Logotipo

Descripción generada automáticamente

Creación Chatbot en Motor AWS Basado en Algoritmo Neuronal

Diego Rodrigo Fernández Zamora Emmanuel Manzano Maya Miguel Angel Garcia Garay

# Resumen

El desarrollo de los chatbots ha permitido un nuevo giro en las comunicaciones, en especial a la atención de los usuarios que en general siempre van a tener dudas, dando respuestas a las preguntas frecuentes que se llegan a tener, que en el caso de las ventas de las diferentes tiendas de autoservicio, es más con el fin de proporcionar apoyo con la venta de los productos de sus diferentes departamentos que tienen, desde los más comunes hasta los demás que no son tan comunes en que tengan ventas. Llegando así a tener una mejor experiencia hacia los clientes que en general necesitan más ayuda para poder completar sus pedidos de una manera más adecuada.

Contenido

[Resumen 1](#_Toc87912985)

[Introducción 2](#_Toc87912986)

[Desarrollo 3](#_Toc87912987)

[Historia de la Tecnología 4](#_Toc87912988)

[Avance 5](#_Toc87912989)

[Código de Implementación 6](#_Toc87912990)

[Aplicación 14](#_Toc87912991)

[Problemática Para Resolver 15](#_Toc87912992)

[Conclusiones 17](#_Toc87912993)

[Bibliografía 18](#_Toc87912994)

# Introducción

Los chatbots están teniendo más presencia, ayudan bastante a las ventas y la automatización en el servicio a los clientes. Sin embargo, las experiencias de los chatbots se encuentran limitadas al especio otorgado dentro de la pantalla.

Los chatbots para web pueden proporcionar una experiencia interactiva que se encuentra mas a fondo que el especio delimitado en donde interactúa con el usuario, tambien conocidos como sistemas expertos, en donde están basados en diferentes casos que se presentan comúnmente dentro del entorno en donde se implementan, estos casos son conocidos como CBR (Case Base Reasoning).

La base de estos chatbots es una conversación mediante texto, aunque tambien hay otros modelos que disponen de una interfaz multimedia que ya permiten la entrada de audio, otros tambien ya más recientes permiten tambien la parte de dictar por comando de voz, que son en si los más recientes, algunos han comenzado a utilizar programas conversacionales mas realistas, de igual manera han reducido el tiempo de respuesta hacia los clientes.

En las conversaciones que se tienen dentro de los diferentes chatbots, usan frases fácilmente comprensibles y coherentes, aunque muchos tienen en cuenta las palabras o frases que puede usar el interlocutor (usuario) que les permiten usar una serie de repuestas predefinidas, a la vez que son capaces de reconocer la formulación de las diferentes frases en base a patrones de reconocimiento a las posibles estructuras de las frases para poder dar una respuesta concreta ya definida, permitiendo mantener una conversación más adecuada.

# Desarrollo

¿Qué pasaría si estos chatbots basados ​​en la web pudieran proporcionar una experiencia interactiva que se expandiera más allá de la ventana de chat para incluir contenido web relevante basado en las entradas del usuario? En una publicación anterior, le mostramos cómo implementar una interfaz de usuario web para su chatbot. En esta publicación le mostraremos cómo mejorar esa experiencia.

Ahora ha configurado la interfaz de usuario de chatbot mejorada. Se recomienda que se autentique para un entorno de producción. Esta publicación utiliza Amazon Cognito para agregar un proveedor de identidad social (Google) a su grupo de usuarios. Para obtener instrucciones, consulte Agregar proveedores de identidad social a un grupo de usuarios.

Por ejemplo, si le pide al bot que ordene flores, la interfaz de mensajería del bot y la página web cambian. Este ejemplo crea HTML de forma activa sobre la marcha con valores que el bot envía al usuario final.

Existen ciertas limitaciones para mostrar contenido y páginas web externas a través de la página principal de la interfaz de usuario del chatbot.

Si el uso compartido de recursos de origen cruzado (CORS) está habilitado en el contenido externo que se está introduciendo en el portal de navegación de iframe de la página principal, el navegador bloquea el contenido. Los navegadores no bloquean diferentes páginas web del mismo dominio o páginas web externas que no tienen CORS habilitado (por ejemplo, Wikipedia). Para obtener más información, consulte Intercambio de recursos entre orígenes (CORS) en el sitio web de documentos web de MDN.

En la mayoría de los casos de uso, debe utilizar el portal de navegación para extraer contenido de su propio dominio, debido a las limitaciones inherentes de los iframes y CORS.

Nuestra interfaz de usuario web de Amazon Lex de muestra, conocida como la interfaz de usuario de chatbot, ya hace la mayor parte del trabajo pesado asociado con la provisión de un cliente web con todas las funciones para los chatbots de Amazon Lex. Puede aprovechar rápidamente sus funciones y minimizar el tiempo de rentabilidad de sus aplicaciones impulsadas por chatbot. (Mohamed Khalil, 2020)

# Historia de la Tecnología

Los chatos se desarrollaron por primera vez a mediados de los años 60. “Elizza” fue el primer chatbot creado en el MIT en 1966, que mas tarde se denominó “chatter bot”. El objetivo principal de este chatbot era habilitar una interfaz de comunicación para conectarse entre la máquina y el hombre. Al ser una innovación pionera de la época.

En 1972, la universidad de Stanford creo otro famoso catbot con el nombre de Perry. Era a versión modificada de Elizza y venía con una actitud emocional. Estos dos primeros chatbots se conectaron a través de ARPANET y se comunicaron entre sí.

Después del éxito de los chatbots iniciales, Jabberwacky se lanzó como el primer chatbot que imitaba la voz humana. El objetivo principal de su desarrollo era permitir que el chatbot pasara la prueba de Turing. (ChatBot, 2019)

El Dr. Sbaitso se desarrolló en 1992 y vino con una interfaz de usuario e imitó las respuestas de un psicólogo.

ALICE es creada en 1995, fue el chatbot más famoso de su época ya que tenía la capacidad de comunicarse con los humanos de manera más eficiente debido a sus funciones experimentales. Otro hecho interesante es que Alice tambien fue la inspiración para Siri de APPLE. Además de hallar el camino para futuras innovaciones en ciencias de la computación, fue la inspiración de muchas películas de ciencia ficción de la época.

La revolución de los chatbots se produjo con el lanzamiento del asistente personal exclusivo de APPLE Siri, presenta una amplia gama de comandos de usuario, que incluyen la programación de eventos, la verificación de información básica, el manejo de la configuración del dispositivo, la realización de acciones telefónicas recordatorios, la navegación por internet, la navegación por áreas y las posibilidades de interactuar con las aplicaciones de IOS.

Watson de IBM se expandió aún más en la interpretación a través del juego de procesamiento de lenguaje natural.

Amazon lanzó Alexa y Google desarrollo, su propio agente de dialogo “dialogDlow” que ofrece capacidades de procesamiento de lenguaje natural. El agente de Google tambien ofrece una plataforma de integración de una ventana para trabajar con marcos de Facebook, Twitter, Cortana, Skype, Slack, Alexa y muchos más.

# Avance

Nuestra interfaz de usuario web de Amazon Lex de muestra, conocida como la interfaz de usuario de chatbot, ya hace la mayor parte del trabajo pesado asociado con la provisión de un cliente web con todas las funciones para los chatbots de Amazon Lex. Puede aprovechar rápidamente sus funciones y minimizar el tiempo de rentabilidad de sus aplicaciones impulsadas por chatbot.

La interfaz de usuario del chatbot admite las siguientes funciones:

* Funciona con los bots Lex o Lex V2
* Interfaz de usuario receptiva lista para dispositivos móviles con pantalla completa o modos de widget integrables
* Soporte completo para voz y texto, con la capacidad de alternar sin problemas entre ellos
* Funciones de voz que incluyen detección automática de silencio, transcripciones, grabación y reproducción de audio, y la capacidad de interrumpir la reproducción de respuesta de Amazon Lex
* Soporte de [tarjetas de respuesta](https://docs.aws.amazon.com/lex/latest/dg/ex-resp-card.html) *tanto* para *texto como para voz*
* Capacidad para interactuar mediante programación con la interfaz de usuario del chatbot desde el sitio de alojamiento
* Varias opciones de implementación
* Web accesible a través de Amazon Cloudfront
* Inicio de sesión de usuario totalmente integrado a través del grupo de usuarios de Amazon Cognito: token de usuario al que puede acceder el bot Lex como atributo de sesión. El inicio de sesión puede ser opcional o obligatorio.
* Soporte de Markdown para texto / imágenes / video enriquecidos, etc.
* Soporte para botones en los que se puede hacer clic en las tarjetas de respuesta Lex
* Botones opcionales 'Pulgar hacia arriba' y 'Pulgar hacia abajo': envía un mensaje de respuesta al bot Lex
* Botón 'Ayuda' opcional en la barra de título: envía un mensaje de ayuda al bot de Lex
* Reenviar cualquier mensaje anterior

Transferir al chat en vivo de Amazon Connect: permite a los usuarios de la interfaz de usuario web de Lex solicitar y realizar conversaciones de chat en vivo con un agente humano mediante Amazon Connect. Aquí hay un breve video de demostración. Para obtener más información sobre cómo configurar y usar esta función, consulte el archivo README de GitHub.

## Código de Implementación

import { ComponentFixture, TestBed } from '@angular/core/testing';

import { AuthComponent } from './auth.component';

describe('AuthComponent', () => {

  let component: AuthComponent;

  let fixture: ComponentFixture<AuthComponent>;

  beforeEach(async () => {

    await TestBed.configureTestingModule({

      declarations: [ AuthComponent ]

    })

    .compileComponents();

  });

  beforeEach(() => {

    fixture = TestBed.createComponent(AuthComponent);

    component = fixture.componentInstance;

    fixture.detectChanges();

  });

  it('should create', () => {

    expect(component).toBeTruthy();

  });

});

import { Component, OnInit } from '@angular/core';

@Component({

  selector: 'app-auth',

  templateUrl: './auth.component.html',

  styleUrls: ['./auth.component.scss']

})

export class AuthComponent implements OnInit {

  constructor() { }

  ngOnInit(): void {

  }

}

import { ComponentFixture, TestBed } from '@angular/core/testing';

import { HomeComponent } from './home.component';

describe('HomeComponent', () => {

  let component: HomeComponent;

  let fixture: ComponentFixture<HomeComponent>;

  beforeEach(async () => {

    await TestBed.configureTestingModule({

      declarations: [ HomeComponent ]

    })

    .compileComponents();

  });

  beforeEach(() => {

    fixture = TestBed.createComponent(HomeComponent);

    component = fixture.componentInstance;

    fixture.detectChanges();

  });

  it('should create', () => {

    expect(component).toBeTruthy();

  });

});

import { Component, OnInit } from '@angular/core';

@Component({

  selector: 'app-home',

  templateUrl: './home.component.html',

  styleUrls: ['./home.component.scss']

})

export class HomeComponent implements OnInit {

  constructor() { }

  ngOnInit(): void {

  }

}

import { NgModule } from '@angular/core';

import { RouterModule, Routes } from '@angular/router';

import { AuthComponent } from 'src/app/auth/auth.component';

import { HomeComponent } from 'src/app/home/home.component';

const routes: Routes = [

  {

    path: 'login',

    component: AuthComponent

  },

  {

    path: 'home',

    component: HomeComponent

  },

  {

    path: '',

    redirectTo: '/login',

    pathMatch: 'full'

  },

  { path: '\*\*', component: AuthComponent }

];

@NgModule({

  imports: [RouterModule.forRoot(routes)],

  exports: [RouterModule]

})

export class AppRoutingModule { }

import { TestBed } from '@angular/core/testing';

import { RouterTestingModule } from '@angular/router/testing';

import { AppComponent } from './app.component';

describe('AppComponent', () => {

  beforeEach(async () => {

    await TestBed.configureTestingModule({

      imports: [

        RouterTestingModule

      ],

      declarations: [

        AppComponent

      ],

    }).compileComponents();

  });

  it('should create the app', () => {

    const fixture = TestBed.createComponent(AppComponent);

    const app = fixture.componentInstance;

    expect(app).toBeTruthy();

  });

  it(`should have as title 'myBot'`, () => {

    const fixture = TestBed.createComponent(AppComponent);

    const app = fixture.componentInstance;

    expect(app.title).toEqual('myBot');

  });

  it('should render title', () => {

    const fixture = TestBed.createComponent(AppComponent);

    fixture.detectChanges();

    const compiled = fixture.nativeElement as HTMLElement;

    expect(compiled.querySelector('.content span')?.textContent).toContain('myBot app is running!');

  });

});

import { Component } from '@angular/core';

@Component({

  selector: 'app-root',

  templateUrl: './app.component.html',

  styleUrls: ['./app.component.scss']

})

export class AppComponent {

  title = 'myBot';

}

import { NgModule } from '@angular/core';

import { BrowserModule } from '@angular/platform-browser';

import { AppRoutingModule } from './app-routing.module';

import { AppComponent } from './app.component';

import { AuthComponent } from './auth/auth.component';

import { HomeComponent } from './home/home.component';

import {AmplifyAngularModule, AmplifyService} from 'aws-amplify-angular'

@NgModule({

  declarations: [

    AppComponent,

    AuthComponent,

    HomeComponent

  ],

  imports: [

    BrowserModule,

    AppRoutingModule,

    AmplifyAngularModule

  ],

  providers: [AmplifyService],

  bootstrap: [AppComponent]

})

export class AppModule { }

export const environment = {

  production: true

};

const awsmobile = {

    "aws\_project\_region": "us-east-1",

    "aws\_cognito\_identity\_pool\_id": "us-east-1:827a9b6a-70a5-4294-83f8-b4a479ecad8c",

    "aws\_cognito\_region": "us-east-1",

    "aws\_user\_pools\_id": "us-east-1\_Njqd8G2yd",

    "aws\_user\_pools\_web\_client\_id": "5ejd9kutv5fthg5953of9ojlds",

    "oauth": {},

    "aws\_cognito\_username\_attributes": [

        "EMAIL"

    ],

    "aws\_cognito\_social\_providers": [],

    "aws\_cognito\_signup\_attributes": [

        "EMAIL"

    ],

    "aws\_cognito\_mfa\_configuration": "OFF",

    "aws\_cognito\_mfa\_types": [

        "SMS"

    ],

    "aws\_cognito\_password\_protection\_settings": {

        "passwordPolicyMinLength": 8,

        "passwordPolicyCharacters": []

    },

    "aws\_cognito\_verification\_mechanisms": [

        "EMAIL"

    ]

};

export default awsmobile;

const awsmobile = {

    "aws\_project\_region": "us-east-1",

    "aws\_cognito\_identity\_pool\_id": "us-east-1:827a9b6a-70a5-4294-83f8-b4a479ecad8c",

    "aws\_cognito\_region": "us-east-1",

    "aws\_user\_pools\_id": "us-east-1\_Njqd8G2yd",

    "aws\_user\_pools\_web\_client\_id": "5ejd9kutv5fthg5953of9ojlds",

    "oauth": {},

    "aws\_cognito\_username\_attributes": [

        "EMAIL"

    ],

    "aws\_cognito\_social\_providers": [],

    "aws\_cognito\_signup\_attributes": [

        "EMAIL"

    ],

    "aws\_cognito\_mfa\_configuration": "OFF",

    "aws\_cognito\_mfa\_types": [

        "SMS"

    ],

    "aws\_cognito\_password\_protection\_settings": {

        "passwordPolicyMinLength": 8,

        "passwordPolicyCharacters": []

    },

    "aws\_cognito\_verification\_mechanisms": [

        "EMAIL"

    ]

};

export default awsmobile;

import { enableProdMode } from '@angular/core';

import { platformBrowserDynamic } from '@angular/platform-browser-dynamic';

import { AppModule } from './app/app.module';

import { environment } from './environments/environment';

import Amplify from 'aws-amplify';

import amplify from './aws-exports';

Amplify.configure(amplify);

Amplify.configure({

  Interactions: {

    bots: {

      "ClimaBot":{

        "name": "ClimaBot",

        "alias": "$LASTEST",

        "region": "us-east-1",

      },

    }

  }

});

if (environment.production) {

  enableProdMode();

}

platformBrowserDynamic().bootstrapModule(AppModule)

  .catch(err => console.error(err));

(window as any).global = window;

import 'zone.js';

import 'zone.js/testing';

import { getTestBed } from '@angular/core/testing';

import {

  BrowserDynamicTestingModule,

  platformBrowserDynamicTesting

} from '@angular/platform-browser-dynamic/testing';

declare const require: {

  context(path: string, deep?: boolean, filter?: RegExp): {

    keys(): string[];

    <T>(id: string): T;

  };

};

// First, initialize the Angular testing environment.

getTestBed().initTestEnvironment(

  BrowserDynamicTestingModule,

  platformBrowserDynamicTesting(),

);

// Then we find all the tests.

const context = require.context('./', true, /\.spec\.ts$/);

// And load the modules.

context.keys().map(context);

# Aplicación

Los chatbots ahora son lo suficientemente inteligentes como para analizar información y tomar decisiones clave por su cuenta. Convirtiéndose rápidamente en una parte integral de nuestras vidas, los chatbots de la nueva era están permitiendo a las personas operar de manera efectiva y eficiente.

Las empresas líderes ahora están utilizando chatbots para aumentar su eficiencia. Se utilizan ampliamente en diversas industrias, como el comercio minorista, el entretenimiento y las redes sociales, entre muchas otras, no solo para adquirir información sino también para actuar de acuerdo con la intención.

Los chatbots han penetrado mucho en todas las operaciones que van desde responder consultas, obtener patrones para aconsejar a los usuarios.

Una encuesta reciente revela que alrededor del 80% de las empresas en todo el mundo planean implementar un sistema de chatbot funcional para 2020. Como las empresas orientadas al consumidor tienen que ver con la experiencia, tener un sistema de chatbot integrado con su infraestructura actual solo aumentará su ventaja.

Cuando se trata de las innovaciones en el desarrollo de chatbot, el cielo es el límite. Debido a la infusión de Inteligencia Artificial, las posibilidades de beneficios comerciales que ofrecen los chatbots se han ampliado aún más.

# Problemática Para Resolver

Ahora ha configurado la interfaz de usuario de chatbot mejorada. Se recomienda que se autentique para un entorno de producción. Esta publicación utiliza Amazon Cognito para agregar un proveedor de identidad social (Google) a su grupo de usuarios. Para obtener instrucciones, consulte Agregar proveedores de identidad social a un grupo de usuarios.

En el modo autenticado (con Google), el iframe cambia a su calendario de Google para ayudarlo a programar un día de entrega. El gráfico de área todavía cambia al gráfico de rosas vendidas.

Esta poderosa herramienta permite que aparezca contenido de varias partes del sitio web o de Internet al interactuar con el bot. Esto también permite que el bot reconozca si está autenticado o no y adapte su experiencia de navegación.

El uso de nuestro programa es otorgar una guía al cliente para que puedan realizar sus compras de la mejor manera posible, además de estar disponible todo el tiempo con solo realizar el algoritmo necesario para brindar esta atención por medio de preguntas, respuestas y búsquedas.

Por ejemplo**, de seguro has realizado una compra en línea y te has topado con alguna pregunta ya sea del producto o el servicio de envío.** Es normal, los clientes tienen dudas al momento de realizar su compra, o al considerar hacerla. **Resolver las dudas del cliente es una de las principales acciones que hace posible la conversión,** o sea que tus clientes compren contigo.

Diagrama

Descripción generada automáticamente

![Diagrama

Descripción generada automáticamente]()

# Conclusiones

Esta publicación demuestra cómo mejorar el bot Order Flowers con una función Lambda que analiza su token JWT y extrae la información relevante. Si está autenticado a través de Google, el bot extrae información como su nombre y dirección de correo electrónico, y muestra su calendario de Google para ayudarlo a programar su fecha de entrega. La función también verifica que la firma del token JWT sea válida.

La interfaz de usuario del chatbot en esta publicación se basa en el proyecto de código abierto aws-lex-web-ui . Para obtener más información, consulte el [repositorio de GitHub](https://github.com/aws-samples/aws-lex-web-ui) .

# Bibliografía

ChatBot, P. (21 de Agosto de 2019). *Planeta Chatbot*. Obtenido de La línea de tiempo de la historia de los Chatbots: antes, ahora y mañana: https://planetachatbot.com/linea-tiempo-historia-de-chatbots-antes-ahora-y-manana/

FrutoSeco. (15 de Octubre de 2003). *Bot conversacional*. Obtenido de Wikipedia: https://es.wikipedia.org/wiki/Bot\_conversacional

Mohamed Khalil, B. P. (24 de Julio de 2020). *Mejorando su experiencia de chatbot con la navegación web*. Obtenido de Blog de aprendizaje automático de AWS: https://aws.amazon.com/es/blogs/machine-learning/enhancing-your-browsing-experience-with-amazon-lex/

Strahan, O. A. (26 de Enero de 2018). *Implemente una interfaz de usuario web para su chatbot* . Obtenido de Blog de aprendizaje automático de AWS: https://aws.amazon.com/es/blogs/machine-learning/deploy-a-web-ui-for-your-chatbot/