.

### 2.3 Algoritma

Algoritma yang digunakan pada pembuatan Chatbot ini adalah Fuzzy String Matching. Fuzzy secara bahasa dapat diartikan samar, dengan kata lain logika Fuzzy adalah logika yang samar, dimana pada logika Fuzzy suatu nilai dapat bernilai 'true' dan 'false' secara bersamaan. Tingkat 'true' atau 'false' nilai dalam logika Fuzzy tergantung pada bobot keanggotaan yang dimilikinya. Logika Fuzzy memiliki derajat keanggotaan rentang antara 0 hingga 1, berbeda dengan logika digital yang hanya memiliki dua keanggotaan 0 atau 1 saja pada satu waktu. Logika Fuzzy sering digunakan untuk mengekspresikan suatu nilai yang diterjemahkan dalam bahasa (linguistic), semisal untuk mengekspresikan suhu dalam ruangan apakah ruangan tersebut dingin, hangat, atau panas.

Fuzzy String Matching adalah salah satu metode pencarian string yang menggunakan proses pendekatan terhadap pola dari string yang dicari. Melakukan pencarian terhadap string yang sama dan juga string yang mendekati dengan string lain yang terkumpul dalam sebuah penampung atau kamus. Kunci dari konsep pencarian ini adalah bagaimana memutuskan bahwa sebuah string yang dicari memiliki kesamaan dengan string tertampung di kamus, meskipun tidak sama persis dalam susunan karakternya. Untuk memutuskan “kesamaan” ini dipergunakan sebuah fungsi yang diistilahkan sebagai *similarity function*. Fungsi ini akan bertugas memutuskan string hasil pencarian jika ditemukan string hasil pendekatan (aproksimasi).

### 2.4 Telegram

Telegram adalah sebuah aplikasi layanan pengirim pesan instan multiplatform berbasis awan yang bersifat gratis dan nirlaba. Klien Telegram tersedia untuk perangkat telepon seluler (Android, iOS, Windows Phone, Ubuntu Touch) dan sistem perangkat komputer (Windows, OS X, Linux). Para pengguna dapat mengirim pesan dan bertukar foto, video, stiker, audio, dan tipe berkas



lainnya. Telegram juga menyediakan pengiriman pesan ujung ke ujung terenkripsi opsional.

Telegram dikembangkan oleh Telegram Messenger LLP dan didukung oleh wirausahawan Rusia Pavel Durov. Kode pihak kliennya berupa perangkat lunak sistem terbuka namun mengandung blob binari, dan kode sumber untuk versi terbaru tidak selalu segera dipublikasikan, sedangkan kode sisi servernya bersumber tertutup dan berpaten. Layanan ini juga menyediakan API kepada pengembang independen. Pada Februari 2016, Telegram menyatakan bahwa mereka memiliki 100 juta pengguna aktif bulanan, mengirimkan 15 miliar pesan per hari.

Keamanan Telegram telah menghadapi pemeriksaan teliti yang menjadi perhatian; para kritikus mengklaim bahwa model keamanan Telegram dirusak oleh penggunaan protokol enkripsi yang dirancang khusus yang belum terbukti andal dan aman, dan dengan tidak mengaktifkan percakapan aman secara default. Telegram juga menghadapi kritik karena penggunaan skala luas oleh organisasi teroris Negara Islam (NIIS). NIIS telah merekomendasikan Telegram kepada para pendukung dan anggotanya dan pada Oktober 2015 mereka mampu melipatgandakan jumlah pengikut saluran resmi mereka menjadi 9.000 orang.

## **2.5 Webhook**

Webhook merupakan HTTP *callback* yang menggunakan konsep API, Webhook akan bekerja apabila terdapat suatu kejadian yang datang dari luar dan fungsi dari Webhook adalah menerima data atau informasi secara *real-time* (*link* URL ditambahkan agar data yang dikirim dapat langsung diterima). Kelebihan dari penggunaan Webhook untuk pengembangan aplikasi adalah dalam hal integrasi antara satu aplikasi satu dengan yang lainnya.

## **2.6 Python**

Python adalah bahasa pemrograman interpretatif multiguna dengan filosofi perancangan yang berfokus pada tingkat keterbacaan kode. Bahasa pemrograman Python disebut sebagai bahasa yang kemampuan, menggabungkan kapabilitas, dan

sintaksis kode yang sangat jelas, dan juga dilengkapi dengan fungsionalitas pustaka standar yang besar serta komprehensif.

Python adalah sebuah bahasa pemrograman scripting tingkat tinggi atau high-level, interpreted, interactive, dan object-oriented. Python dengan desain yang sangat mudah di baca dan dipahami, karena sama seperti bahasa pemrograman yang lainnya yaitu dengan menggunakan kata bahasa inggris. Selain itu juga lebih sedikit dalam penggunaan rumus atau syntac.

1. Interpreted: Python diproses pada saat runtime oleh interpreter, artinya Anda tidak perlu untuk mengkompilasi program Anda sebelum dijalankan. Sama seperti di bahasa pemrograman PHP dan PERL
2. Interactive: Maksudnya Anda dapat secara langsung berinteraksi dan menafsirkan scripting menggunakan Prompt Python pada saat menulis program Anda
3. Object-oriented: Python juga mendukung sistem object-oriented atau teknik pemrograman yang merangkum kode dalam objek.

Python dibuat dan dikembangkan lebih lanjut oleh Guido Van Rossum, yaitu seorang programmer yang berasal dari Negara Belanda. Dibuat dan dikembangkan di kota Amsterdam, Belanda pada tahun 1990. Pada tahun 1995 Python dikembangkan lagi agar lebih kompatibel oleh Guido Van Rossum.

Kemudian di awal tahun 2000 versi Python dikembangkan dan diperbaharui lagi sehingga kini Bahasa Pemrograman Python telah mencapai Versi 3. Sebenarnya awal mula dari kata “Python” Diambil oleh Guido Van Rossum dari sebuah acara televisi yang lumayan terkenal yang bernama Mothy Python Flying Circus, Yaitu sebuah acara sirkus yang disukai oleh Guido Van Rossum.

## **2.7 Heroku**

Heroku adalah sebuah *cloud platform* atau tempat penyimpanan yang menjalankan bahasa pemrograman tertentu. Heroku juga termasuk ke dalam kriteria jenis *Platform As A Service* atau yang disingkat menjadi PaaS, yang dimana fungsi PaaS ini ialah *user* yang ingin melakukan penyebaran atau *deploy* aplikasi proyeknya ke Heroku cukup dengan melakukan konfigurasi pada aplikasi proyek

yang ingin di *deploy* dan juga sudah terdapat *platform* untuk memungkinkan pengguna menjalankan, mengembangkan, dan bahkan mengelola aplikasi tanpa kompleksitas membangun serta memelihara infrastruktur data yang sudah terkait ke dalam pengembangan dan peluncuran aplikasi proyek.

### BAB III

#### PEMBAHASAN JURNAL

Sistem yang diusulkan adalah *chat-bot* berbasis Ontologi yang sebagian besar didasarkan pada domain *E-commerce*. API situs web *E-commerce* (situs web eBay yang tersedia secara bebas) digunakan sebagai sumber data. *Template* ontologi dibangun menggunakan *platform* WEBPROTEGE yang dimiliki oleh Stanford.edu memiliki basis pemrograman berbasis objek yang mengambil data dari sumber data (menggunakan aturan Jape). Setelah data sepenuhnya diambil dalam *template* maka pengguna dapat mengajukan pertanyaan ke bot dan sesuai pertanyaan, maksud dan properti lainnya akan diatur menggunakan NLP di Wit.ai. Menggunakan maksud dan properti lain yang diatur, panggilan fungsi akan dibuat menggunakan bahasa Python seperti panggilan API ke *template* Ontology dan permintaan akan dijawab oleh chatbot dari data yang diambil dalam *template* Ontology.

DialogFlow pada Chatbot ini bekerja dengan menggunakan *Natural Language Processing* (NLP), jadi apa pun yang pengguna tulis atau katakan (*Training phrase*) akan dilanjutkan ke *Intent*, dimana *Intent* yang akan memetakan apa yang dikatakan pengguna untuk tindakan tertentu yang harus diambil. *Intent* dapat dipicu dengan *Event*, yang berasal dari sumber selain pengguna, seperti API eksternal.

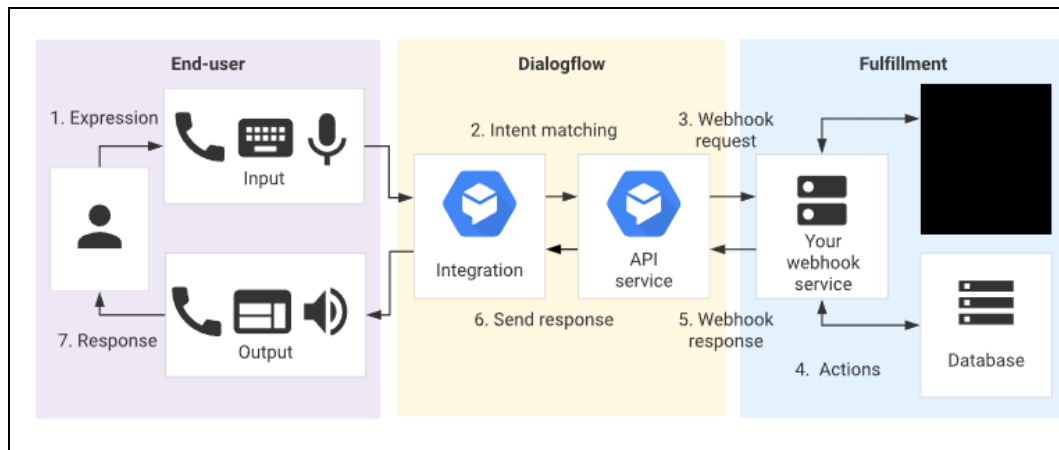
*Context* dapat menugaskan *Intents* untuk menentukan konteks atau topik percakapan, dan dapat dikombinasikan dengan Parameter untuk menyimpan informasi yang diminta pengguna. *Context* dan Parameter dapat diakses dengan aplikasi *backend*, dimana ini seperti *Action* yang dapat digunakan untuk menghasilkan respon tertentu (mengirim *e-mail* atau menyimpan data ke basis data).

Dialogflow memerlukan aplikasi *backend* untuk untuk mengumpulkan informasi pengguna, seperti nama pengguna (nama yang mereka ingin Chatbot untuk memanggil mereka), jenis kelamin dan usia yang berupa kisaran.

Komponen-komponen pada Dialogflow yang digunakan dalam Chatbot ini seperti:

- a. *Action* dan *Parameters*: Komponen yang digunakan untuk mendefinisikan suatu kata kedalam suatu aksi yang berisi *entity* dan nilai parameter dari *Actions*.
- b. *Agent*: Modul Dialogflow menggunakan NLU (*Natural Language Understanding*), ini merupakan pengatur aliran percakapan.
- c. *Context*: Komponen untuk menyimpan informasi dan percakapan sebelumnya untuk dilanjutkan ke percakapan selanjutnya.
- d. *Entity*: Komponen yang digunakan untuk mengambil nilai parameter dari input yang dikirim oleh pengguna.
- e. *Event*: Digunakan untuk memicu Intents dari sumber eksternal.
- f. *Intent*: Komponen yang digunakan memetakan *Training phrase* dan *Action* untuk dilakukan sebuah tindakan atau fitur yang digunakan untuk mengenali apa yang pengguna inginkan atau katakan dan aksi apa yang seharusnya diambil.
- g. *Response*: Kalimat yang akan diterima oleh pengguna berdasarkan kalimat yang dikatakan atau ditulis oleh pengguna.
- h. *Training phrase*: Merupakan tempat untuk menginputkan kata maupun kalimat yang akan dikatakan oleh pengguna, walau tidak ada dalam *list*, *Machine Learning* akan mendeteksi kalimat pengguna dan mencocokkan dari beberapa kalimat contoh yang telah dibuat sebelumnya.

Respon untuk pengguna dapat dicapai dengan sebuah *Intent* tertentu (yang sudah ditentukan sebelumnya dalam *Agent*) atau dengan mengirim pesan langsung dari aplikasi *backend* menggunakan API Telegram.



**Gambar 3.1** Cara kerja Chatbot

Gambar 3.1 merupakan rancangan atau alur dari Chatbot yang terintegrasi dengan beberapa komponen lainnya seperti layanan pesan Telegram dan layanan Dialogflow, dimana Chatbot ini di jembatani oleh Webhook agar dapat terhubung dengan sistem pengelolaan pesanan Telegram. Komponen-komponen yang digunakan di dalam Chatbot ini di antaranya seperti *web services*, *database*, Webhook, Messenger (Telegram), layanan Dialogflow. *Database* ini akan di *hosting* dan dapat diakses secara *online*, untuk dapat mengelola data yang terdapat di dalam *database* maka dibutuhkan sebuah antarmuka yang dapat melakukan tugas-tugas pengelolaan data yang ada di dalam *database*.

Cara kerja dari Chatbot ini, pertama adalah *user* mengirimkan pertanyaan berupa teks ke dalam *platform* pesan teks Messenger, kemudian pertanyaan tersebut akan diolah di *platform* NLP atau Dialogflow melalui Webhook agar dapat saling berinteraksi. Pertanyaan tersebut kemudian diolah pada Dialogflow untuk dicari tahu dan diidentifikasi maksud dari pertanyaan tersebut, apabila pertanyaan tersebut merupakan pertanyaan untuk melakukan “Self Report” maka Dialogflow akan memberi identifikasi bahwa maksud dari pertanyaan tersebut adalah “Self Report”, dan Dialogflow akan mengirimkan informasi kembali ke Webhook untuk diolah lebih lanjut. Webhook juga akan berkomunikasi dengan Heroku untuk mendapatkan data yang dibutuhkan, dan ini akan mendapatkan informasi dari sumber daya luar sehingga *user* akan mendapatkan jawaban dari pertanyaan tersebut.

## BAB IV

### PEMBUATAN CHATBOT

#### 4.1 Rancangan Chatbot

Pembuatan Chatbot ini mencakup pembuatan rancangan yang terdiri dari komunikasi pada *Intent* yaitu masukkan dari *user* dan respon yang akan dikirimkan oleh bot, serta *conversational flow* atau alur percakapan dari bot yang akan dibuat, dimana komunikasi tersebut dibuat kedalam bentuk tabel.

**Tabel 4.1** Tabel Intent dan Respon

No.	Intent	Training Phase	Respon
1.	Menu	pelayanan yang tersedia, menu yang tersedia, pelayanan menu, pelayanan,menu	Halo username, Silahkan pilih menu di bawah
2.	Menu.pengajuan.dagang	usaha perdagangan	Silahkan pilih layanan di Bawah
3.	Pengajuan_reklame	Pengajuan reklame	Silahkan pilih layanan di Bawah
4.	Pengajuan_reklame_form	Form reklame	Masukan nama anda
5.	Pengajuan_reklame_form-namauser	yoga	Masukan nama kegiatan
6.	Pengajuan_reklame_form-namaacara	itcc	Masukan Tanggal Acara (contoh : 12-12-

			2019 sampai 13-12-2019)
7.	Pengajuan_reklame_form-tglacara	Masukan Tanggal Acara (contoh : 12-12-2019 sampai 13-12-2019)	Selamat, data anda berhasil di masukan
8.	Pengajuan_reklame_syarat	Syarat reklame	1. Fotocopy Lokasi Bangunan yang akan didirikan reklame (TLB Reklame) 2. Fotocopy Izin Mendirikan Bangunan (IMB) jika reklame melekat di bangunan 3. Izin Pelaksanaan Teknis Bangunan (IPTB) dengan penanggung Jawab dari perencana Arsitektur 4. Fotocopy Bukti Kepemilikan



			<p>Tanah dimana jenis Bukti Kepemilikan tanah harus yang dapat diterima di PTPS seperti Sertifikat Hak Guna Bangunan, Sertifikat Hak Milik, Sertifikat Hak Pengelolaan dan Sertifikat Hak Pakai</p> <p>5. Mengajukan Surat Permohonan dan pernyataan dengan materai 6000 (dapat dibuat di sini)</p> <p>6. Identitas pemohon seperti KTP, KK, dan NPWP</p> <p>7. Saat Diwakilkan atau dikuasakan maka harus membuat surat kuasa di atas kertas dengan materai</p>
--	--	--	--

			<p>Rp. 6000 dan KTP Orang yang diberikan kuasa</p> <p>8. Fotocopy Bukti Pembayaran PBB Tahun Terakhir</p> <p>9. Surat Pernyataan Pemohon tidak akan mengubah bentuk reklame di atas materai Rp. 6000</p> <p>10. Proposal Teknis</p> <p>11. Jika melakukan perpanjangan maka dilengkapi dengan Izin</p>
9.	Surat_pengantar_dagang	Form perdagangan	Masukan nama anda
10.	Surat_pengantar_dagang-nama	yoga	Masukan nik anda
11.	Surat_pengantar_dagang-nama- nik	1234567890984356	Masukan alamat anda
12.	Surat_pengantar_dagang-nama- alamat	denpasar	Masukan nomer telepon/hp :

13.	Surat_pengantar_dagang-nama-notlpn	085333864698	Selamat, data anda berhasil di masukan
14.	Surat_pengantar_dagang_detail	restaurant	Masukan Alamat usaha :
15.	Surat_pengantar_dagang_detail-alamatusaha	Denpasar	Pilih jenis usaha :
16.	Surat_pengantar_dagang_detail-jenissusaha	pt	Masukan Nomer Telepon Usaha :
17.	Surat_pengantar_dagang_detail-notlpnusaha	085333864698	Selamat, data anda berhasil di masukan
18.	Syarat_perdagangan	Syarat perdagangan	<p>1. Badan usaha telah memiliki akta pendirian, persetujuan dari lembaga yang berwenang, memiliki domisili usaha dan memiliki NPWP badan usaha</p> <p>2. Pemilik atau pelaku usaha mengurus sendiri atau melalui kuasa yang dikuasakan PTSP dimana</p>

			<p>lokasi usaha berada</p> <p>3. Mengambil formulir pendaftaran, mengisi formulir SIUP dan ditandatangani bermaterai Rp 6.000 yang ditandatangani oleh pemilik usaha. (dapat di buat di sini)</p> <p>4. Mengisi pernyataan-pernyataan yang dibutuhkan oleh permohonan SIUP</p> <p>5. Dilengkapi dengan syarat – syarat berikut :</p> <ul style="list-style-type: none"> <li>- Fotocopy akte pendirian badan usaha dan persetujuan dari lembaga yang berwenang</li> <li>- Fotocopy KTP direktur</li> </ul>
--	--	--	---

			<ul style="list-style-type: none"> <li>- Fotocopy NPWP direktur</li> <li>- Fotocopy NPWP badan usaha</li> <li>- Fotocopy domisili usaha</li> <li>- Neraca perusahaan</li> <li>- Pas foto direktur 3 x 4, 2 lembar.</li> <li>Background, tergantung PTSP yang dimohonkan</li> </ul>
--	--	--	--

Tabel 4.1 menunjukkan komunikasi yang terjadi antara *user* dan bot, dimana respon yang diterima oleh *user* akan sesuai dengan respon yang ada pada *Intent* sesuai dengan permintaan atau *training phase* yang *user* inputkan.

#### 4.1.1 Pembuatan Agent

Langkah awal adalah membuat sebuah *Agent* yang akan digunakan sebagai modul, di dalam *Agent* atau modul inilah permintaan yang datang akan diproses. Pembuatan *Agent* yang perlu dilengkapi adalah nama *Agent* dan *description* yang berifat optional.

The screenshot shows the Google Assistant console interface for creating a new agent. The agent is named 'OtonlogyBot'. The 'General' tab is selected, showing a description field (empty), a default time zone dropdown set to '(GMT+8:00) Asia/Hong\_Kong', and a table for Google Project settings.

GOOGLE PROJECT	
Project ID	otonlogybot-jsiadw
Service Account ⓘ	dialogflow-ytxehh@otonlogybot-jsiadw.iam.gserviceaccount.com

**Gambar 4.2** Pembuatan Agent

Gambar 4.2 merupakan pembuatan *Agent* dengan nama OtonlogyBot, dalam pembuatan *Agent* ini hanya memasukkan nama saja tidak menyertakan deskripsi dari *Agent* tersebut.

#### 4.1.2 Pembuatan Intent

*Agent* perlu memiliki beberapa sampel yang berkaitan dengan pertanyaan tersebut, oleh karena itu diperlukan untuk mendefinisikan terlebih dahulu persamaan pertanyaan yang akan memiliki kesamaan dengan pertanyaan yang dikirim *user* ke dalam sebuah *Intent*. Semakin banyak variasi pertanyaan yang didefinisikan maka akan sangat membantu sistem untuk menentukan jawaban yang tepat. Membuat *Intent* dapat dilakukan dengan menekan tombol CREATE INTENT.

•
menu

SAVE

” pelayanan yang tersedia
” menu yang tersedia
” pelayanan menu
” pelayanan
” menu

Action and parameters

pelayanan&menu

REQUIRED ?	PARAMETER NAME ?	ENTITY ?	VALUE	IS LIST ?
<input type="checkbox"/>	pelayanan	@sys_pelaya nan	Spelayanan	<input type="checkbox"/>
<input type="checkbox"/>	menu	@sys_pelaya nan	Smenu	<input type="checkbox"/>

Fulfillment ?

☒ Enable webhook call for this intent

☐ Enable webhook call for slot filling

**Gambar 4.3** Pembuatan Intent

Gambar 4.3 merupakan pembuatan atau indentifikasi pertanyaan pada sebuah *Intent*, yang berisikan beberapa kata terutama “menu” dan “pelayanan” . Kata tersebut merupakan sebuah perkiraan pertanyaan yang memiliki kemungkinan besar *user* tanyakan. *Intent* ini dibuat dengan tujuan untuk menampung segala

pertanyaan yang berkaitan dengan pelayanan dari bot. Parameter ini adalah parameter yang digunakan untuk mendefinisikan suatu kata kedalam suatu aksi yang berisi entitas dan nilai parameter dari sebuah aksi. *Fulfillment* diaktifkan untuk menggunakan informasi yang diekstraksi oleh NLP *Dialogflow* untuk menghasilkan respon dinamis atau memicu tindakan di *back-end*.



## Intents

[CREATE INTENT](#)

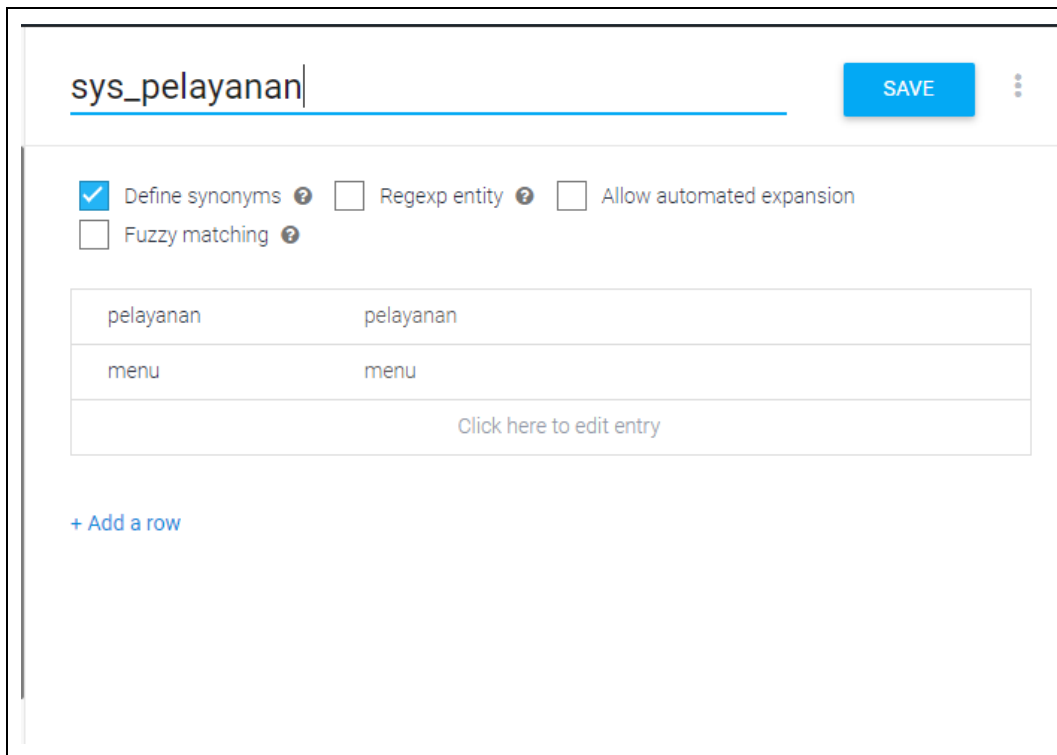
- ☒ Default Fallback Intent
- ☒ Default Welcome Intent
- ☒ menu
- ☒ menu.pengajuan.dagang
- ☒ pengajuan\_reklame
- ☒ pengajuan\_reklame\_form ^
- ☒ ↳ pengajuan\_reklame\_form - namauser ^
- ☒ ↳ pengajuan\_reklame\_form - namaacara ^ Add follow-up intent
- ☒ ↳ pengajuan\_reklame\_form - tglacara Add follow-up intent
- ☒ pengajuan\_reklame\_syarat
- ☒ surat\_pengantar\_dagang ^
- ☒ ↳ surat\_pengantar\_dagang - nama ^
- ☒ ↳ surat\_pengantar\_dagang - nama - nik ^
- ☒ ↳ surat\_pengantar\_dagang - nama - alamat ^
- ☒ ↳ surat\_pengantar\_dagang - nama - no.tlpn
- ☐ surat\_pengantar\_dagang\_detail ^ Add follow-up intent
- ☒ ↳ surat\_pengantar\_dagang\_detail - alamatusaha ^
- ☒ ↳ surat\_pengantar\_dagang\_detail - jenususaha ^
- ☒ ↳ surat\_pengantar\_dagang\_detail - notlpnusaha
- ☒ syarat\_perdagangan
- ☒ webhook-intent v

**Gambar 4.4** Pembuatan Intent

Gambar 4.4 menunjukkan *Intent-intent* yang telah dibuat. *Intent* merupakan sebuah tempat untuk memetakan pertanyaan apa yang dikirim oleh pengguna dan tindakan apa yang harus dilakukan. Tujuan dari *Intents* adalah mendefinisikan tata bahasa percakapan dan tugas apa yang harus dilakukan saat pengguna menggunakan frasa tertentu.

#### 4.1.3 Pembuatan Entitas

Entitas yang dibuat pada proses pembuatan *Entities* ini adalah *Entities* sumber daya, dimana pada entitas ini terdapat daftar sumber daya yang dapat dipesan atau di *booking* oleh karyawan. Membuat entitas dapat menekan tombol CREATE ENTITY.



sys\_pelayanan | SAVE

☒ Define synonyms ⓘ ☐ Regex entity ⓘ ☐ Allow automated expansion  
☐ Fuzzy matching ⓘ

pelayanan	pelayanan
menu	menu
<a href="#">Click here to edit entry</a>	

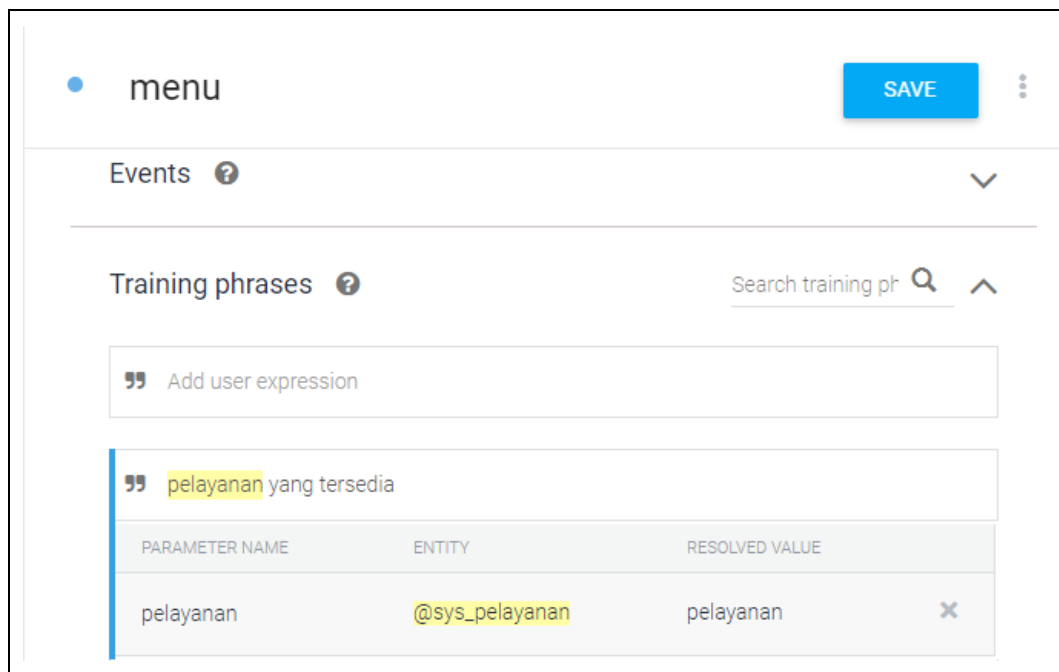
+ Add a row

**Gambar 4.5** Pembuatan Entitas

Gambar 4.5 merupakan pembuatan *Entities*, dimana sinonim pada *Entities* “sys\_pelayanan” bertujuan untuk memberikan variasi kata, karena setiap *user* akan

memiliki perbedaan kata, sehingga dibuatlah sinonim atau persamaan kata untuk menangani perbedaan kata tetapi memiliki arti dan tujuan yang sama.

Pengambilan informasi di dalam *Entities*, diperlukan informasi salam sapa berupa kata “menu” atau “pelayanan”, di dalam *Intents* “menu” inilah penggalian informasi salam sapa dilakukan. Frasa atau kamus yang telah di buat memiliki parameter berupa “sys\_pelayanan”, dimana apabila *user* mengirim *request text* berupa kata yang ada pada *Entities* “sys\_pelayanan” maka akan langsung dikenali sebagai sebuah sapaan dan disimpan di dalam parameter untuk dikelola lebih lanjut.



**Gambar 4.6** Memanggil Entitas

*Entities* yang digunakan adalah *entities built in* yang terdapat di dalam sistem ini yaitu *Entities* yang di definisikan sendiri yaitu “sys\_pelayanan”. Kedua *Entities* yang diterapkan di dalam frasa nantinya yang akan dicocokkan dengan pertanyaan yang datang apakah terdapat kata yang teridentifikasi memiliki kesesuaian baik berupa kata yang mirip dengan kata kunci “menu”.

### Action and parameters

pelayanan&menu

REQUIRED ?	PARAMETER NAME ?	ENTITY ?	VALUE	IS LIST ?
<input type="checkbox"/>	pelayanan	@sys_pelaya nan	Spelayanan	<input type="checkbox"/>
<input type="checkbox"/>	menu	@sys_pelaya nan	Smenu	<input type="checkbox"/>
<input type="checkbox"/>	Enter name	Enter entity	Enter value	<input type="checkbox"/>

[+ New parameter](#)

**Gambar 4.7** Parameter di Intent

Gambar 4.7 terdapat nama parameter “sys\_pelayanan”. Parameter ini adalah parameter yang digunakan untuk mendefinisikan suatu kata kedalam suatu aksi yang berisi entitas dan nilai parameter dari sebuah aksi, dimana nilai yang diberikan pada parameter ini adalah “\$pelayanan” agar mampu memberikan respon kepada *user* sesuai dengan apa yang *user* ucapkan, dan kata tersebut sebelumnya sudah didefinisikan pada entitas “sys\_pelayanan”.

## 4.2 Tahap Fulfillment

Tahap *fulfillment* ini akan dijelaskan sisi *back-end* dari rancangan bot yang dibuat pada Dialogflow. Tahap ini membahas bagaimana pembuatan data menu pada MySQL dan bagaimana memanggil serta memasukkan data pesanan dan *booking* tempat pada *fulfillment* Dialogflow.

#### 4.2.1 Pembuatan Data pada MySQL

Tahap pembuatan data di MySQL yang dihosting pada db4free, data yang akan dibuat adalah data pengajuan izin usaha dagang yang nantinya dapat didaftarkan oleh *user*, data respon, dan beberapa tabel.


id	id_user	nik	nama_user	alamat	no_tlp	tanggal
2	349670871	31223123123123123	yoga	dps	0897123712637123	2019-10-16
3	349670871	213123123123	yoga sukra	denpasar	089322312312312	2019-10-16
4	349670871	170555201212	Angga	Seroja	089120398129871	2019-10-16
5	349670871	12312312312332	yoga	denpasar	0892312312	2019-10-16
6	349670871	12313123123	yoga	dps	123123123	2019-10-16
7	349670871	2312312312312	dwik	dps	0892313123	2019-10-16
8	349670871	123123123123	yoga	dps	0892312312312	2019-10-16
9	349670871	23131312313	angga	dps	0893123123123	2019-10-16
10	349670871	123123123123	doni	dps	21312312323	2019-10-16
11	349670871	123123123123	yoga	denpasar	08213123123	2019-10-16
12	349670871	123123123123123	yoga	denpsasa	098391232	2019-10-16

**Gambar 4.8** Pembuatan data tb\_user\_pengaju

Gambar 4.8 merupakan isi dari tabel tb\_user\_pengaju, dimana data yang terdapat pada tabel tersebut ini adalah id\_user, nik, nama\_user, alamat, no\_tlp, dan tanggal dari pengajuan.


#### 4.2.2 Pembuatan Fulfillment

Pembuatan *fulfillment*, yang harus dilakukan pertama kali adalah membuat akun pada Heroku dan kemudian dihubungkan dengan repositori yang telah dibuat pada Github. Menghubungkan *fulfillment* Webhook dengan aplikasi *back-end* dapat dilakukan dengan mengaktifkan Webhook pada menu *Fulfillment* Dialogflow dan memberikan alamat dari *server* Heroku.

 Fulfillment

---

## Webhook

ENABLED 

Your web service will receive a POST request from Dialogflow in the form of the response to a user query matched by intents with webhook enabled. Be sure that your web service meets all the [webhook requirements](#) specific to the API version enabled in this agent.

URL\*

https://patenbot12.herokuapp.com/

BASIC AUTH

yogasukrasena

.....


HEADERS

Enter key

Enter value

Enter key

Enter value

 Add header

SMALL TALK

Disable webhook for Smalltalk

▼

**Gambar 4.9** Pembuatan data menu

Gambar 4.9 merupakan langkah untuk menghubungkan Dialogflow dengan aplikasi *back-end* yang berada pada *server* Heroku. Membuat API dengan Python, kode program pada *file* `app.py` dapat dilihat sebagai berikut.

```
# import flask dependencies
from flask import Flask, request, jsonify
import os
import pymysql.cursors
import json
from datetime import date
import telebot
from fpdf import FPDF

bot =
telebot.TeleBot(token='816398857:AAEZGcAQZ00QR1kYqEBV7nZTFWzNv2gD26g')

# initialize the flask app
app = Flask(__name__)
PORT = int(os.environ.get("PORT", 5000))
```

```

connection = pymysql.connect(host='db4free.net',
                             user='yogasukrasena',
                             password='Ikadek07',
                             db='db_paten_yoga',
                             charset='utf8mb4',

cursorclass=pymysql.cursors.DictCursor)

# create a route for webhook
@app.route('/', methods=['POST'])
def webhook():
    data = request.get_json()
    intent_name =
data.get("queryResult").get("intent").get("displayName")
    print(data)

    if intent_name == 'menu':
        return menu(data)

    elif intent_name == 'menu.pengajuan.dagang':
        return perdagangan()

    elif intent_name == 'surat_pengantar_dagang_detail -
notlpnusaha':
        return dataUserPengaju(data)

    elif intent_name == 'pengajuan_reklame':
        return reklame()

    elif intent_name == 'pengajuan_reklame_form - tglacara':
        return formReklame(data)

    return jsonify(request.get_json())

def menu(data):
    cekUserID =
data.get("originalDetectIntentRequest").get("payload").get("from
").get("id")
    idPesan =
data.get("originalDetectIntentRequest").get("payload").get("mess
age_id")
    isiPesan =
data.get("originalDetectIntentRequest").get("payload").get("text
")
    userNama =
data.get("originalDetectIntentRequest").get("payload").get("from
").get("username")
    id_inbox = ""

    try:
        result = ""
        with connection.cursor() as cursor:
            sql = "INSERT INTO tb_inbox (id_pesan, pesan,
userID, tanggal) VALUES (%s, %s, %s, %s)"
            cursor.execute(sql, (idPesan, isiPesan, cekUserID,

```

```

date.today().strftime("%Y-%m-%d"))
    id_inbox = cursor.lastrowid
    result = cursor.fetchone()
    connection.commit()

    response = {
        'fulfillmentMessages': [
            {
                "card": {
                    "title": "Menu",
                    "subtitle": "Halo {}, Silahkan pilih
menu di bawah".format(userName),
                    "buttons": [
                        {
                            "text": "Surat izin Usaha
Perdagangan",
                            "postback": "usaha perdagangan"
                        },
                        {
                            "text": "Pengajuan Izin
Reklame",
                            "postback": "pengajuan reklame"
                        }
                    ]
                }
            }
        ]
    }

    with connection.cursor() as cursor:
        sql = "INSERT INTO tb_outbox (id_inbox, response)
VALUES (%s, %s)"
        cursor.execute(sql, (id_inbox, "Halo {}, Silahkan
pilih menu di bawah".format(userName)))
        sql = "UPDATE tb_inbox SET tb_inbox.status = '1'
WHERE tb_inbox.id = %s"
        cursor.execute(sql, (id_inbox))
        result = cursor.fetchone()
        connection.commit()

    return response

except Exception as error:
    print(error)
    response = {
        'fulfillmentText': "Data anda gagal di Daftarkan"
    }
    return jsonify(response)

def perdagangan():
    response = {
        'fulfillmentMessages': [
            {
                "card": {
                    "title": "SIUP",
                    "subtitle": "Silahkan pilih layanan di

```



```

Bawah",
        "buttons": [
            {
                "text": "Form Pendaftaran",
                "postback": "form perdagangan"
            },
            {
                "text": "Syarat Pengajuan
Perdagangan",
                "postback": "syarat Perdagangan"
            }
        ]
    }
}
]
}
return response

def reklame():
    response = {
        'fulfillmentMessages': [
            {
                "card": {
                    "title": "Izin Reklame",
                    "subtitle": "Silahkan pilih layanan di
Bawah",
                    "buttons": [
                        {
                            "text": "Form Pendaftaran",
                            "postback": "form reklame"
                        },
                        {
                            "text": "Syarat Pemasangan Reklame",
                            "postback": "syarat reklame"
                        }
                    ]
                }
            }
        ]
    }
    return response

def dataUserPengaju(data):
    parameter_index = 0
    outputContexts = data['queryResult']['outputContexts']

    for index, parameter in enumerate(outputContexts):
        if parameter['name'] == "projects/otonlogybot-
jsiadw/agent/sessions/" \
                                "53cffdfa-98d0-3832-8930-
b0dd520ef777/contexts/surat_pengantar_dagang-nama-followup":
            parameter_index = index

    cekUserID =
data.get("originalDetectIntentRequest").get("payload").get("from
").get("id")

```

```

        namauser =
outputContexts[parameter_index]['parameters']['namauser']
        nikuser =
outputContexts[parameter_index]['parameters']['nik']
        alamat =
outputContexts[parameter_index]['parameters']['alamat']
        notlpn =
outputContexts[parameter_index]['parameters']['notlpn']
        namausaha =
outputContexts[parameter_index]['parameters']['namausaha']
        alamatusaha =
outputContexts[parameter_index]['parameters']['alamatusaha']
        jenisusaha =
outputContexts[parameter_index]['parameters']['jenisusaha']
        notlpnusaha =
outputContexts[parameter_index]['parameters']['notlpnusaha']

    try:
        result = ""
        with connection.cursor() as cursor:
            sql = "INSERT INTO tb_user_pengaju (id_user, nik,
nama_user, alamat, no_tlp, tanggal) VALUES (%s, %s, %s, %s, %s,
%s)"
            cursor.execute(sql, (cekUserID, nikuser, namauser,
alamat, notlpn, date.today().strftime("%Y-%m-%d")))
            id_inbox = cursor.lastrowid
            sql2 = "INSERT INTO tb_detail_user_pengaju
(id_user_pengaju, nama_usaha, alamat_usaha, jenis_usaha,
no_tlpn_usaha) VALUES (%s, %s, %s, %s, %s)"
            cursor.execute(sql2, (id_inbox, namausaha,
alamatusaha, jenisusaha, notlpnusaha,))
            result = cursor.fetchone()
            connection.commit()

        pdf = FPDF()
        pdf.add_page()
        pdf.set_font("Arial", size=12)
        pdf.cell(200, 10, txt="Pengajuan Surat Izin Usaha
Perdagangan", ln=1, align="C")
        pdf.cell(200, 10, txt="Kecamatan Ubud, Kabupaten
Gianyar, Provinsi Bali", ln=1, align="C")
        pdf.cell(200, 10, txt="Nama Pengaju :
{}".format(namauser), ln=1, align="J")
        pdf.cell(200, 10, txt="Nik : {}".format(nikuser), ln=1,
align="J")
        pdf.cell(200, 10, txt="Alamat : {}".format(alamat),
ln=1, align="J")
        pdf.cell(200, 10, txt="Nomer Telepon :
{}".format(notlpn), ln=1, align="J")
        pdf.cell(200, 10, txt="Usaha Pengaju", ln=1, align="C")
        pdf.cell(200, 10, txt="Nama Usaha :
{}".format(namausaha), ln=1, align="J")
        pdf.cell(200, 10, txt="Alamat Usaha :
{}".format(alamatusaha), ln=1, align="J")
        pdf.cell(200, 10, txt="Jenis Usaha :
{}".format(jenisusaha), ln=1, align="J")

```

```

        pdf.cell(200, 10, txt="Nomer Telepon Usaha :
{}".format(notlpnusaha), ln=1, align="J")
        pdf.output("form_pendaftaran.pdf")

        bot.send_document(cekUserID,
open('form_pendaftaran.pdf', 'rb'))

        with connection.cursor() as cursor:
            sql = "INSERT INTO tb_outbox (id_inbox, response)
VALUES (%s, %s)"
            cursor.execute(sql, (id_inbox,
"form_pendaftaran.pdf"))
            sql = "UPDATE tb_inbox SET tb_inbox.status = '1'
WHERE tb_inbox.id = %s"
            cursor.execute(sql, (id_inbox))
            result = cursor.fetchone()
            connection.commit()

        response = {
            'fulfillmentText': "Selamat, data anda berhasil di
masukan"
        }
        return jsonify(response)

    except Exception as error:
        print(error)

        response = {
            'fulfillmentText': "Data anda gagal di Daftarkan"
        }
        return jsonify(response)

def formReklame(data):
    parameter_index = 0
    outputContexts = data['queryResult']['outputContexts']

    for index, parameter in enumerate(outputContexts):
        if parameter['name'] == "projects/otonlogybot-
jsiadw/agent/sessions/" \
                                "53cffdfa-98d0-3832-8930-
b0dd520ef777/contexts/pengajuan_reklame_form-followup":
            parameter_index = index

    cekUserID =
data.get("originalDetectIntentRequest").get("payload").get("from
").get("id")
    namauser =
outputContexts[parameter_index]['parameters']['namapenga']
    namaacara =
outputContexts[parameter_index]['parameters']['namaacara']
    tglacara =
outputContexts[parameter_index]['parameters']['tglacara']

    try:
        result = ""

```

```

        with connection.cursor() as cursor:
            sql = "INSERT INTO tb_reklame (id_user,
nama_pengaju, nama_acara, tanggal_acara, tanggal_pengajuan)
VALUES (%s, %s, %s, %s, %s)"
            cursor.execute(sql, (cekUserID, namauser, namaacara,
tglacara, date.today().strftime("%Y-%m-%d")))
            id_inbox = cursor.lastrowid
            result = cursor.fetchone()
            connection.commit()

        pdf = FPDF()
        pdf.add_page()
        pdf.set_font("Arial", size=12)
        pdf.cell(200, 10, txt="Pengajuan Surat Izin Pemasangan
Baliho", ln=1, align="C")
        pdf.cell(200, 10, txt="Kecamatan Ubud, Kabupaten
Gianyar, Provinsi Bali", ln=1, align="C")
        pdf.cell(200, 10, txt="Nomor: 156/C/PANPEL-{} /I/2019
".format(namaacara), ln=1, align="J")
        pdf.cell(200, 10, txt="Lampiran: - ", ln=1, align="J")
        pdf.cell(200, 10, txt="Perihal: Permohonan Izin
Pemasangan Baliho", ln=1, align="J")
        pdf.cell(200, 10, txt="Kepada : ", ln=1, align="J")
        pdf.cell(200, 10, txt="Yth.      Camat ", ln=1, align="J")
        pdf.cell(200, 10, txt="Kecamatan Ubud", ln=1, align="J")
        pdf.cell(200, 10, txt="di- ", ln=1, align="J")
        pdf.cell(200, 10, txt="Tempat", ln=1, align="J")
        pdf.cell(200, 10, txt="Dengan hormat, ", ln=1,
align="J")
        pdf.cell(200, 10, txt="Sehubungan dengan akan
diselenggarakan kegiatan {} oleh {} , ".format(namaacara,
namauser), ln=1, align="J")
        pdf.cell(200, 10, txt="maka kami selaku panitia
pelaksana mohon izin pemasangan baliho, pada:", ln=1, align="J")
        pdf.cell(200, 10, txt="Tanggal : {} ".format(tglacara),
ln=1, align="J")
        pdf.cell(200, 10, txt="Nama Acara :
{}".format(namaacara), ln=1, align="J")
        pdf.cell(200, 10, txt="Demikian surat permohonan ini
kami sampaikan, besar harapan kami agar Bapak/Ibu bersedia ",
ln=1, align="J")
        pdf.cell(200, 10, txt="dan berkenan untuk membantu
mensukseskan acara tersebut. Atas perhatian dan bantuan ", ln=1,
align="J")
        pdf.cell(200, 10, txt="Bapak/Ibu kami ucapkan terima
kasih. ", ln=1, align="J")

        pdf.output("form_pendaftaran.pdf")

        bot.send_document(cekUserID,
open('form_pendaftaran.pdf', 'rb'))

        with connection.cursor() as cursor:
            sql = "INSERT INTO tb_outbox (id_inbox, response)
VALUES (%s, %s)"
            cursor.execute(sql, (id_inbox,

```

```

"form_pendaftaran.pdf"))
        sql = "UPDATE tb_inbox SET tb_inbox.status = '1'
WHERE tb_inbox.id = %s"
        cursor.execute(sql, (id_inbox))
        result = cursor.fetchone()
        connection.commit()

        response = {
            'fulfillmentText': "Selamat, data anda berhasil di
masukan"
        }
        return jsonify(response)

    except Exception as error:
        print(error)
        response = {
            'fulfillmentText': "Data anda gagal di Daftarkan"
        }
        return jsonify(response)

# run the app
if __name__ == '__main__':
    app.run(port=PORT, host='0.0.0.0')

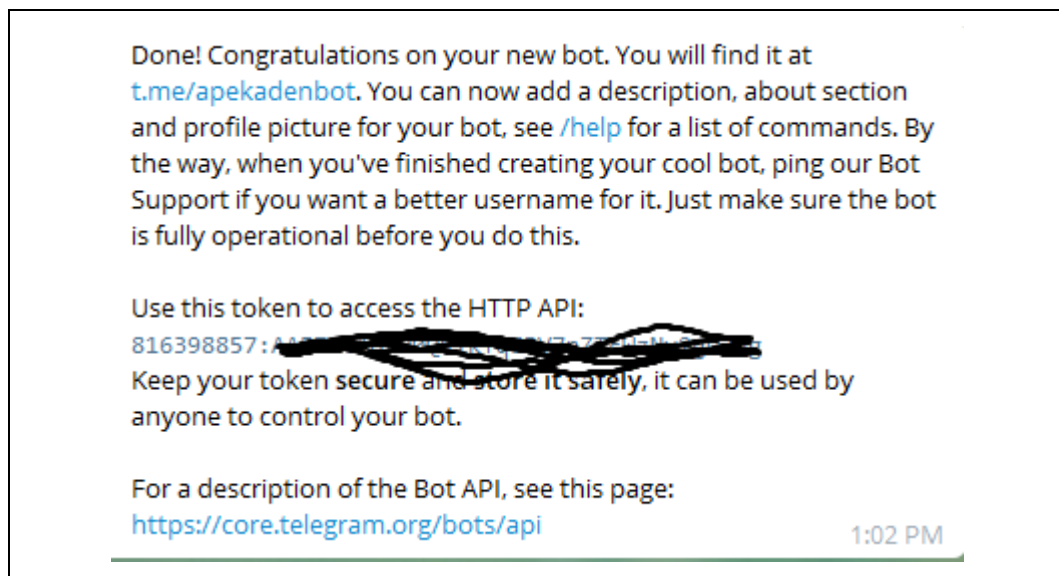
```

**Kode Program 4.1** app.py

Kode program 4.1 merupakan kode program dari *file* app.py, dimana isi dari kode program ini termasuk cara menghubungkan dengan Dialogflow *Fulfillment* dan *database*.

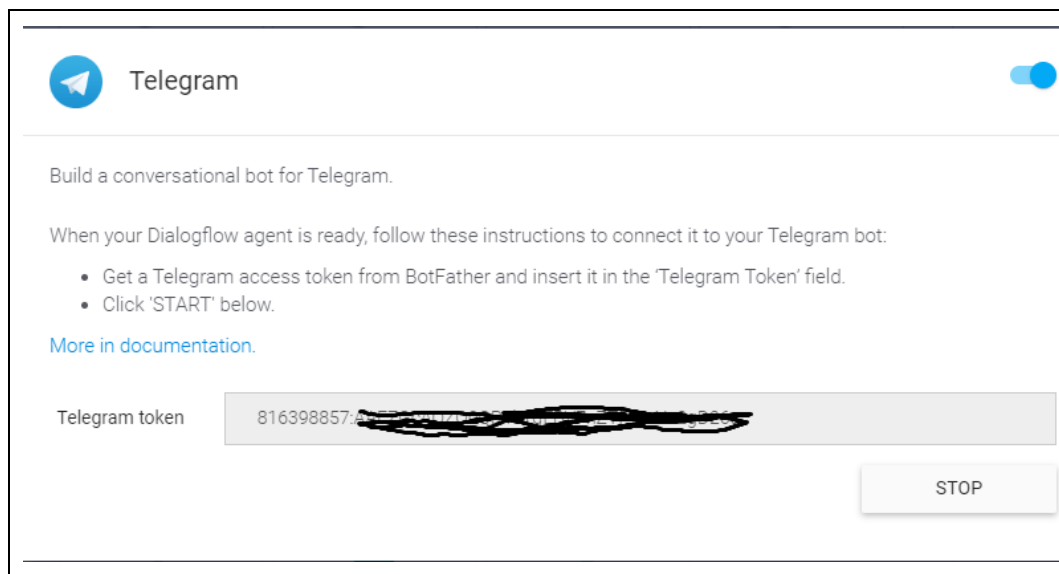
#### **4.2.3 Percobaan Bot**

Langkah pertama adalah membuat akun bot melalui BotFather, kemudian dilanjutkan dengan konfigurasi akun bot tersebut. Langkah selanjutnya adalah menghubungkan Dialogflow dengan Telegram, setelah melakukan pendaftaran pada BotFather maka akan mendapatkan token access untuk dimasukan ke dalam dialogflow.



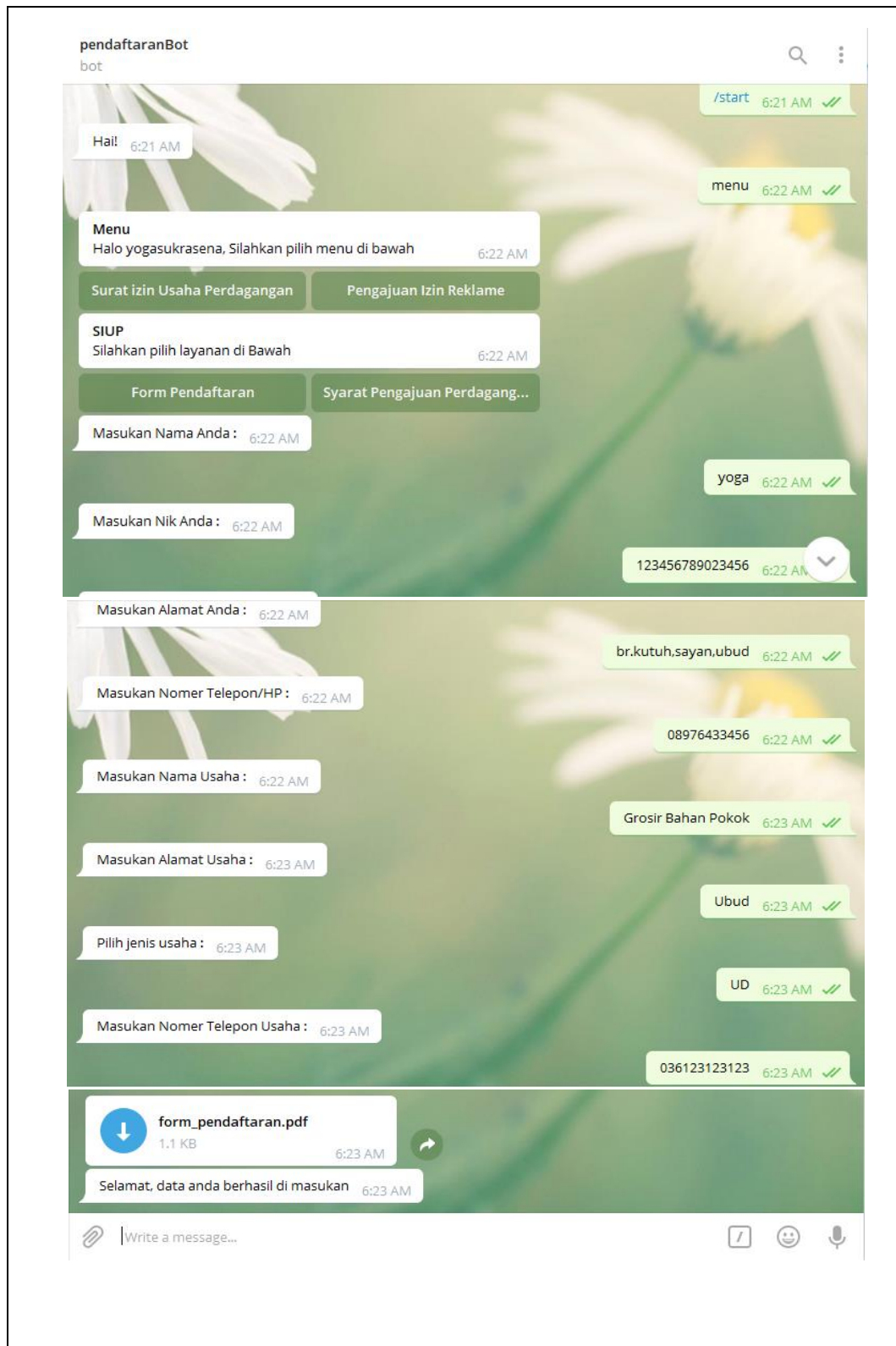
**Gambar 4.10** Pembuatan token

Gambar 4.10 menunjukkan pembuatan token. Selanjutnya setelah token berhasil dibuat, pindah pada Dialogflow, pada menu Intergrations pilih & aktifkan Telegram, selanjutnya akan muncul panel konfigurasi dan masukan token yang telah di dapatkan tadi.



**Gambar 4.11** Integrasi ke Dialogflow

Gambar 4.11 merupakan langkah untuk menghubungkan Dialogflow dengan Telegram. Integrasi bot dengan Telegram telah selesai dibuat, sekarang melakukan uji coba bot yang dibuat tadi dengan *Intent* yang telah ditentukan.



**Gambar 4.13** Percakapan Bot

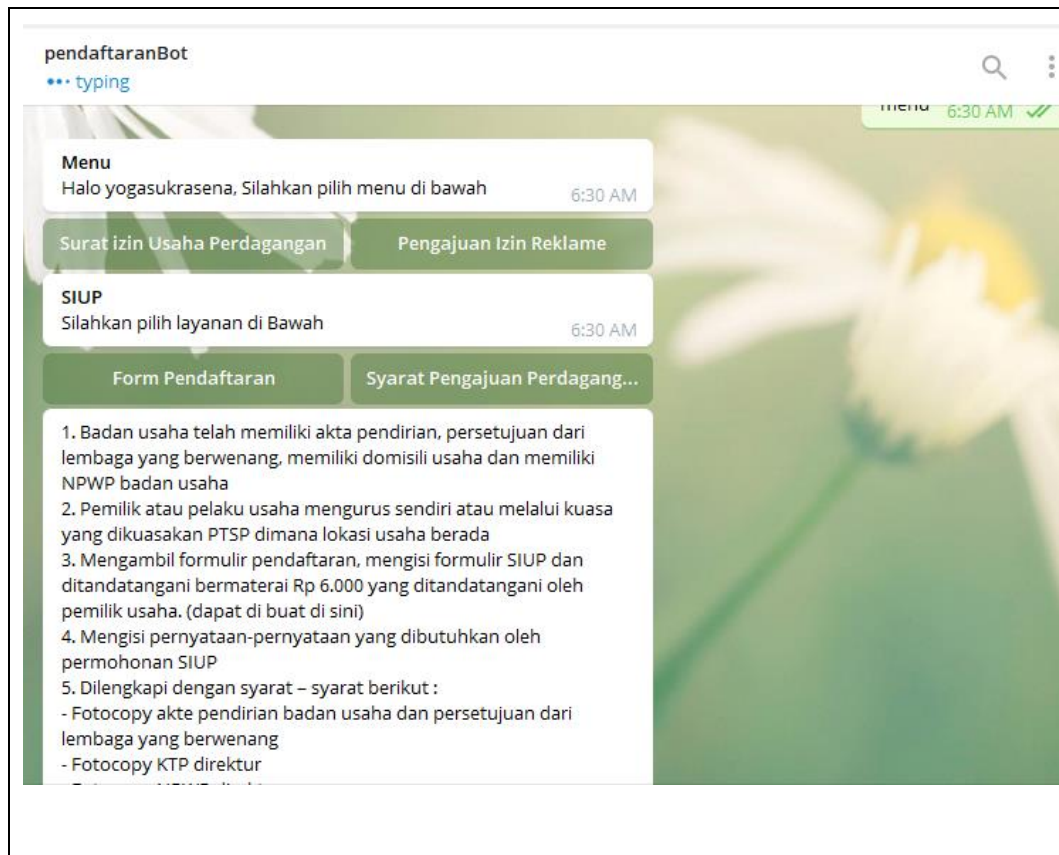


Gambar 4.13 merupakan salah satu percakapan pada bot yang menawarkan layanan pembuatan surat pengantar pengajuan izin usaha perdagangan, dengan memilih menu siup dan mengikuti data yang harus di inputkan, maka akan menghasilkan surat pengantar berbentuk pdf dengan isi seperti gambar di bawah

Pengajuan Surat Izin Usaha Perdagangan	
Kecamatan Ubud, Kabupaten Gianyar, Provinsi Bali	
Nama Pengaju :	yoga
Nik :	123456789023456
Alamat :	br.kutuh,sayan,ubud
Nomer Telepon :	08976433456
Usaha Pengaju	
Nama Usaha :	Grosir Bahan Pokok
Alamat Usaha :	Ubud
Jenis Usaha :	UD
Nomer Telepon Usaha :	036123123123

**Gambar 4.14** hasil generate data percakapan

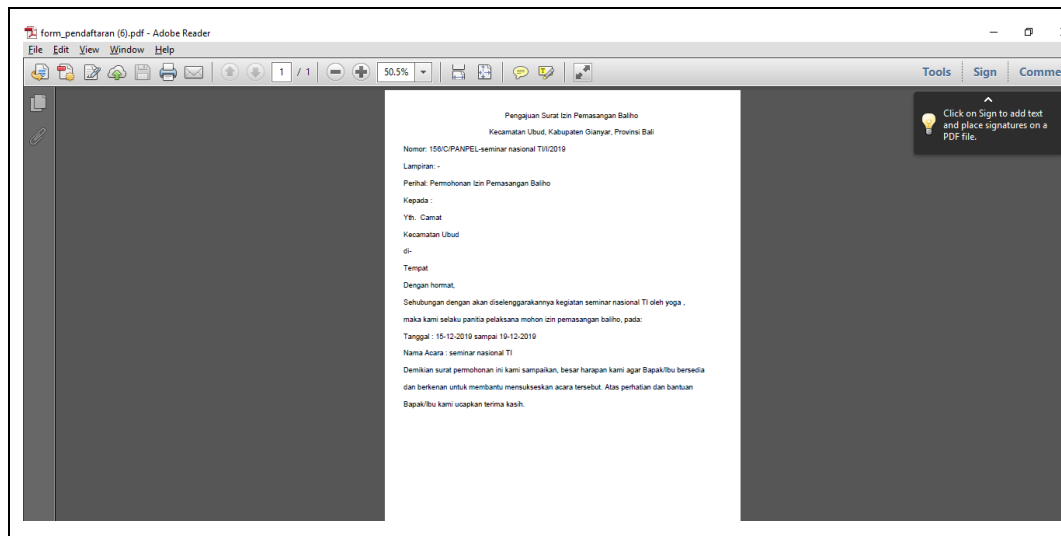
Gambar 4.14 merupakan hasil dari percakapan pada bot yang menunjukkan bahwa *user* telah membuat surat pengantar sebagai salah satu syarat pembuatan SIUP.



**Gambar 4.15** Percakapan Syarat pengajuan SIUP

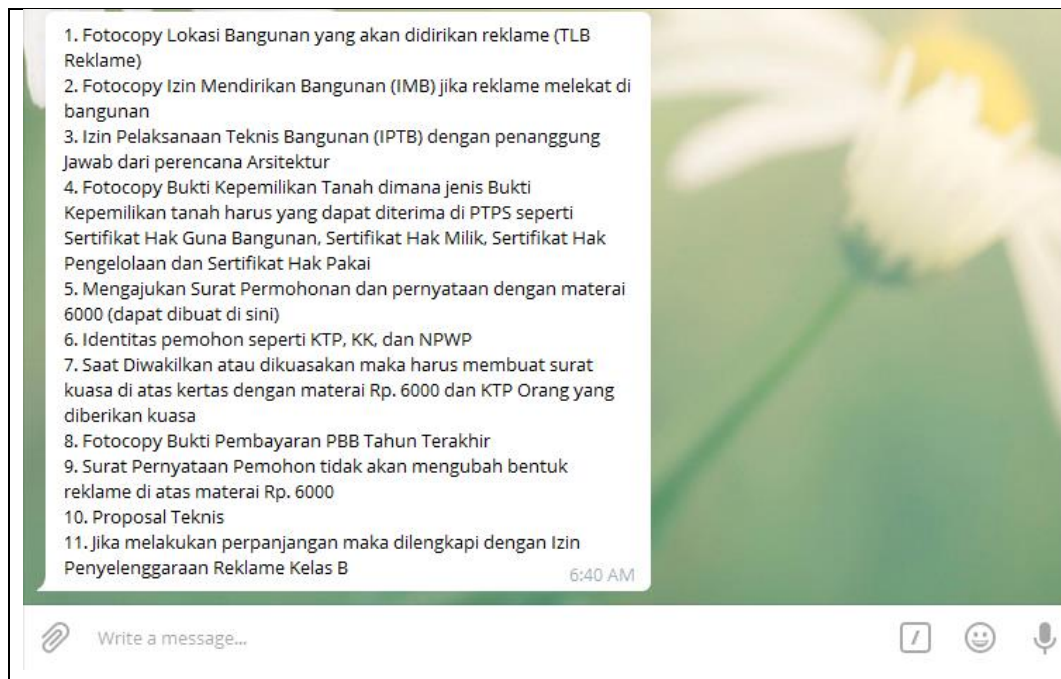
Gambar 4.15 merupakan percakapan syarat pembuatan SIUP, dengan menekan tombol syarat pengajuan perdagangan maka akan di tampilkan data yang dibutuhkan untuk pembuatan SIUP.





**Gambar 4.17** hasil pembuatan izin reklame

Gambar 4.17 merupakan percakapan surat pengantar sebagai salah satu syarat izin pemasangan reklame, untuk melihat syarat apa saja yang dibutuhkan dalam pemasangan reklame, tekan menu syarat pemasangan reklame maka hasilnya akan seperti gambar di bawah.



**Gambar 4. 18** syarat pemasangan reklame

Gambar 4.18 merupakan hasil dari pemilihan menu syarat pemasangan reklame, pada menu tersebut akan dihasilkan syarat-syarat yang dibutuhkan untuk pemasangan reklame.

## **BAB V**

### **PENUTUP**

#### **5.1 Kesimpulan**

Projek chatbot ini menggunakan Dialogflow dan Webhook yang sudah terhubung dengan aplikasi *back-end*, dimana berperan untuk membuat bot yang memiliki tindakan kompleks. Chatbot sangat efisien dan dapat menghemat waktu dalam pembuatan surat pengantar SIUP dan pemasangan reklame

#### **5.2 Saran**

Saran yang dapat diberikan dari pembuatan laporan ini yaitu Chatbot perlu dilakukan pengembangan lebih lanjut dalam pembuatannya untuk meningkatkan fungsionalitas dalam penggunaannya. Peningkatan fungsionalitas dapat dilakukan baik itu secara *back-end* maupun *front-end*. Perlunya pembahasan lebih mendalam mengenai Dialogflow untuk membantu dalam hal pemahaman terkait dengan Dialogflow dan kegunaan fulfillment pada Dialogflow agar dapat menciptakan respon atau balasan yang dinamis. Penggunaan webhook atau API luar yang dapat membantu untuk memudahkan dalam pembuatan bot sangatlah dianjurkan. Perlu penyempurnaan lebih jauh untuk menyempurnakan fitur yang tersedia di dalam chatbot yang telah dibuat supaya lebih layak untuk dipakai dan diterapkan.

## DAFTAR PUSTAKA

- [1] Amaliyah, R. (2019). *Pengertian Python Beserta Kelebihan dan Kekurangan Python, Sudah Tahu?*  
<https://www.nesabamedia.com/pengertian-python/>.
- [2] Hariyadi, E. (2017). *Membuat Proyek Pertama di Heroku*.  
<https://www.codepolitan.com/membuat-proyek-pertama-heroku-58b872c6217eb>.
- [3] LAB, I. (2018). *MENGENAL APA ITU CHATBOT DAN CARA KERJANYA*. <https://www.immersa-lab.com/mengenal-apa-itu-chatbot-dan-cara-kerjanya.htm>.
- [4] Mtarget. (2019). *Apa Itu Webhook dan Bagaimana Cara Menggunakannya?* <https://blog.mtarget.co/apa-itu-webhook-dan-bagaimana-cara-menggunakannya/>.
- [5] Wikipedia. (2018). *Telegram (aplikasi)*.  
[https://id.wikipedia.org/wiki/Telegram\\_\(aplikasi\)](https://id.wikipedia.org/wiki/Telegram_(aplikasi)).