**Front-end Source code**

**JSON editor:**

{

"interactionModel": {

"languageModel": {

"invocationName": "dag",

"intents": [

{

"name": "AMAZON.FallbackIntent",

"samples": []

},

{

"name": "AMAZON.CancelIntent",

"samples": []

},

{

"name": "AMAZON.HelpIntent",

"samples": []

},

{

"name": "AMAZON.StopIntent",

"samples": []

},

{

"name": "AMAZON.NavigateHomeIntent",

"samples": []

},

{

"name": "RandomTankCapacity",

"slots": [],

"samples": [

"tell me the random capacity",

"tell me the random tank capacity",

"give me the tank capacity",

"Random tank capacity"

]

},

{

"name": "VehicleTankCapacity",

"slots": [

{

"name": "vehicle",

"type": "LIST\_OF\_VEHICLES"

}

],

"samples": [

"get the capacity from {vehicle}",

"get the tank capacity from {vehicle} ",

"get the tank capacity of {vehicle}",

"i need the tank capacity of {vehicle} ",

"what is {vehicle} tank capacity",

"tell me the tank capacity of {vehicle}"

]

}

],

"types": [

{

"name": "LIST\_OF\_VEHICLES",

"values": [

{

"name": {

"value": "Dio"

}

},

{

"name": {

"value": "Pleasure"

}

},

{

"name": {

"value": "Activa"

}

},

{

"name": {

"value": "YamahaFz"

}

},

{

"name": {

"value": "Pulsar"

}

},

{

"name": {

"value": "TVSApache"

}

},

{

"name": {

"value": "YamahaCD100"

}

},

{

"name": {

"value": "HondaDreamYuga"

}

},

{

"name": {

"value": "Acitiva5G"

}

},

{

"name": {

"value": "Activa3G"

}

},

{

"name": {

"value": "HondaCliq"

}

},

{

"name": {

"value": "HondaAchiver"

}

},

{

"name": {

"value": "HondaSplender"

}

},

{

"name": {

"value": "HondaShine"

}

},

{

"name": {

"value": "Piaggio"

}

},

{

"name": {

"value": "TVS50"

}

},

{

"name": {

"value": "Royalenfiled"

}

},

{

"name": {

"value": "KtmDuke"

}

},

{

"name": {

"value": "HarleyDavidson"

}

},

{

"name": {

"value": "Vespa"

}

},

{

"name": {

"value": "YamahaRay"

}

},

{

"name": {

"value": "YamahaFascino"

}

},

{

"name": {

"value": "YamahaAlpha"

}

},

{

"name": {

"value": "TVSWego"

}

},

{

"name": {

"value": "TVSjupiter"

}

},

{

"name": {

"value": "SuzukiSwish"

}

},

{

"name": {

"value": "SuzukiLets"

}

},

{

"name": {

"value": "SuzukiAccess"

}

},

{

"name": {

"value": "MahindraRodeo"

}

},

{

"name": {

"value": "MahindraGusto"

}

},

{

"name": {

"value": "MahindraDuro"

}

},

{

"name": {

"value": "HondaDio"

}

},

{

"name": {

"value": "HondaAviator"

}

},

{

"name": {

"value": "hondaActive125"

}

},

{

"name": {

"value": "HeroPleasure"

}

},

{

"name": {

"value": "HeroMaestroEdge"

}

},

{

"name": {

"value": "HeroMaestro"

}

},

{

"name": {

"value": "HeroDuet"

}

}

]

}

]

}

}

}

**10.2 Back-end Source Code:**

**BUCKET BACKUP:**

console.log('Loading function');

const aws = require('aws-sdk');

const s3 = new aws.S3({

apiVersion: '2006-03-01'

});

exports.handler = async (event, context) => {

//console.log('Received event:', JSON.stringify(event, null, 2));

// Get the object from the event and show its content type

const bucket = event.Records[0].s3.bucket.name;

const key = decodeURIComponent(event.Records[0].s3.object.key.replace(/\+/g, ' '));

const params = {

Bucket: bucket,

Key: key,

};

try {

const {

ContentType

} = await s3.getObject(params).promise();

console.log('CONTENT TYPE:', ContentType);

// return ContentType;

var targetBucket = bucket + "-backup"; //bikebucket1-backup

console.log("copying" + key + "from" + bucket + "to" + targetBucket);

const paramsNew = {

CopySource: bucket + "/" + encodeURIComponent(key),

Bucket: targetBucket,

Key: key

};

constFinalOutput = await s3.copyObject(paramsNew).promise();

console.log("File" + key + "copied from" + bucket + "to" + targetBucket + "sucessfully");

return "File copied sucessfully";

} catch (err) {

console.log(err);

const message = `Error getting object ${key} from bucket ${bucket}. Make sure they exist and your bucket is in the same region as this function.`;

console.log(message);

throw new Error(message);

}

};

**index.js:**

const Alexa = require("ask-sdk");

const actions = require("./function");

const TankCapacities = {

HeroDuet: ["6Litres."],

HeroMaestro: ["5.3Litres."],

HeroMaestroEdge: ["6Litres."],

HeroPleasure: ["5Litres."],

HondaActive125: ["5.3Litres."],

HondaAviater: ["6Litres."],

HeroDio: ["6Litres."],

MahindraDuro: ["6.5Litres."],

MahindraGusto: ["6Litres."],

MahindraRodeo: ["6Litres."],

SuzukiAccess: ["6Litres."],

SuzukiLets: ["5.2Litres."],

SuzukiSwish: ["6Litres."],

TVSJupiter: ["5Litres."],

TVSWego: ["5Litres."],

YamahaAlpha: ["5.2Litres."],

YamahaFascino: ["5.2Litres."],

YamahaRay: ["5Litres."],

Vespa: ["5.2Litres."],

KTMDuke: ["6Litres."],

RoyalEnfield: ["5Litres."],

TVS50: ["5.3Litres."],

Piaggio: ["6Litres."],

HondaShine: ["6Litres."],

HondaSplender: ["6.5Litres."],

HondaAchiver: ["6Litres."],

HondaCliq: ["4.5Litres."],

Activa3G: ["6Litres."],

Activa5G: ["5.2Litres."],

HondaDreamyuga: ["6Litres."],

YamahaCD100: ["5Litres."],

TVSApache: ["6Litres."],

Pulsar: ["5.2Litres."],

YamahaFz: ["5.2Litres."],

Activa: ["6Litres."],

Dio: ["6Litres."],

Pleasure: ["5Litres."]

};

// Launch Request Handler -- When a skill is launched

const LaunchRequestHandler = {

canHandle(handlerInput) {

return handlerInput.requestEnvelope.request.type === "LaunchRequest";

},

handle(handlerInput) {

console.log("Launch Request Handler Called");

let speechText =

"Hello, I am DAG. You can ask me to read out tankcapacity for different vehicles.";

let repromptText =

"I did not receive any input. You can say, tell me the tankcapacity.";

return handlerInput.responseBuilder

.speak(speechText)

.reprompt(repromptText)

.getResponse();

}

};

// Handler for Random Tankcapacity

const RandomTankCapacity = {

canHandle(handlerInput) {

return (

handlerInput.requestEnvelope.request.type === "IntentRequest" &&

handlerInput.requestEnvelope.request.intent.name === "RandomTankCapacity"

);

},

handle(handlerInput) {

console.log("RandomTankCapacity intent handler called");

let getTankCapacity = actions.getTankCapacity(TankCapacities);

let vehicle = getTankCapacity[0];

let tankcapacity = getTankCapacity[1];

let cardTitle = "Quotation from " + vehicle;

let cardContent = tankcapacity;

let speechText = vehicle + " said " + tankcapacity;

return handlerInput.responseBuilder

.speak(speechText)

.withSimpleCard(cardTitle, cardContent)

.getResponse();

}

};

const VehicleTankCapacity = {

canHandle(handlerInput) {

return (

handlerInput.requestEnvelope.request.type === "IntentRequest" &&

handlerInput.requestEnvelope.request.intent.name === "VehicleTankCapacity"

);

},

handle(handlerInput) {

console.log("VehicleTankCapacity Intent handler called");

// Get the vehicle Name

let vehicle =

handlerInput.requestEnvelope.request.intent.slots.vehicle.value;

let getTankCapacity = actions.getTankCapacity(TankCapacities, vehicle);

if (!getTankCapacity) {

return UnhandledHandler.handle(handlerInput);

}

vehicle = getTankCapacity[0].toLowerCase();

let tankcapacity = getTankCapacity[1];

let cardTitle = "tankcapacity from " + vehicle;

let cardContent = tankcapacity;

let speechText = vehicle + " said " + tankcapacity;

return handlerInput.responseBuilder

.speak(speechText)

.withSimpleCard(cardTitle, cardContent)

.getResponse();

}

};

// Unhandled Requests

const UnhandledHandler = {

canHandle() {

return true;

},

handle(handlerInput, error) {

console.log(`Error Handler : ${error.message}`);

return handlerInput.responseBuilder

.speak(

"Sorry, I am unable to understand what you said. You can ask me to say a RandomTankCapacity and VehicleTankCapacity"

)

.getResponse();

}

};

exports.handler = Alexa.SkillBuilders.custom()

.addRequestHandlers(

LaunchRequestHandler,

RandomTankCapacity,

VehicleTankCapacity

)

.addErrorHandlers(UnhandledHandler)

.lambda();

**function.js:**

const myfunctions = {

// Take out the tank capacity and return vehicle as well as tank capacity

getTankCapacity: function(tankcapacity, vehicle) {

console.log("Getting into getTankCapacityFunction");

// Get random vehicle if vehicle is not defined

if (vehicle === undefined) {

var totalvehicle = Object.keys(tankcapacity).length;

var rand = Math.floor(Math.random() \* totalvehicle);

// random vehicle name

vehicle = Object.keys(tankcapacity)[rand];

}

// check the vehicle if it exists, and have a single vehicle name

switch (vehicle) {

case "HeroDuet":

vehicle = "HeroDuet";

break;

case "HeroMaestro":

vehicle = "HeroMaestro";

break;

case "HeroMaestroEdge":

vehicle = "HeroMaestroEdge";

break;

case "HeroPleasure":

vehicle = "HeroPleasure";

break;

case "HondaActive125":

vehicle = "HondaActive125";

break;

case "HondaAviater":

vehicle = "HondaAviater";

break;

case "Dio":

vehicle = "Dio";

break;

case "MahindraDuro":

vehicle = "MahindraDuro";

break;

case "MahindraGusto":

vehicle = "MahindraGusto";

break;

case "MahindraRodeo":

vehicle = "MahindraRodeo";

break;

case "SuzukiAccess":

vehicle = "SuzukiAccess";

break;

case "SuzukiLets":

vehicle = "SuzukiLets";

break;

case "SuzukiSwish":

vehicle = "SuzukiSwish";

break;

case "TVSJupiter":

vehicle = "TVSJupiter";

break;

case "YamahaAlpha":

vehicle = "YamahaAlpha";

break;

case "YamahaFascino":

vehicle = "YamahaFascino";

break;

case "YamahaRay":

vehicle = "YamahaRay";

break;

case "Vespa":

vehicle = "Vespa";

break;

case "KTMDuke":

vehicle = "KTMDuke";

break;

case "RoyalEnfield":

vehicle = "RoyalEnfield";

break;

case "TVS50":

vehicle = "TVS50";

break;

case "Piaggio":

vehicle = "Piaggio";

break;

case "HondaShine":

vehicle = "HondaShine";

break;

case "HondaSplender":

vehicle = "HondaSplender";

break;

case "HondaAchiver":

vehicle = "HondaAchiver";

break;

case "HondaCliq":

vehicle = "HondaCliq";

break;

case "Activa3G":

vehicle = "Activa3G";

break;

case "Activa5G":

vehicle = "Activa5G";

break;

case "Activa":

vehicle = "Activa";

break;

case "HondaDreamyuga":

vehicle = "HondaDreamyuga";

break;

case "YamahaCD100":

vehicle = "YamahaCD100";

break;

case "TVSApache":

vehicle = "TVSApache";

break;

case "YamahaFz":

vehicle = "YamahaFz";

break;

case "Pulsar":

vehicle = "Pulsar";

break;

default:

vehicle = "Unknown";

}

// Get total tank capacity for the vehicle from the vehicle object

var totaltankcapacity = tankcapacity[vehicle].length;

// Select a random tankcapacity

var randtankcapacity = Math.floor(Math.random() \* totaltankcapacity);

var tankcapacity1 = tankcapacity[vehicle][randtankcapacity];

console.log("Return vehicle and tankcapacity");

// return both the vehicle name and the tankcapacity as an array

return [vehicle, tankcapacity1];

}

};

module.exports = myfunctions;