**Code Backup – 14th November**

**Index.js**

// Lambda Function for Intent Fitness Alexa Skill by Martin Nolan

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

const invocationName = "intent fitness";

const MAX\_NUMBER = 13; // This is the max number of audio streams in the meditation section (14 sounds)

const MAX\_TIPS = 24; // This is the max number of tips in the tips array (25)

const MAX\_SHORT\_TIPS = 13; // This is the max number of tips that is displayed on visual output devices such as an Echo Show

// Array of tips that are read of at the end of a session

const tips = [

"to ensure good health, eat lightly, breathe deeply, live moderately, cultivate cheerfulness, and maintain an interest in life.",

"to keep the body in good health is a duty, otherwise we will not be able to keep the mind strong and clear.",

"do something today that your future self will thank you for.",

"exercising before going to bed makes your muscles burn more calories during the night.",

"a mid-day nap improves your memory and reduces the chances of developing heart diseases.",

"if you\’re feeling anxious and stressed, eat a melon. Melons help relieve anxiety and stress, plus they boost your metabolism.",

"drink two cups of cold water before a meal, as this boosts metabolism by up to 30%.",

"drinking a lot of water during the day helps you sleep better at night.",

"a cold shower can help relieve depression and also helps keep your skin and hair healthier.",

"the first day of the week defines your thinking patterns for the rest of the week. It\’s best to exercise on Mondays to ensure a healthy routine.",

"your body can stand almost anything. It\’s your mind that you have to convince.",

"fitness is like a relationship. You can\’t cheat and expect it to work.",

"do something today that your future self will thank you for.",

"life begins at the end of your comfort zone.",

"the difference between try, and triumph is a little \‘umph\’.",

"don\’t count the days, make the days count.",

"when you feel like quitting, think about why you started.",

"all progress takes place outside the comfort zone.",

"success starts with self-discipline.",

"the only bad workout is the one that didn\’t happen.",

"rome wasn\’t built in a day. Work hard and good results will come.",

"motivation is what gets you started. Habit is what keeps you going.",

"fitness is not about being better than someone else. It\’s about being better than you used to be.",

"the pain you feel today will be the strength you feel tomorrow.",

"age is no barrier. It\’s a limitation you put on your mind."

];

// Array of tips that are displayed to a user on visual output devices such as an Echo Show

const shortTips = [

"drinking a lot of water during the day helps you sleep better at night.",

"the pain you feel today will be the strength you feel tomorrow.",

"age is no barrier. It\’s a limitation you put on your mind.",

"do something today that your future self will thank you for.",

"life begins at the end of your comfort zone.",

"the difference between try, and triumph is a little \‘umph\’.",

"don\’t count the days, make the days count.",

"when you feel like quitting, think about why you started.",

"all progress takes place outside the comfort zone.",

"success starts with self-discipline.",

"the only bad workout is the one that didn\’t happen.",

"rome wasn\’t built in a day. Work hard and good results will come.",

"motivation is what gets you started. Habit is what keeps you going.",

"do something today that your future self will thank you for.",

];

// Array of streams for meditation and guided breathing sessions, including the mp3, artwork and background urls hosted on dropbox

const STREAMS = [{

token: '50',

url: 'https://dl.dropboxusercontent.com/s/k3qleiq76448agw/Birds%201.mp3?dl=1',

metadata: {

title: 'Birds',

subtitle: 'Unwind with relaxing bird sounds.',

art: {

sources: [{

contentDescription: 'Yellow Bird on Sakura Tree',

url: 'https://dl.dropboxusercontent.com/s/bace03559m40xgp/Birds%201%20Art.jpg?dl=1',

widthPixels: 512,

heightPixels: 512,

}, ],

},

backgroundImage: {

sources: [{

contentDescription: 'Peaceful Beach Setting',

url: 'https://dl.dropboxusercontent.com/s/x09l1kz72e25sj0/Birds%201%20Back.jpg?dl=1',

widthPixels: 1200,

heightPixels: 800,

}, ],

},

},

},

{

token: '51',

url: 'https://dl.dropboxusercontent.com/s/p7gjl9boun1t391/Birds%202.mp3?dl=1',

metadata: {

title: 'Birds',

subtitle: 'Unwind with relaxing bird sounds',

art: {

sources: [{

contentDescription: 'Blue, Yellow & Green Macaw Parrots',

url: 'https://dl.dropboxusercontent.com/s/yjudmzklumsed8u/Birds%202%20Art.jpg?dl=1',

widthPixels: 512,

heightPixels: 512,

}, ],

},

backgroundImage: {

sources: [{

contentDescription: 'Peaceful Beach Setting',

url: 'https://dl.dropboxusercontent.com/s/cdqalm4yndc4tl4/Birds%202%20Back.jpg?dl=1',

widthPixels: 1200,

heightPixels: 800,

}, ],

},

},

},

{

token: '52',

url: 'https://dl.dropboxusercontent.com/s/kip65qbw4zwa1bu/Fire%201.mp3?dl=1',

metadata: {

title: 'Fire',

subtitle: 'Unwind with relaxing fire sounds.',

art: {

sources: [{

contentDescription: 'Person Standing in Front of Fire',

url: 'https://dl.dropboxusercontent.com/s/yzt60blskso3o4v/Fire%201%20Art.jpg?dl=1',

widthPixels: 512,

heightPixels: 512,

}, ],

},

backgroundImage: {

sources: [{

contentDescription: 'Campfire on a Peaceful Night',

url: 'https://dl.dropboxusercontent.com/s/fx9luegdkef41u0/Fire%201%20Back.jpg?dl=1',

widthPixels: 1200,

heightPixels: 800,

}, ],

},

},

},

{

token: '53',

url: 'https://dl.dropboxusercontent.com/s/vcqf9k8xjpicuf9/Fire%202.mp3?dl=1',

metadata: {

title: 'Fire',

subtitle: 'Unwind with relaxing fire sounds.',

art: {

sources: [{

contentDescription: 'Campfire on Beach',

url: 'https://dl.dropboxusercontent.com/s/dpla0hanksge31f/Fire%202%20Art.jpg?dl=1',

widthPixels: 512,

heightPixels: 512,

}, ],

},

backgroundImage: {

sources: [{

contentDescription: 'Large Campfire',

url: 'https://dl.dropboxusercontent.com/s/eerj64fh6efrx5l/Fire%202%20Back.jpg?dl=1',

widthPixels: 1200,

heightPixels: 800,

}, ],

},

},

},

{

token: '54',

url: 'https://dl.dropboxusercontent.com/s/m4x5hg7ldsh9n2h/Ocean%20Waves%201.mp3?dl=1',

metadata: {

title: 'Ocean Waves',

subtitle: 'Unwind with relaxing ocean sounds.',

art: {

sources: [{

contentDescription: 'Underwater Coral & Fish',

url: 'https://dl.dropboxusercontent.com/s/kwby3v5ze5i5wzj/Ocean%20Waves%201%20Art.jpg?dl=1',

widthPixels: 512,

heightPixels: 512,

}, ],

},

backgroundImage: {

sources: [{

contentDescription: 'Peaceful Beach Setting',

url: 'https://dl.dropboxusercontent.com/s/5ib3qpk7lfbv0or/Ocean%20Waves%201%20Back.jpg?dl=1',

widthPixels: 1200,

heightPixels: 800,

}, ],

},

},

},

{

token: '55',

url: 'https://dl.dropboxusercontent.com/s/pm0wjhtxmfil29w/Ocean%20Waves%202.mp3?dl=1',

metadata: {

title: 'Ocean Waves',

subtitle: 'Unwind with relaxing ocean sounds.',

art: {

sources: [{

contentDescription: 'Person Floating in Clear Sea',

url: 'https://dl.dropboxusercontent.com/s/rkib3k4d47sfn30/Ocean%20Waves%202%20Art.jpg?dl=1',

widthPixels: 512,

heightPixels: 512,

}, ],

},

backgroundImage: {

sources: [{

contentDescription: 'Cathedral Cove, New Zealand',

url: 'https://dl.dropboxusercontent.com/s/9eqqtafdp49q4zk/Ocean%20Waves%202%20Back.jpg?dl=1',

widthPixels: 1200,

heightPixels: 800,

}, ],

},

},

},

{

token: '56',

url: 'https://dl.dropboxusercontent.com/s/y2o99nb63eesjtx/Rain%20on%20Car%201.mp3?dl=1',

metadata: {

title: 'Rain on a Car',

subtitle: 'Unwind with relaxing rain sounds.',

art: {

sources: [{

contentDescription: 'Raindrops on Window',

url: 'https://dl.dropboxusercontent.com/s/7de0ugrzhr814s5/Rain%20on%20Car%201%20Art.jpg?dl=1',

widthPixels: 512,

heightPixels: 512,

}, ],

},

backgroundImage: {

sources: [{

contentDescription: 'Raindrops on Car Window & Wingmiror',

url: 'https://dl.dropboxusercontent.com/s/4qixiqplj8brnc3/Rain%20on%20Car%201%20Back.jpg?dl=1',

widthPixels: 1200,

heightPixels: 800,

}, ],

},

},

},

{

token: '28',

url: 'https://dl.dropboxusercontent.com/s/vamvt35jlv1dw3s/Rain%20on%20Car%202.mp3?dl=1',

metadata: {

title: 'Rain on a Car',

subtitle: 'Unwind with relaxing rain sounds.',

art: {

sources: [{

contentDescription: 'Raining in a Busy City Centre',

url: 'https://dl.dropboxusercontent.com/s/crtbd66hyl9s7h6/Rain%20on%20Car%202%20Art.jpg?dl=1',

widthPixels: 512,

heightPixels: 512,

}, ],

},

backgroundImage: {

sources: [{

contentDescription: 'Person Holding an Umbrealla in the Rain',

url: 'https://dl.dropboxusercontent.com/s/jbtql4ziklq5zal/Rain%20on%20Car%202%20Back.jpg?dl=1',

widthPixels: 1200,

heightPixels: 800,

}, ],

},

},

},

{

token: '29',

url: 'https://dl.dropboxusercontent.com/s/lpj9ygob7o8l03p/Rain%201.mp3?dl=1',

metadata: {

title: 'Rain',

subtitle: 'Unwind with relaxing rain sounds.',

art: {

sources: [{

contentDescription: 'Raining in a Busy City Centre',

url: 'https://dl.dropboxusercontent.com/s/bgl88dh5ios65ox/Rain%201%20Art.jpg?dl=1',

widthPixels: 512,

heightPixels: 512,

}, ],

},

backgroundImage: {

sources: [{

contentDescription: 'Raindrops on a House Window',

url: 'https://dl.dropboxusercontent.com/s/8neu6qhwiek3ghg/Rain%201%20Back.jpg?dl=1',

widthPixels: 1200,

heightPixels: 800,

}, ],

},

},

},

{

token: '57',

url: 'https://dl.dropboxusercontent.com/s/oj677s7gsa1nnue/Rain%202.mp3?dl=1',

metadata: {

title: 'Rain',

subtitle: 'Unwind with relaxing rain sounds.',

art: {

sources: [{

contentDescription: 'Person Holding an Umbrealla in the Rain',

url: 'https://dl.dropboxusercontent.com/s/rp81ukbi9rovkse/Rain%202%20Art.jpg?dl=1',

widthPixels: 512,

heightPixels: 512,

}, ],

},

backgroundImage: {

sources: [{

contentDescription: 'Person Holding an Umbrealla in the Rain',

url: 'https://dl.dropboxusercontent.com/s/g92gt4gv4uuy21u/Rain%202%20Back.jpg?dl=1',

widthPixels: 1200,

heightPixels: 800,

}, ],

},

},

},

{

token: '31',

url: 'https://dl.dropboxusercontent.com/s/k31t5gwzy66hvrp/Tranquility.mp3?dl=1',

metadata: {

title: 'Tranquility',

subtitle: 'Unwind with lo-fi beats.',

art: {

sources: [{

contentDescription: 'Peaceful Costal Setting',

url: 'https://dl.dropboxusercontent.com/s/gjpl9w3081xv2dy/Tranquility%20Art.jpg?dl=1',

widthPixels: 512,

heightPixels: 512,

}, ],

},

backgroundImage: {

sources: [{

contentDescription: 'Neon Sign at Dock',

url: 'https://dl.dropboxusercontent.com/s/4zys450vwttq21d/Tranquility%20Back.jpg?dl=1',

widthPixels: 1200,

heightPixels: 800,

}, ],

},

},

},

{

token: '58',

url: 'https://dl.dropboxusercontent.com/s/yqwjgoeqyvca1gx/Crusing%20Along.mp3?dl=1',

metadata: {

title: 'Cruising Along',

subtitle: 'Unwind with lo-fi beats.',

art: {

sources: [{

contentDescription: 'Person Walking Along Peaceful Road',

url: 'https://dl.dropboxusercontent.com/s/0gtje9gp927pzhz/Cruising%20Along%20Art.jpg?dl=1',

widthPixels: 512,

heightPixels: 512,

}, ],

},

backgroundImage: {

sources: [{

contentDescription: 'City Landscape from Afar',

url: 'https: //dl.dropboxusercontent.com/s/6ybrf5u3gi4mvle/Cruising%20Along%20Back.jpg?dl=1',

widthPixels: 1200,

heightPixels: 800,

}, ],

},

},

},

{

token: '59',

url: 'https://dl.dropboxusercontent.com/s/xcz1kwdcx28zpat/Tropical.mp3?dl=1',

metadata: {

title: 'Tropical',

subtitle: 'Unwind with lo-fi beats.',

art: {

sources: [{

contentDescription: 'Person Walking Along Peaceful Road',

url: 'https://dl.dropboxusercontent.com/s/0gtje9gp927pzhz/Cruising%20Along%20Art.jpg?dl=1',

widthPixels: 512,

heightPixels: 512,

}, ],

},

backgroundImage: {

sources: [{

contentDescription: 'City Landscape from Afar',

url: 'https://dl.dropboxusercontent.com/s/6ybrf5u3gi4mvle/Cruising%20Along%20Back.jpg?dl=1',

widthPixels: 1200,

heightPixels: 800,

}, ],

},

},

},

{

token: '60',

url: 'https://dl.dropboxusercontent.com/s/qr7xwigc4l1fzhi/Mindulness%20of%20Breath.mp3?dl=1',

metadata: {

title: 'Mindfulness of Breath',

subtitle: 'Unwind with guided breathing.',

art: {

sources: [{

contentDescription: 'Person Meditating on a Beach',

url: 'https://dl.dropboxusercontent.com/s/91xamz50kw2hq07/Mindfulness%20of%20Breath%20Art.jpg?dl=1',

widthPixels: 512,

heightPixels: 512,

}, ],

},

backgroundImage: {

sources: [{

contentDescription: 'Peaceful Mountain Setting',

url: 'https://dl.dropboxusercontent.com/s/64cx88vu84h8vgp/Mindfulness%20of%20Breath%20Back.jpg?dl=1',

widthPixels: 1200,

heightPixels: 800,

}, ],

},

},

},

{

token: '61',

url: 'https://dl.dropboxusercontent.com/s/f07hxbwsilehwky/Mountain%20Meditation.mp3?dl=1',

metadata: {

title: 'Mountain Meditation',

subtitle: 'Unwind with guided breathing.',

art: {

sources: [{

contentDescription: 'Person Looking Out Onto Mountains',

url: 'https://dl.dropboxusercontent.com/s/3hgvgt7p993i4hl/Mountain%20Meditation%20Art.jpg?dl=1',

widthPixels: 512,

heightPixels: 512,

}, ],

},

backgroundImage: {

sources: [{

contentDescription: 'Peaceful Mountain Setting',

url: 'https://dl.dropboxusercontent.com/s/p2gziee71smrb6a/Mountain%20Meditation%20Back.jpg?dl=1',

widthPixels: 1200,

heightPixels: 800,

}, ],

},

},

},

{

token: '62',

url: 'https://dl.dropboxusercontent.com/s/jtw55p4c8mk5ol6/Wisdom%20Meditation.mp3?dl=1',

metadata: {

title: 'Wisdom Meditation',

subtitle: 'Unwind with guided breathing.',

art: {

sources: [{

contentDescription: 'Person Meditating on a Beach',

url: 'https://dl.dropboxusercontent.com/s/91xamz50kw2hq07/Mindfulness%20of%20Breath%20Art.jpg?dl=1',

widthPixels: 512,

heightPixels: 512,

}, ],

},

backgroundImage: {

sources: [{

contentDescription: 'Peaceful Mountain Setting',

url: 'https://dl.dropboxusercontent.com/s/64cx88vu84h8vgp/Mindfulness%20of%20Breath%20Back.jpg?dl=1',

widthPixels: 1200,

heightPixels: 800,

}, ],

},

},

},

];

const ranShortTips = Math.floor(Math.random() \* MAX\_SHORT\_TIPS); // Allocate a random tip to be displayed to the user

// Array of streams for workout sessions, including the mp3, artwork and background urls hosted on dropbox

const WORKOUT\_STREAMS = [{

token: '41',

url: 'https://dl.dropboxusercontent.com/s/sd5al0w6shjml5p/Lower%20Workout%201.mp3?dl=1',

metadata: {

title: 'Lower Body Workout',

subtitle: 'Remember, ' + shortTips[ranShortTips],

art: {

sources: [{

contentDescription: 'Person Working out With Weights',

url: 'https://dl.dropboxusercontent.com/s/1vwi8ozqs590shh/Workout%20Art.jpg?dl=1',

widthPixels: 512,

heightPixels: 512,

}, ],

},

backgroundImage: {

sources: [{

contentDescription: 'Person on a Beach',

url: 'https://dl.dropboxusercontent.com/s/wrwqn5j4ldgu2l7/Workout%20Back.jpg?dl=1',

widthPixels: 1200,

heightPixels: 800,

}, ],

},

},

},

{

token: '42',

url: 'https://dl.dropboxusercontent.com/s/05xayqai2bh7hvc/Lower%20Workout%202.mp3?dl=1',

metadata: {

title: 'Lower Body Workout',

subtitle: 'Remember, ' + shortTips[ranShortTips],

art: {

sources: [{

contentDescription: 'Person Working out With Weights',

url: 'https://dl.dropboxusercontent.com/s/1vwi8ozqs590shh/Workout%20Art.jpg?dl=1',

widthPixels: 512,

heightPixels: 512,

}, ],

},

backgroundImage: {

sources: [{

contentDescription: 'Person on a Beach',

url: 'https://dl.dropboxusercontent.com/s/wrwqn5j4ldgu2l7/Workout%20Back.jpg?dl=1',

widthPixels: 1200,

heightPixels: 800,

}, ],

},

},

},

{

token: '43',

url: 'https://dl.dropboxusercontent.com/s/p3hrojke29fwjlo/Upper%20Workout%201.mp3?dl=1',

metadata: {

title: 'Upper Body Workout',

subtitle: 'Remember, ' + shortTips[ranShortTips],

art: {

sources: [{

contentDescription: 'Person Working out With Weights',

url: 'https://dl.dropboxusercontent.com/s/1vwi8ozqs590shh/Workout%20Art.jpg?dl=1',

widthPixels: 512,

heightPixels: 512,

}, ],

},

backgroundImage: {

sources: [{

contentDescription: 'Person on a Beach',

url: 'https://dl.dropboxusercontent.com/s/wrwqn5j4ldgu2l7/Workout%20Back.jpg?dl=1',

widthPixels: 1200,

heightPixels: 800,

}, ],

},

},

},

{

token: '44',

url: 'https://dl.dropboxusercontent.com/s/2mhweuy34yminz0/Upper%20Workout%202.mp3?dl=1',

metadata: {

title: 'Upper Body Workout',

subtitle: 'Remember, ' + shortTips[ranShortTips],

art: {

sources: [{

contentDescription: 'Person Working out With Weights',

url: 'https://dl.dropboxusercontent.com/s/1vwi8ozqs590shh/Workout%20Art.jpg?dl=1',

widthPixels: 512,

heightPixels: 512,

}, ],

},

backgroundImage: {

sources: [{

contentDescription: 'Person on a Beach',

url: 'https://dl.dropboxusercontent.com/s/wrwqn5j4ldgu2l7/Workout%20Back.jpg?dl=1',

widthPixels: 1200,

heightPixels: 800,

}, ],

},

},

},

{

token: '45',

url: 'https://dl.dropboxusercontent.com/s/esg7bqm7s4rxgwh/Full%20Workout%201.mp3?dl=1',

metadata: {

title: 'Full Body Workout',

subtitle: 'Remember, ' + shortTips[ranShortTips],

art: {

sources: [{

contentDescription: 'Person Working out With Weights',

url: 'https://dl.dropboxusercontent.com/s/1vwi8ozqs590shh/Workout%20Art.jpg?dl=1',

widthPixels: 512,

heightPixels: 512,

}, ],

},

backgroundImage: {

sources: [{

contentDescription: 'Person on a Beach',

url: 'https://dl.dropboxusercontent.com/s/wrwqn5j4ldgu2l7/Workout%20Back.jpg?dl=1',

widthPixels: 1200,

heightPixels: 800,

}, ],

},

},

},

{

token: '46',

url: 'https://dl.dropboxusercontent.com/s/tlnoxpujt05k2am/Full%20Workout%202.mp3?dl=1',

metadata: {

title: 'Full Body Workout',

subtitle: 'Remember, ' + shortTips[ranShortTips],

art: {

sources: [{

contentDescription: 'Person Working out With Weights',

url: 'https://dl.dropboxusercontent.com/s/1vwi8ozqs590shh/Workout%20Art.jpg?dl=1',

widthPixels: 512,

heightPixels: 512,

}, ],

},

backgroundImage: {

sources: [{

contentDescription: 'Person on a Beach',

url: 'https://dl.dropboxusercontent.com/s/wrwqn5j4ldgu2l7/Workout%20Back.jpg?dl=1',

widthPixels: 1200,

heightPixels: 800,

}, ],

},

},

},

];

//Intent Handlers =============================================

// Built-in Cancel Intent

/\*

const AMAZON\_CancelIntent\_Handler = {

canHandle(handlerInput) {

const request = handlerInput.requestEnvelope.request;

return request.type === 'IntentRequest' && (request.intent.name === 'AMAZON.CancelIntent');

},

handle(handlerInput) {

const responseBuilder = handlerInput.responseBuilder;

const ranTips = Math.floor(Math.random() \* MAX\_TIPS);

let say = 'Thank you for using ' + invocationName + '. You can schedule a reminder to workout or meditate at this time tomorrow by saying, "remind me to exercise with Intent Fitness". Remember, ' + tips[ranTips] + ' Have a great day!';

handlerInput.responseBuilder

.addAudioPlayerClearQueueDirective('CLEAR\_ALL')

.addAudioPlayerStopDirective();

return responseBuilder

.speak(say)

.withShouldEndSession(true)

.getResponse();

},

};

\*/

// Intent to stop audio playback

const PlaybackStoppedIntentHandler = {

canHandle(handlerInput) {

return handlerInput.requestEnvelope.request.type === 'PlaybackController.PauseCommandIssued'

|| handlerInput.requestEnvelope.request.type === 'AudioPlayer.PlaybackStopped';

},

handle(handlerInput) {

handlerInput.responseBuilder

.addAudioPlayerClearQueueDirective('CLEAR\_ALL')

.addAudioPlayerStopDirective();

return handlerInput.responseBuilder

.getResponse();

},

};

// Intent to start audio playback

const PlaybackStartedIntentHandler = {

canHandle(handlerInput) {

return handlerInput.requestEnvelope.request.type === 'AudioPlayer.PlaybackStarted';

},

handle(handlerInput) {

handlerInput.responseBuilder

.addAudioPlayerClearQueueDirective('CLEAR\_ENQUEUED');

return handlerInput.responseBuilder

.getResponse();

},

};

// Built-in Stop & Cancel Intent

const CancelAndStopIntentHandler = {

canHandle(handlerInput) {

return handlerInput.requestEnvelope.request.type === 'IntentRequest'

&& (

handlerInput.requestEnvelope.request.intent.name === 'AMAZON.StopIntent'

|| handlerInput.requestEnvelope.request.intent.name === 'AMAZON.PauseIntent'

|| handlerInput.requestEnvelope.request.intent.name === 'AMAZON.CancelIntent'

|| handlerInput.requestEnvelope.request.intent.name === 'AMAZON.LoopOffIntent'

|| handlerInput.requestEnvelope.request.intent.name === 'AMAZON.ShuffleOffIntent'

);

},

handle(handlerInput) {

const ranTips = Math.floor(Math.random() \* MAX\_TIPS);

let say = 'Thank you for using ' + invocationName + '. You can schedule a reminder to workout or meditate at this time tomorrow by saying, "remind me to exercise with Intent Fitness". Remember, ' + tips[ranTips] + ' Have a great day!';

handlerInput.responseBuilder

.addAudioPlayerClearQueueDirective('CLEAR\_ALL')

.addAudioPlayerStopDirective();

handlerInput.responseBuilder

.addAudioPlayerClearQueueDirective('CLEAR\_ALL')

.addAudioPlayerStopDirective();

return handlerInput.responseBuilder

.speak(say)

.withShouldEndSession(true)

.getResponse();

},

};

// Built-in Help Intent

const AMAZON\_HelpIntent\_Handler = {

canHandle(handlerInput) {

const request = handlerInput.requestEnvelope.request;

return request.type === 'IntentRequest' && request.intent.name === 'AMAZON.HelpIntent';

},

handle(handlerInput) {

const responseBuilder = handlerInput.responseBuilder;

let say = 'Sure, I can help. This app allows you to get started with a quick 10 minute workout, unwind with 10 minutes of meditation or follow a 10 minute guided breathing session. Which will it be?';

return responseBuilder

.speak(say)

.reprompt()

.getResponse();

},

};

// Create Reminder Intent allows users to set a reminder to workout or meditate 24 hours after they use the app

const CreateReminderIntentHandler = {

canHandle(handlerInput) {

return Alexa.getRequestType(handlerInput.requestEnvelope) === 'IntentRequest' &&

Alexa.getIntentName(handlerInput.requestEnvelope) === 'getReminder';

},

async handle(handlerInput) {

const request = handlerInput.requestEnvelope.request;

// delegate to Alexa to collect all the required slots

const currentIntent = request.intent;

if (request.dialogState && request.dialogState !== 'COMPLETED') {

return handlerInput.responseBuilder

.addDelegateDirective(currentIntent)

.getResponse();

}

let slotValues = getSlotValues(request.intent.slots);

if (slotValues && slotValues.reminder) {

const reminderApiClient = handlerInput.serviceClientFactory.getReminderManagementServiceClient(),

{ permissions } = handlerInput.requestEnvelope.context.System.user;

// Ensure user has given permission to set reminders in the Alexa mobile app

if (!permissions) {

return handlerInput.responseBuilder

.speak("Please go to the Alexa mobile app to grant reminders permissions.")

.withAskForPermissionsConsentCard(['alexa::alerts:reminders:skill:readwrite'])

.getResponse();

}

// Send notification and reminder for user to workout

const reminderRequestWorkout = {

"trigger": {

"type": "SCHEDULED\_RELATIVE",

"offsetInSeconds": "86400"

},

"alertInfo": {

"spokenInfo": {

"content": [{

"locale": "en-GB",

"text": "Intent Fitness: Remember to exercise today!",

"ssml": "<speak> Remember to exercise today with Intent Fitness!</speak>"

}]

}

},

"pushNotification": {

"status": "ENABLED"

}

};

// Send notification and reminder for user to meditate

const reminderRequestMeditate = {

"trigger": {

"type": "SCHEDULED\_RELATIVE",

"offsetInSeconds": "86400"

},

"alertInfo": {

"spokenInfo": {

"content": [{

"locale": "en-GB",

"text": "Intent Fitness: Remember to meditate today!",

"ssml": "<speak> Remember to meditate today with Intent Fitness!</speak>"

}]

}

},

"pushNotification": {

"status": "ENABLED"

}

};

// Send reminder request depending on user input and catch error if slot value is not acceptable

if (slotValues.reminder.heardAs == 'meditate' || slotValues.reminder.heardAs == 'meditation' || slotValues.reminder.heardAs == 'meditating' || slotValues.reminder.heardAs == 'relax') {

try {

await reminderApiClient.createReminder(reminderRequestMeditate);

}

catch (error) {

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

return handlerInput.responseBuilder

.speak('There was an error scheduling your reminder. Please try again later.')

.getResponse();

}

const speakOutput = 'You have successfully scheduled a reminder to meditate at this time tomorrow!';

return handlerInput.responseBuilder

.speak(speakOutput)

.getResponse();

}

// Send reminder request depending on user input and catch error if slot value is not acceptable

else if (slotValues.reminder.heardAs == 'workout' || slotValues.reminder.heardAs == 'exercise') {

try {

await reminderApiClient.createReminder(reminderRequestWorkout);

}

catch (error) {

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

return handlerInput.responseBuilder

.speak('There was an error scheduling your reminder. Please try again later.')

.getResponse();

}

const speakOutput = 'You have successfully scheduled a reminder to exercise at this time tomorrow!';

return handlerInput.responseBuilder

.speak(speakOutput)

.getResponse();

}

}

}

};

const AMAZON\_PauseIntent\_Handler = {

canHandle(handlerInput) {

const request = handlerInput.requestEnvelope.request;

return request.type === 'IntentRequest' && request.intent.name === 'AMAZON.PauseIntent' ;

},

handle(handlerInput) {

const responseBuilder = handlerInput.responseBuilder;

let say = ' ';

return responseBuilder

.speak(say)

.reprompt('try again, ' + say)

.getResponse();

},

};

// Built-in Navigate Home Intent

const AMAZON\_NavigateHomeIntent\_Handler = {

canHandle(handlerInput) {

const request = handlerInput.requestEnvelope.request;

return request.type === 'IntentRequest' && request.intent.name === 'AMAZON.NavigateHomeIntent';

},

handle(handlerInput) {

const responseBuilder = handlerInput.responseBuilder;

return responseBuilder

.speak('Sorry, I didn\'t quite get that. You can get started with a quick 10 minute workout, unwind with 10 minutes of meditation or follow a 10 minute guided breathing session. Which will it be?')

.reprompt()

.getResponse();

},

};

// Built-in Fallback Intent

const AMAZON\_FallbackIntent\_Handler = {

canHandle(handlerInput) {

const request = handlerInput.requestEnvelope.request;

return request.type === 'IntentRequest' && request.intent.name === 'AMAZON.FallbackIntent';

},

handle(handlerInput) {

const responseBuilder = handlerInput.responseBuilder;

return responseBuilder

.speak('Sorry, I didn\'t quite get that. You can get started with a quick 10 minute workout, unwind with 10 minutes of meditation or follow a 10 minute guided breathing session. Which will it be?')

.reprompt()

.getResponse();

},

};

// GetSound Intent to allow users to get a sound so they can meditate and follow guided breathing sessions

const getSounds\_Handler = {

canHandle(handlerInput) {

const request = handlerInput.requestEnvelope.request;

return request.type === 'IntentRequest' && request.intent.name === 'getSounds';

},

handle(handlerInput) {

const request = handlerInput.requestEnvelope.request;

const responseBuilder = handlerInput.responseBuilder;

// Generate random numbers to allocate a specific sound or allow user to choose a random sound

const ranSong = Math.floor(Math.random() \* MAX\_NUMBER);

const ranBirds = Math.floor(Math.random() \* 2); // 0 and 1

const ranFire = Math.floor((Math.random() \* 2) + 2); // 2 and 3

const ranOcean = Math.floor((Math.random() \* 2) + 4); // 4 and 5

const ranRainonCar = Math.floor((Math.random() \* 2) + 6); // 6 and 7

const ranRain = Math.floor((Math.random() \* 2) + 8); // 8 and 9

const ranLoFi = Math.floor((Math.random() \* 3) + 10); //10, 11 and 12

// Allocate sounds to index in stream array

const sBirds = STREAMS[ranBirds];

const sFire = STREAMS[ranFire];

const sOcean = STREAMS[ranOcean];

const sRainonCar = STREAMS[ranRainonCar];

const sRain = STREAMS[ranRain];

const sLoFi = STREAMS[ranLoFi];

const sRandom = STREAMS[ranSong];

const sMindofBreath = STREAMS[13];

const sMountBreathing = STREAMS[14];

const sWisdomBreathing = STREAMS[15];

// delegate to Alexa to collect all the required slots

const currentIntent = request.intent;

if (request.dialogState && request.dialogState !== 'COMPLETED') {

return handlerInput.responseBuilder

.addDelegateDirective(currentIntent)

.getResponse();

}

let say = '';

let slotValues = getSlotValues(request.intent.slots);

// Play sounds depending on the user input and slot values

if (slotValues && slotValues.sound) {

if (slotValues.sound.heardAs == 'birds' || slotValues.sound.heardAs == 'bird') {

say = 'Great choice! I will play bird sounds for you.';

handlerInput.responseBuilder

.addAudioPlayerPlayDirective('REPLACE\_ALL', sBirds.url, sBirds.token, 0, null, sBirds.metadata);

}

else if (slotValues.sound.heardAs == 'fire' || slotValues.sound.heardAs == 'crackling') {

say = 'Great choice! I will play fire sounds for you.';

handlerInput.responseBuilder

.addAudioPlayerPlayDirective('REPLACE\_ALL', sFire.url, sFire.token, 0, null, sFire.metadata);

}

else if (slotValues.sound.heardAs == 'ocean waves' || slotValues.sound.heardAs == 'ocean' || slotValues.sound.heardAs == 'waves' || slotValues.sound.heardAs == 'ocean wave') {

say = 'Great choice! I will play ocean sounds for you.';

handlerInput.responseBuilder

.addAudioPlayerPlayDirective('REPLACE\_ALL', sOcean.url, sOcean.token, 0, null, sOcean.metadata);

}

else if (slotValues.sound.heardAs == 'rain on car' || slotValues.sound.heardAs == 'raining on a car' || slotValues.sound.heardAs == 'rain on a car' || slotValues.sound.heardAs == 'raining on car') {

say = 'Great choice! I will play rain on a car sounds for you.';

handlerInput.responseBuilder

.addAudioPlayerPlayDirective('REPLACE\_ALL', sRainonCar.url, sRainonCar.token, 0, null, sRainonCar.metadata);

}

else if (slotValues.sound.heardAs == 'rain' || slotValues.sound.heardAs == 'rain drop' || slotValues.sound.heardAs == 'rain drops') {

say = 'Great choice! I will play rain sounds for you.';

handlerInput.responseBuilder

.addAudioPlayerPlayDirective('REPLACE\_ALL', sRain.url, sRain.token, 0, null, sRain.metadata);

}

else if (slotValues.sound.heardAs == 'chill' || slotValues.sound.heardAs == 'chilled' || slotValues.sound.heardAs == 'relaxing') {

say = 'Great choice! I will play chilled lo-fi sounds for you.';

handlerInput.responseBuilder

.addAudioPlayerPlayDirective('REPLACE\_ALL', sLoFi.url, sLoFi.token, 0, null, sLoFi.metadata);

}

else if (slotValues.sound.heardAs == 'random') {

say = 'Great choice! I will choose a random relaxing sound to play for you.';

handlerInput.responseBuilder

.addAudioPlayerPlayDirective('REPLACE\_ALL', sRandom.url, sRandom.token, 0, null, sRandom.metadata);

}

else if (slotValues.sound.heardAs == 'mountain' || slotValues.sound.heardAs == 'mountain meditation' || slotValues.sound.heardAs == 'session one') {

say = 'Great choice!';

handlerInput.responseBuilder

.addAudioPlayerPlayDirective('REPLACE\_ALL', sMountBreathing.url, sMountBreathing.token, 0, null, sMountBreathing.metadata);

}

else if (slotValues.sound.heardAs == 'mindfulness of breath' || slotValues.sound.heardAs == 'mindfulness' || slotValues.sound.heardAs == 'breath' || slotValues.sound.heardAs == 'session one') {

say = 'Great choice!';

handlerInput.responseBuilder

.addAudioPlayerPlayDirective('REPLACE\_ALL', sMindofBreath.url, sMindofBreath.token, 0, null, sMindofBreath.metadata);

}

else if (slotValues.sound.heardAs == 'wisdom meditation' || slotValues.sound.heardAs == 'wisdom' || slotValues.sound.heardAs == 'session three') {

say = 'Great choice!';

handlerInput.responseBuilder

.addAudioPlayerPlayDirective('REPLACE\_ALL', sWisdomBreathing.url, sWisdomBreathing.token, 0, null, sWisdomBreathing.metadata);

}

}

return responseBuilder

.speak(say)

.getResponse();

},

};

// GetMode Intent to allow user to choose to meditate, follow guided breathing sessions or workout

const getMode\_Handler = {

canHandle(handlerInput) {

const request = handlerInput.requestEnvelope.request;

return request.type === 'IntentRequest' && request.intent.name === 'getMode';

},

handle(handlerInput) {

const request = handlerInput.requestEnvelope.request;

const responseBuilder = handlerInput.responseBuilder;

// delegate to Alexa to collect all the required slots

const currentIntent = request.intent;

if (request.dialogState && request.dialogState !== 'COMPLETED') {

return handlerInput.responseBuilder

.addDelegateDirective(currentIntent)

.getResponse();

}

let say = '';

let slotValues = getSlotValues(request.intent.slots);

// Choose a mode, depending on user input and slot validation

if (slotValues && slotValues.mode) {

if (slotValues.mode.heardAs == 'meditation' || slotValues.mode.heardAs == 'meditate' || slotValues.mode.heardAs == 'meditating' || slotValues.mode.heardAs == 'ten minute meditate' || slotValues.mode.heardAs == 'ten minute meditation') {

say = 'Sure! We will get started with a meditation session. You can choose to listen to sounds such as birds, rain, fire and waves. You can also say random and I will choose a sound for you. What would you like to listen to?';

}

else if (slotValues.mode.heardAs == 'workout' || slotValues.mode.heardAs == 'exercise' || slotValues.mode.heardAs == 'ten minute workout' || slotValues.mode.heardAs == 'ten minute exercise') {

say = 'Sure! We will get started with a workout session. You can choose a light, moderate or intense workout. Which intensity level would you like?';

}

else if (slotValues.mode.heardAs == 'guided breathing' || slotValues.mode.heardAs == 'breathing' || slotValues.mode.heardAs == 'ten minute guided breathing' || slotValues.mode.heardAs == 'ten minute breathing') {

say = 'Sure! We will get started with a guided breathing session. You can choose either Mindfulness of Breath, Mountain Meditation or Wisdom Meditation to start different sessions. Which will it be?';

}

}

return responseBuilder

.speak(say)

.reprompt('Sorry, I didn\'t quite get that. You can get started with a quick 10 minute workout, unwind with 10 minutes of meditation or follow a 10 minute guided breathing session. Which will it be?')

.getResponse();

},

};

// GetTarget Intent to allow user to choose an area of their body to workout

const getTargetArea\_Handler = {

canHandle(handlerInput) {

const request = handlerInput.requestEnvelope.request;

return request.type === 'IntentRequest' && request.intent.name === 'getTargetArea';

},

handle(handlerInput) {

const request = handlerInput.requestEnvelope.request;

const responseBuilder = handlerInput.responseBuilder;

// delegate to Alexa to collect all the required slots

const currentIntent = request.intent;

if (request.dialogState && request.dialogState !== 'COMPLETED') {

return handlerInput.responseBuilder

.addDelegateDirective(currentIntent)

.getResponse();

}

const ranLower = Math.floor(Math.random() \* 2); // 0 and 1

const ranUpper = Math.floor((Math.random() \* 2) + 2); // 2 and 3

const ranFull = Math.floor((Math.random() \* 2) + 4); // 4 and 5

// Allocate workout audio files to index of workout stream array

const sLower1 = WORKOUT\_STREAMS[ranLower];

const sUpper1 = WORKOUT\_STREAMS[ranUpper];

const sFull1 = WORKOUT\_STREAMS[ranFull];

let say = '';

let slotValues = getSlotValues(request.intent.slots);

// Follow an upper body, lower body or full body workout and play the audio stream depending on user input and slot validaiton

if (slotValues && slotValues.area) {

if (slotValues.area.heardAs == 'upper body' || slotValues.area.heardAs == 'upper' || slotValues.area.heardAs == 'arms' || slotValues.area.heardAs == 'chest') {

say = 'Great choice! We will get started with an upper body workout.';

handlerInput.responseBuilder

.addAudioPlayerPlayDirective('REPLACE\_ALL', sUpper1.url, sUpper1.token, 0, null, sUpper1.metadata);

}

else if (slotValues.area.heardAs == 'lower body' || slotValues.area.heardAs == 'lower' || slotValues.area.heardAs == 'legs') {

say = 'Great choice! We will get started with a lower body workout.';

handlerInput.responseBuilder

.addAudioPlayerPlayDirective('REPLACE\_ALL', sLower1.url, sLower1.token, 0, null, sLower1.metadata);

}

else if (slotValues.area.heardAs == 'full body' || slotValues.area.heardAs == 'full' || slotValues.area.heardAs == 'whole' || slotValues.area.heardAs == 'whole body') {

say = 'Great choice! We will get started with a full body workout.';

handlerInput.responseBuilder

.addAudioPlayerPlayDirective('REPLACE\_ALL', sFull1.url, sFull1.token, 0, null, sFull1.metadata);

}

}

return responseBuilder

.speak(say)

//.reprompt('try again, ' + say)

.getResponse();

},

};

// GetTarget Intent to allow user to choose an intensity level for their workout

const getIntensity\_Handler = {

canHandle(handlerInput) {

const request = handlerInput.requestEnvelope.request;

return request.type === 'IntentRequest' && request.intent.name === 'getIntensity';

},

handle(handlerInput) {

const request = handlerInput.requestEnvelope.request;

const responseBuilder = handlerInput.responseBuilder;

// delegate to Alexa to collect all the required slots

const currentIntent = request.intent;

if (request.dialogState && request.dialogState !== 'COMPLETED') {

return handlerInput.responseBuilder

.addDelegateDirective(currentIntent)

.getResponse();

}

let say = '';

// Allocate an intensity level depending on the users input and slot validation

let slotValues = getSlotValues(request.intent.slots);

if (slotValues && slotValues.intensity) {

if (slotValues.intensity.heardAs == 'vigorous' || slotValues.intensity.heardAs == 'intense' || slotValues.intensity.heardAs == 'difficult' || slotValues.intensity.heardAs == 'hard') {

say = 'Great choice! An intense workout will really get you sweating! Perform each exercise as fast as you can, doing as many reps as you can within the time. Between each exercise, push yourself to have a short rest time, making sure you still stay hydrated. You can choose to target your lower body, upper body, or complete a full body workout. Which will it be?';

}

else if (slotValues.intensity.heardAs == 'moderate' || slotValues.intensity.heardAs == 'regular' || slotValues.intensity.heardAs == 'medium' || slotValues.intensity.heardAs == 'average') {

say = 'Great choice! A moderate workout will kick up your heart rate and burn more calories, all while making you sweat. Between each exercise, push yourself to reduce your rest time. You can choose to target your lower body, upper body, or complete a full body workout. Which will it be?';

}

else if (slotValues.intensity.heardAs == 'light' || slotValues.intensity.heardAs == 'easy' || slotValues.intensity.heardAs == 'easier') {

say = 'Great choice! For a light workout, use a normal, comfortable pace and make sure you rest and drink water between each exercise. You can choose to target your lower body, upper body, or complete a full body workout. Which will it be?';

}

}

return responseBuilder

.speak(say)

.reprompt('try again, ' + say)

.getResponse();

},

};

// Built-in Launch request to start the application

const LaunchRequest\_Handler = {

canHandle(handlerInput) {

const request = handlerInput.requestEnvelope.request;

return request.type === 'LaunchRequest';

},

handle(handlerInput) {

const responseBuilder = handlerInput.responseBuilder;

let say = 'Hello' + ' and welcome to ' + invocationName + '! You can get started with a quick 10 minute workout, unwind with 10 minutes of meditation or follow a 10 minute guided breathing session. Which will it be?';

return responseBuilder

.speak(say)

.reprompt('Sorry, I didn\'t quite get that. You can get started with a quick 10 minute workout, unwind with 10 minutes of meditation or follow a 10 minute guided breathing session. Which will it be?')

.getResponse();

},

};

// Built-in Session End request

const SessionEndedHandler = {

canHandle(handlerInput) {

const request = handlerInput.requestEnvelope.request;

return request.type === 'SessionEndedRequest';

},

handle(handlerInput) {

console.log(`Session ended with reason: ${handlerInput.requestEnvelope.request.reason}`);

return handlerInput.responseBuilder.getResponse();

}

};

// Helper function to get slot values from user and validate them

function getSlotValues(filledSlots) {

const slotValues = {};

Object.keys(filledSlots).forEach((item) => {

const name = filledSlots[item].name;

if (filledSlots[item] &&

filledSlots[item].resolutions &&

filledSlots[item].resolutions.resolutionsPerAuthority[0] &&

filledSlots[item].resolutions.resolutionsPerAuthority[0].status &&

filledSlots[item].resolutions.resolutionsPerAuthority[0].status.code) {

switch (filledSlots[item].resolutions.resolutionsPerAuthority[0].status.code) {

case 'ER\_SUCCESS\_MATCH':

slotValues[name] = {

heardAs: filledSlots[item].value,

resolved: filledSlots[item].resolutions.resolutionsPerAuthority[0].values[0].value.name,

ERstatus: 'ER\_SUCCESS\_MATCH'

};

break;

case 'ER\_SUCCESS\_NO\_MATCH':

slotValues[name] = {

heardAs: filledSlots[item].value,

resolved: '',

ERstatus: 'ER\_SUCCESS\_NO\_MATCH'

};

break;

default:

break;

}

}

else {

slotValues[name] = {

heardAs: filledSlots[item].value || '', // may be null

resolved: '',

ERstatus: ''

};

}

}, this);

return slotValues;

}

// Exports handler function and setup =================================================

const skillBuilder = Alexa.SkillBuilders.custom();

exports.handler = skillBuilder

.addRequestHandlers(

AMAZON\_HelpIntent\_Handler,

AMAZON\_NavigateHomeIntent\_Handler,

PlaybackStoppedIntentHandler,

AMAZON\_FallbackIntent\_Handler,

AMAZON\_PauseIntent\_Handler,

PlaybackStartedIntentHandler,

getSounds\_Handler,

getMode\_Handler,

getTargetArea\_Handler,

LaunchRequest\_Handler,

CreateReminderIntentHandler,

getIntensity\_Handler,

SessionEndedHandler,

CancelAndStopIntentHandler,

)

.withApiClient(new Alexa.DefaultApiClient())

.lambda();

**Interaction Model JSON**

{

"interactionModel": {

"languageModel": {

"invocationName": "intent fitness",

"modelConfiguration": {

"fallbackIntentSensitivity": {

"level": "LOW"

}

},

"intents": [

{

"name": "AMAZON.CancelIntent",

"samples": []

},

{

"name": "AMAZON.HelpIntent",

"samples": []

},

{

"name": "AMAZON.StopIntent",

"samples": []

},

{

"name": "AMAZON.NavigateHomeIntent",

"samples": []

},

{

"name": "AMAZON.FallbackIntent",

"samples": []

},

{

"name": "getSounds",

"slots": [

{

"name": "sound",

"type": "SOUNDS",

"samples": [

"play {sound} sounds for me",

"i want to listen to {sound} sounds",

"{sound}",

"i want to listen to {sound}",

"i would like to listen to {sound}",

"play {sound} for me",

"play {sound} sounds",

"play {sound}"

]

}

],

"samples": [

"play {sound} sounds for me",

"play {sound} sounds",

"play {sound} ",

"play {sound} for me",

"I would like to listen to {sound}",

"relax me",

"I want to listen to {sound}",

"{sound}",

"I want to listen to {sound} sounds"

]

},

{

"name": "PlayStreamIntent",

"slots": [],

"samples": [

"to play",

"to start playing",

"to begin playing"

]

},

{

"name": "AboutIntent",

"slots": [],

"samples": [

"about",

"what is this",

"tell me about this"

]

},

{

"name": "AMAZON.PauseIntent",

"samples": []

},

{

"name": "AMAZON.ResumeIntent",

"samples": []

},

{

"name": "AMAZON.LoopOnIntent",

"samples": []

},

{

"name": "AMAZON.NextIntent",

"samples": []

},

{

"name": "AMAZON.PreviousIntent",

"samples": []

},

{

"name": "AMAZON.RepeatIntent",

"samples": []

},

{

"name": "AMAZON.ShuffleOnIntent",

"samples": []

},

{

"name": "AMAZON.StartOverIntent",

"samples": []

},

{

"name": "AMAZON.LoopOffIntent",

"samples": []

},

{

"name": "AMAZON.ShuffleOffIntent",

"samples": []

},

{

"name": "getMode",

"slots": [

{

"name": "mode",

"type": "MODE",

"samples": [

"i want to {mode}",

"i would like to {mode}",

"{mode}"

]

}

],

"samples": [

"i want to {mode}",

"{mode}",

"I would like to {mode}"

]

},

{

"name": "getReminder",

"slots": [

{

"name": "reminder",

"type": "REMINDER",

"samples": [

"remind me to {reminder}",

"set up a reminder to {reminder}",

"remember me to {reminder}",

"i would like to {reminder}",

"{reminder}"

]

}

],

"samples": [

"remember me",

"remember me to {reminder}",

"remind me",

"set up a reminder to {reminder}",

"remind me to {reminder}"

]

},

{

"name": "getTargetArea",

"slots": [

{

"name": "area",

"type": "AREA",

"samples": [

"i want to exercise my {area}",

"{area} workout",

"i would like to complete a {area} workout",

"i want to complete a {area} workout",

"i want to target {area}",

"i would like to target {area}",

"{area}",

"i want to complete an {area} workout",

"i would like to complete an {area} workout"

]

}

],

"samples": [

"i would like to complete an {area} workout",

"i want to complete an {area} workout",

"i want to exercise my {area}",

"{area} workout",

"i would like to complete a {area} workout",

"i want to complete a {area} workout",

"i want to target {area}",

"i would like to target {area}",

"{area}"

]

},

{

"name": "getIntensity",

"slots": [

{

"name": "intensity",

"type": "INTENSITY",

"samples": [

"{intensity} workout",

"i would like to complete a {intensity} workout",

"i would like to complete an {intensity} workour",

"i want to complete a workout",

"i want to complete an {intensity} workout",

"{intensity}"

]

}

],

"samples": [

"i want to complete an {intensity} workout",

"i would like to complete an {intensity} workout",

"{intensity} workout",

"i would like to complete a {intensity} workout",

"i want to complete a {intensity} workout",

"{intensity}"

]

}

],

"types": [

{

"name": "SOUNDS",

"values": [

{

"name": {

"value": "wisdom meditation",

"synonyms": [

"session three",

"wisdom"

]

}

},

{

"name": {

"value": "mindfulness of breath",

"synonyms": [

"session one",

"breath",

"mindfulness"

]

}

},

{

"name": {

"value": "mountain",

"synonyms": [

"session 2",

"mountain meditation"

]

}

},

{

"name": {

"value": "trees",

"synonyms": [

"tree"

]

}

},

{

"name": {

"value": "chill",

"synonyms": [

"chilled",

"relaxing"

]

}

},

{

"name": {

"value": "random"

}

},

{

"name": {

"value": "rain on car",

"synonyms": [

"raining on a car",

"rain on a car",

"raining on car"

]

}

},

{

"name": {

"value": "rain",

"synonyms": [

"rain drop",

"rain drops"

]

}

},

{

"name": {

"value": "ocean waves",

"synonyms": [

"ocean wave",

"waves",

"ocean"

]

}

},

{

"name": {

"value": "fire",

"synonyms": [

"crackling"

]

}

},

{

"name": {

"value": "birds",

"synonyms": [

"bird"

]

}

}

]

},

{

"name": "MODE",

"values": [

{

"name": {

"value": "guided breathing",

"synonyms": [

"10 minute guided breathing",

"10 minute breathing",

"breathing"

]

}

},

{

"name": {

"value": "meditate",

"synonyms": [

"ten minute meditate",

"ten minute meditation",

"meditating",

"meditation"

]

}

},

{

"name": {

"value": "workout",

"synonyms": [

"ten minute workout",

"ten minute exercise",

"exercise"

]

}

}

]

},

{

"name": "REMINDER",

"values": [

{

"name": {

"value": "meditate",

"synonyms": [

"relax",

"meditating",

"meditation"

]

}

},

{

"name": {

"value": "workout",

"synonyms": [

"exercise"

]

}

}

]

},

{

"name": "AREA",

"values": [

{

"name": {

"value": "full body",

"synonyms": [

"whole body",

"whole",

"full"

]

}

},

{

"name": {

"value": "lower body",

"synonyms": [

"legs",

"lower"

]

}

},

{

"name": {

"value": "upper body",

"synonyms": [

"chest",

"arms",

"upper"

]

}

}

]

},

{

"name": "INTENSITY",

"values": [

{

"name": {

"value": "vigorous",

"synonyms": [

"intense",

"difficult",

"hard"

]

}

},

{

"name": {

"value": "moderate",

"synonyms": [

"regular",

"average",

"medium"

]

}

},

{

"name": {

"value": "light",

"synonyms": [

"easier",

"easy"

]

}

}

]

}

]

},

"dialog": {

"intents": [

{

"name": "getSounds",

"confirmationRequired": false,

"prompts": {},

"slots": [

{

"name": "sound",

"type": "SOUNDS",

"confirmationRequired": false,

"elicitationRequired": true,

"prompts": {

"elicitation": "Elicit.Slot.1518147732038.1393534478204"

},

"validations": [

{

"type": "hasEntityResolutionMatch",

"prompt": "Slot.Validation.1059701369172.183985370.529286722304"

}

]

}

]

},

{

"name": "getMode",

"confirmationRequired": false,

"prompts": {},

"slots": [

{

"name": "mode",

"type": "MODE",

"confirmationRequired": false,

"elicitationRequired": true,

"prompts": {

"elicitation": "Elicit.Slot.757103645059.1413855593511"

},

"validations": [

{

"type": "hasEntityResolutionMatch",

"prompt": "Slot.Validation.1198761464102.1268688566038.1021182339498"

}

]

}

]

},

{

"name": "getReminder",

"confirmationRequired": false,

"prompts": {},

"slots": [

{

"name": "reminder",

"type": "REMINDER",

"confirmationRequired": false,

"elicitationRequired": true,

"prompts": {

"elicitation": "Elicit.Slot.1308217539947.242246796500"

}

}

]

},

{

"name": "getTargetArea",

"confirmationRequired": false,

"prompts": {},

"slots": [

{

"name": "area",

"type": "AREA",

"confirmationRequired": false,

"elicitationRequired": true,

"prompts": {

"elicitation": "Elicit.Slot.1216715640341.1373587319490"

},

"validations": [

{

"type": "hasEntityResolutionMatch",

"prompt": "Slot.Validation.1216715640341.1373587319490.55628938039"

}

]

}

]

},

{

"name": "getIntensity",

"confirmationRequired": false,

"prompts": {},

"slots": [

{

"name": "intensity",

"type": "INTENSITY",

"confirmationRequired": false,

"elicitationRequired": true,

"prompts": {

"elicitation": "Elicit.Slot.1346554512009.601913546176"

},

"validations": [

{

"type": "hasEntityResolutionMatch",

"prompt": "Slot.Validation.1346554512009.601913546176.472635673855"

}

]

}

]

}

],

"delegationStrategy": "ALWAYS"

},

"prompts": [

{

"id": "Slot.Validation.1059701369172.183985370.529286722304",

"variations": [

{

"type": "PlainText",

"value": "Sorry, I don't know that one. You can choose to listen to sounds such as birds, rain, fire and waves. You can also say random and I will choose a sound for you. What would you like to listen to?"

}

]

},

{

"id": "Elicit.Slot.1518147732038.1393534478204",

"variations": [

{

"type": "PlainText",

"value": "You can choose to listen to sounds such as birds, rain, fire and waves. You can also say random and I will choose a sound for you. What would you like to listen to?"

}

]

},

{

"id": "Elicit.Slot.757103645059.1413855593511",

"variations": [

{

"type": "PlainText",

"value": "You can get started with a quick ten minute workout, unwind with ten minutes of meditation or follow a ten minute guided breathing session. Which will it be?"

}

]

},

{

"id": "Elicit.Slot.61803317158.47237465252",

"variations": [

{

"type": "PlainText",

"value": "What would you like to be reminded of?"

},

{

"type": "PlainText",

"value": "Great. What message would you like in the reminder?"

}

]

},

{

"id": "Confirm.Slot.61803317158.47237465252",

"variations": [

{

"type": "PlainText",

"value": "The message. {message} will be sent as a reminder"

}

]

},

{

"id": "Confirm.Intent.1350208148512",

"variations": [

{

"type": "PlainText",

"value": "Sure, I will remind you to {reminder}"

}

]

},

{

"id": "Elicit.Slot.1308217539947.242246796500",

"variations": [

{

"type": "PlainText",

"value": "I can remind you to exercise or meditate at this time tomorrow. Which will it be?"

}

]

},

{

"id": "Elicit.Slot.1216715640341.1373587319490",

"variations": [

{

"type": "PlainText",

"value": " You can choose to target your lower body, upper body or complete a full body workout. Which will it be?"

}

]

},

{

"id": "Elicit.Slot.1346554512009.601913546176",

"variations": [

{

"type": "PlainText",

"value": "You can choose a light, moderate or vigorous workout. Which intensity would you like?"

}

]

},

{

"id": "Slot.Validation.1198761464102.1268688566038.1021182339498",

"variations": [

{

"type": "PlainText",

"value": "Sorry, I didn't quite get that. You can get started with a quick ten minute workout, unwind with ten minutes of meditation or follow a ten minute guided breathing session. Which will it be?"

}

]

},

{

"id": "Slot.Validation.1346554512009.601913546176.472635673855",

"variations": [

{

"type": "PlainText",

"value": "Sorry, I don't know that one. You can choose a light, moderate or vigorous workout. Which intensity would you like?"

}

]

},

{

"id": "Slot.Validation.1216715640341.1373587319490.55628938039",

"variations": [

{

"type": "PlainText",

"value": "Sorry, I don't know that one. You can choose to target your lower body, upper body, or complete a full body workout. Which will it be?"

}

]

}

]

}

}