***K.T.S.P Mandal’s***

***K.M.C College Khopoli***

***DEPARTMENT OF COMPUTER SCIENCE***

***KHOPOLI–410203***

*A Project Report*

*On*

***ChatBot***

*Submitted To*

***University of Mumbai***

*By*

***Miss Neha Narendra Ghonge***

*Under Guidance Of*

***Prof. Ashwini Gharat***

*2021-2022*

***K.T.S.P MANDAL’S***

***KMC COLLEGE KHOPOLI***

***DEPARTMENT OF COMPUTER SCIENCE***

***CERTIFICATE***

*This is to certify that* ***Neha Narendra Ghonge***  *has successfully completed the project on the topic of*

*“****ChatBot****” in Sem-II.During the academic year 2021- 2022 as per the guidelines issued by* ***University of Mumbai****.*

***Teacher’s HOD’s Examiner’s***

***Signature Signature Signature***

***Date: Date:***

***Acknowledgment***

*In the accomplishment of this project successfully, many people have best owned upon me their blessings and the heart pledged support, this time I am utilizing to thank all the people who have been concerned with this project.*

*Primarily, I would thank god for being able to complete this project with success. Then I would like to thank my principal* ***Prof****,****Dr.Pratap Patil*** *and my project teacher* ***Prof. Ashwini Gharat*** *whose valuable guidance has been the ones that helped me patch this project and make it full proof success. Her suggestions and her instructions have served as the major contributor towards the completion of the project. I am also thankful to my head of department* ***Prof. Dhanashree Pawar*** *who encourage me and gave me moral support during my project.*

*Technologies Used*

***Software requirement:***

*Software requirements for this system are as listed follows:*

* Frontend : Xml
* Backend : Java
* Operating System : Windows

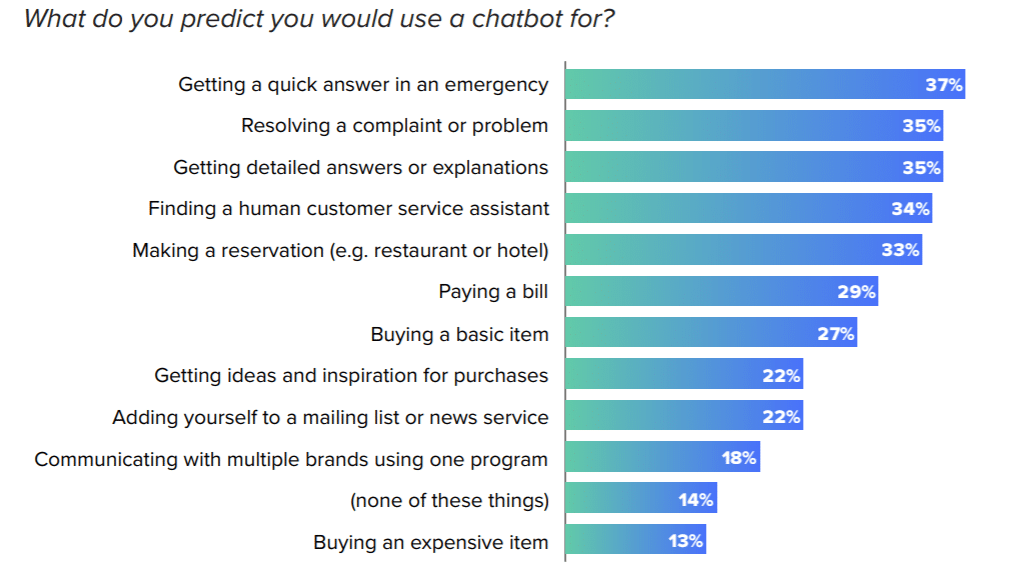
***Hardware requirement:***

*Minimum Hardware requirements for these system are listed below:*

* C.P.U:- RMD Ryzen.
* R.A.M:- 8 Giga Bytes.
* Hard Disk:- 40 Giga Bytes.
* Type Of System : Single User

*Description Of A Project:-*

*Chatbot:-*

We have seen many apps and websites in which we will get to see a chatbot where we can chat along with the chatbot and can easily get solutions for our questions answered from the chatbot.

**What we are going to build in this project?**

We will be building a simple application in which we will be building a simple chatbot where we can ask a question to our bot and the bot will answer our questions. Below is the video in which we will get to see what we are going to build in this article. A sample video is given below to get an idea about what we are going to do in this article. Note that we are going to implement this project using the Java language.

**Step by Step Implementation**

***Step 1: Create a New Project***

To create a new project in Android Studio please refer to How to Create/Start a New Project in Android Studio. Note that select Java as the programming language.

***Step 2: Add the below dependency in your build.gradle file***

Navigate to the app > Gradle Scripts > build.gradle file and add the below dependency to it in the dependencies section.

*implementation ‘com.android.volley:volley:1.1.1’*

After adding this dependency sync your project and now move towards the AndroidManifest.xml part.

***Step 3: Adding permissions to the internet in the AndroidManifest.xml file***

<!--permissions for internet-->

<uses-permission android:name="android.permission.INTERNET" />

<uses-permission android:name="android.permission.ACCESS\_NETWORK\_STATE" />

***Step 4: Working with the activity\_main.xml file***

Navigate to the app > res > layout > activity\_main.xml

***Step 5: Creating a Modal class for storing our messages***

Navigate to the app > java > your app’s package name > Right-click on it > New > Java class and name it as MessageModal

***Step 6: Creating a layout file for user messages***

Icons used in this file are present in the drawable folder. Navigate to the app > res > layout > Right-click on it > New > layout resource file and name the file as user\_msg

***Step 7: Create a layout file for bot messages***

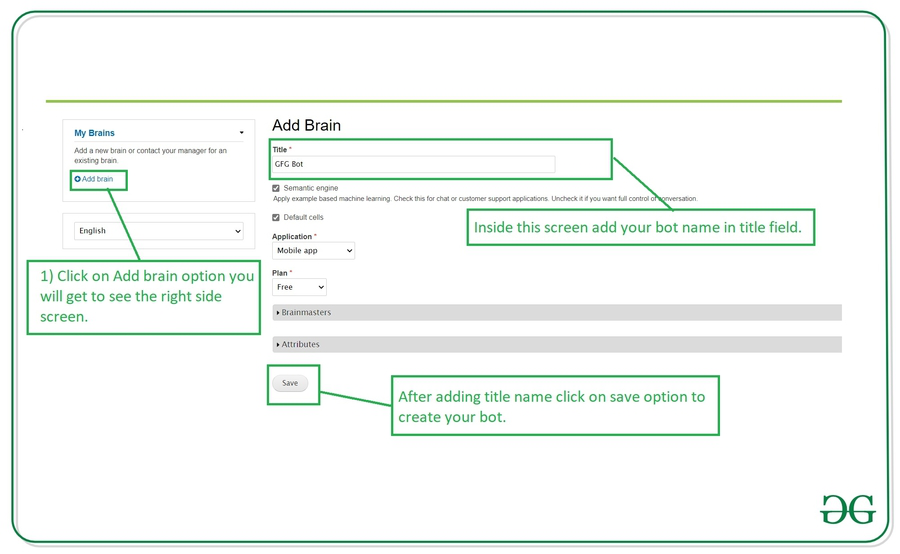
Icons used in this file are present in the drawable folder. Navigate to the app > res > layout > Right-click on it > New > layout resource file and name the file as bot\_msg

***Step 8: Working with the Adapter class***

For setting data to our items of Chat RecyclerView we have to create an Adapter class. Navigate to the app > java > your app’s package name > Right-click on it > New > Java class and name your class as MessageRVAdapter

***Step 9: Generating API key for using the chatbot service***

Go to Brainshop.ai generate your simple account with your username and password. Simply create your account on this website. After creating a new account you will get to see the below screen. After creating your account you have to request a new password from the request password option and enter your email address. After adding your email address you have to add the password to your account. Now we are good to go to generate our API key.

******

Follow the above steps to Generate a new brain for your chatbot. After generating your bot now we will get the API URL for this brain. Navigate to the settings tab inside your created brain you will get to see your bot details as shown below. Note: Now we will be using this API URL only inside the MainActivity.java file.

******

***Step 10: Working with the MainActivity.java file***

Go to the MainActivity.java file and type code there.

***Step 11: Running the code:-***

Click on run button to run the code.

***Coding***

**Layout Files:-**

**Activity\_main.xml**

<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

xmlns:tools="http://schemas.android.com/tools"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

tools:context=".MainActivity">

<!--recycler view to display our chats-->

<androidx.recyclerview.widget.RecyclerView

android:id="@+id/idRVChats"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:layout\_above="@id/idLLMessage" />

<LinearLayout

android:id="@+id/idLLMessage"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_alignParentBottom="true"

android:orientation="horizontal"

android:weightSum="5">

<!--edit text to enter message-->

<EditText

android:id="@+id/idEdtMessage"

android:layout\_width="0dp"

android:layout\_height="wrap\_content"

android:layout\_weight="4"

android:hint="Enter Message"

/>

<!--button to send message-->

<ImageButton

android:id="@+id/idIBSend"

android:layout\_width="0dp"

android:layout\_height="wrap\_content"

android:layout\_gravity="center\_vertical"

android:layout\_weight="1"

android:background="@color/purple\_200"

android:src="@drawable/ic\_send"

android:tint="@color/white"

android:contentDescription="TODO" />

</LinearLayout>

</RelativeLayout>

**Bot\_msg.xml**

<?xml version="1.0" encoding="utf-8"?>

<androidx.cardview.widget.CardView

xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_gravity="start"

android:layout\_margin="5dp"

android:elevation="8dp"

app:cardCornerRadius="8dp">

<LinearLayout

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:orientation="horizontal">

<!--below widget is for image of bot-->

<ImageView

android:layout\_width="50dp"

android:layout\_height="50dp"

android:layout\_margin="10dp"

android:src="@drawable/ic\_bot" />

<!--below widget is for

displaying message of bot-->

<TextView

android:id="@+id/idTVBot"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_gravity="center\_vertical"

android:layout\_margin="5dp"

android:padding="3dp"

android:text="Bot message"

android:textColor="@color/black" />

</LinearLayout>

</androidx.cardview.widget.CardView>

**User\_msg.xml**

<?xml version="1.0" encoding="utf-8"?>

<androidx.cardview.widget.CardView

xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_gravity="end"

android:layout\_margin="5dp"

android:elevation="8dp"

app:cardCornerRadius="8dp">

<LinearLayout

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:orientation="horizontal">

<!--text view for displaying user message-->

<TextView

android:id="@+id/idTVUser"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_gravity="center\_vertical"

android:layout\_margin="5dp"

android:padding="3dp"

android:text="User message"

android:textColor="@color/black" />

<!--we are displaying user icon-->

<ImageView

android:layout\_width="50dp"

android:layout\_height="50dp"

android:layout\_margin="10dp"

android:src="@mipmap/ic\_launcher\_round"

app:srcCompat="@mipmap/ic\_launcher\_round" />

</LinearLayout>

</androidx.cardview.widget.CardView>

**Java files:-**

**MainActivity.java**

package com.example.chatbot;

import android.os.Bundle;

import android.view.View;

import android.widget.EditText;

import android.widget.ImageButton;

import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;

import androidx.recyclerview.widget.LinearLayoutManager;

import androidx.recyclerview.widget.RecyclerView;

import com.android.volley.Request;

import com.android.volley.RequestQueue;

import com.android.volley.Response;

import com.android.volley.VolleyError;

import com.android.volley.toolbox.JsonObjectRequest;

import com.android.volley.toolbox.Volley;

import org.json.JSONException;

import org.json.JSONObject;

import java.util.ArrayList;

public class MainActivity extends AppCompatActivity {

// creating variables for our

// widgets in xml file.

private RecyclerView chatsRV;

private ImageButton sendMsgIB;

private EditText userMsgEdt;

private final String USER\_KEY = "user";

private final String BOT\_KEY = "bot";

// creating a variable for

// our volley request queue.

private RequestQueue mRequestQueue;

// creating a variable for array list and adapter class.

private ArrayList<MessageModal> messageModalArrayList;

private MessageRVAdapter messageRVAdapter;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

// on below line we are initializing all our views.

chatsRV = findViewById(R.id.idRVChats);

sendMsgIB = findViewById(R.id.idIBSend);

userMsgEdt = findViewById(R.id.idEdtMessage);

// below line is to initialize our request queue.

mRequestQueue = Volley.newRequestQueue(MainActivity.this);

mRequestQueue.getCache().clear();

// creating a new array list

messageModalArrayList = new ArrayList<>();

// adding on click listener for send message button.

sendMsgIB.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

// checking if the message entered

// by user is empty or not.

if (userMsgEdt.getText().toString().isEmpty()) {

// if the edit text is empty display a toast message.

Toast.makeText(MainActivity.this, "Please enter your message..", Toast.LENGTH\_SHORT).show();

return;

}

// calling a method to send message

// to our bot to get response.

sendMessage(userMsgEdt.getText().toString());

// below line we are setting text in our edit text as empty

userMsgEdt.setText("");

}

});

// on below line we are initialing our adapter class and passing our array list to it.

messageRVAdapter = new MessageRVAdapter(messageModalArrayList, this);

// below line we are creating a variable for our linear layout manager.

LinearLayoutManager linearLayoutManager = new LinearLayoutManager(MainActivity.this, RecyclerView.VERTICAL, false);

// below line is to set layout

// manager to our recycler view.

chatsRV.setLayoutManager(linearLayoutManager);

// below line we are setting

// adapter to our recycler view.

chatsRV.setAdapter(messageRVAdapter);

}

private void sendMessage(String userMsg) {

// below line is to pass message to our

// array list which is entered by the user.

messageModalArrayList.add(new MessageModal(userMsg, USER\_KEY));

messageRVAdapter.notifyDataSetChanged();

// url for our brain

// make sure to add mshape for uid.

// make sure to add your url.

String url = "\n" +

"http://api.brainshop.ai/get?bid=165785&key=HupKj99OnaJOuur4&uid=[uid]&msg=[msg]" + userMsg;

// creating a variable for our request queue.

RequestQueue queue = Volley.newRequestQueue(MainActivity.this);

// on below line we are making a json object request for a get request and passing our url .

JsonObjectRequest jsonObjectRequest = new JsonObjectRequest(Request.Method.GET, url, null, new Response.Listener<JSONObject>() {

@Override

public void onResponse(JSONObject response) {

try {

// in on response method we are extracting data

// from json response and adding this response to our array list.

String botResponse = response.getString("cnt");

messageModalArrayList.add(new MessageModal(botResponse, BOT\_KEY));

// notifying our adapter as data changed.

messageRVAdapter.notifyDataSetChanged();

} catch (JSONException e) {

e.printStackTrace();

// handling error response from bot.

messageModalArrayList.add(new MessageModal("No response", BOT\_KEY));

messageRVAdapter.notifyDataSetChanged();

}

}

}, new Response.ErrorListener() {

@Override

public void onErrorResponse(VolleyError error) {

// error handling.

messageModalArrayList.add(new MessageModal("hi", BOT\_KEY));

Toast.makeText(MainActivity.this, "No response from the bot..", Toast.LENGTH\_SHORT).show();

}

});

// at last adding json object

// request to our queue.

queue.add(jsonObjectRequest);

}

}

**MessageModal.java**

package com.example.chatbot;

public class MessageModal {

// string to store our message and sender

private String message;

private String sender;

// constructor.

public MessageModal(String message, String sender) {

this.message = message;

this.sender = sender;

}

// getter and setter methods.

public String getMessage() {

return message;

}

public void setMessage(String message) {

this.message = message;

}

public String getSender() {

return sender;

}

public void setSender(String sender) {

this.sender = sender;

}

}

**ChatsModal.java**

package com.example.chatbot;

public class MessageModal {

// string to store our message and sender

private String message;

private String sender;

// constructor.

public MessageModal(String message, String sender) {

this.message = message;

this.sender = sender;

}

// getter and setter methods.

public String getMessage() {

return message;

}

public void setMessage(String message) {

this.message = message;

}

public String getSender() {

return sender;

}

public void setSender(String sender) {

this.sender = sender;

}

}

**MessageRVAdapter.java**

package com.example.chatbot;

import android.content.Context;

import android.view.LayoutInflater;

import android.view.View;

import android.view.ViewGroup;

import android.widget.TextView;

import androidx.annotation.NonNull;

import androidx.recyclerview.widget.RecyclerView;

import java.util.ArrayList;

public class MessageRVAdapter extends RecyclerView.Adapter {

// variable for our array list and context.

private ArrayList<MessageModal> messageModalArrayList;

private Context context;

// constructor class.

public MessageRVAdapter(ArrayList<MessageModal> messageModalArrayList, Context context) {

this.messageModalArrayList = messageModalArrayList;

this.context = context;

}

@NonNull

@Override

public RecyclerView.ViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {

View view;

// below code is to switch our

// layout type along with view holder.

switch (viewType) {

case 0:

// below line we are inflating user message layout.

view = LayoutInflater.from(parent.getContext()).inflate(R.layout.user\_msg, parent, false);

return new UserViewHolder(view);

case 1:

// below line we are inflating bot message layout.

view = LayoutInflater.from(parent.getContext()).inflate(R.layout.bot\_msg, parent, false);

return new BotViewHolder(view);

}

return null;

}

@Override

public void onBindViewHolder(@NonNull RecyclerView.ViewHolder holder, int position) {

// this method is use to set data to our layout file.

MessageModal modal = messageModalArrayList.get(position);

switch (modal.getSender()) {

case "user":

// below line is to set the text to our text view of user layout

((UserViewHolder) holder).userTV.setText(modal.getMessage());

break;

case "bot":

// below line is to set the text to our text view of bot layout

((BotViewHolder) holder).botTV.setText(modal.getMessage());

break;

}

}

@Override

public int getItemCount() {

// return the size of array list

return messageModalArrayList.size();

}

@Override

public int getItemViewType(int position) {

// below line of code is to set position.

switch (messageModalArrayList.get(position).getSender()) {

case "user":

return 0;

case "bot":

return 1;

default:

return -1;

}

}

public static class UserViewHolder extends RecyclerView.ViewHolder {

// creating a variable

// for our text view.

TextView userTV;

public UserViewHolder(@NonNull View itemView) {

super(itemView);

// initializing with id.

userTV = itemView.findViewById(R.id.idTVUser);

}

}

public static class BotViewHolder extends RecyclerView.ViewHolder {

// creating a variable

// for our text view.

TextView botTV;

public BotViewHolder(@NonNull View itemView) {

super(itemView);

// initializing with id.

botTV = itemView.findViewById(R.id.idTVBot);

}

}

}

**RetroApi.java**

package com.example.chatbot;

import retrofit2.Call;

import retrofit2.http.GET;

import retrofit2.http.Url;

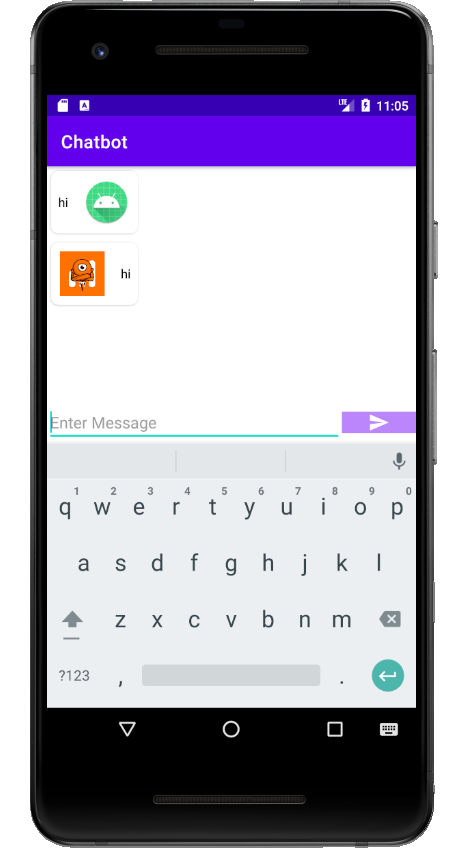
public interface RetrofitAPI {

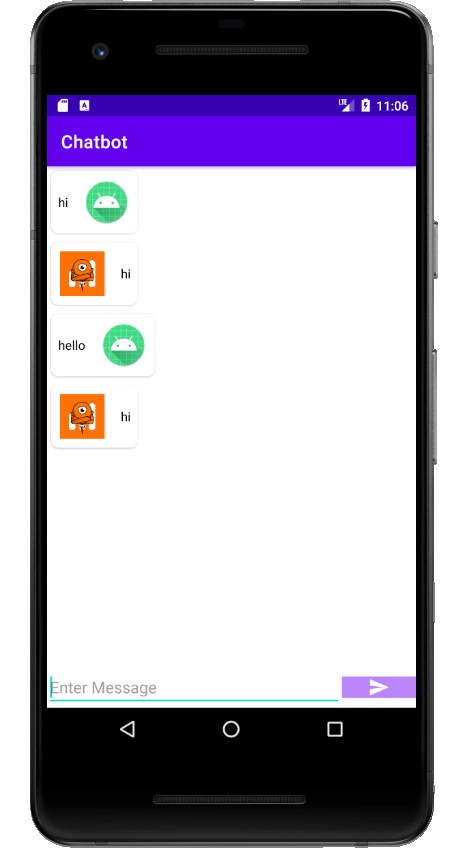
@GET

Call<MessageModal> getMessage(@Url String url);

}

Output(ScreenLayouts)





*Conclusion And Future use*

**Conclusion:-**

Now our chatbot is ready.so One of the greatest advantages is that Chatbots are available 24/7 to help customers. Plus, they respond quickly to every question they receive. This guarantees your customers will always be able to solve their problems, no matter what time of day.

**Future use:-**

Here are some use cases for different industries and examples of how they have been deployed.

**Financial services**: 50% of financial services companies working on a chatbot project. By using the chatbot services, customers can perform tasks such as checking account balances, transferring money to other accounts, and alerting the bank to lost credit and debit cards.

**Real estate:** Real estate is a relationship-oriented industry by nature. Chatbots can’t replace that client-agent relationship, but they can replace agents in answering general real estate questions. Holmes is a real estate chatbot that uses AI to answer customer queries in real-time.

**Travel:** The travel and tourism industry witnesses more international customers willing to do business. One of the problems with the travel business is catering 24×7 support. Chatbot, allows you to handle customer queries and generate leads around the clock. Hello Hipmunk is a completely free and helpful bot that does all the work for you.

**Food services:** The food industry is one of the industries that has experienced the most success with chatbots. Pizza chains have been among the most prolific adopters of chatbot technology. Domino’s is the first major brand to launch a chatbot on Facebook Messenger and generates nearly $5bn a year.

**Airlines:** Many airlines are embracing chatbots to help their customers answer common questions and obtain basic information about their flights. Aeroméxico launched its chatbot to serve 1,000 customers per day.

*References*

* <https://www.geeksforgeeks.org/how-to-create-a-chatbot-in-android-with-brainshop-api/>
* <https://brainshop.ai/chatbot>
* <https://nordicapis.com/15-intelligent-chatbot-apis/>