



Knowledge Transfer Virtual Workshops

Improve your business operations by enabling a SAP Conversational AI chatbot

Build a chatbot from scratch consuming an external service with SAP Conversational AI

Mariajosé Martínez

Platform & Technologies Solution Engineer

October 7, 2020

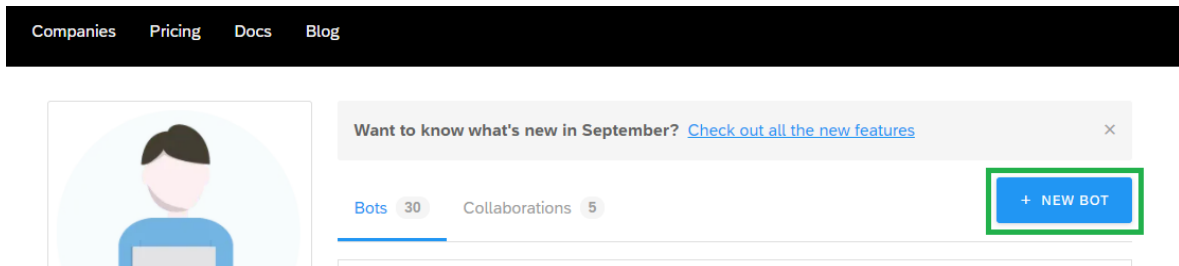
Build a chatbot from scratch consuming an external service with SAP Conversational AI

Prerequisites:

- Have an SAP Conversational AI account or trial account.

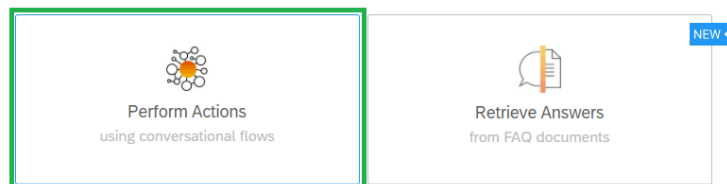
Let's start now 😊

- Create a new bot.

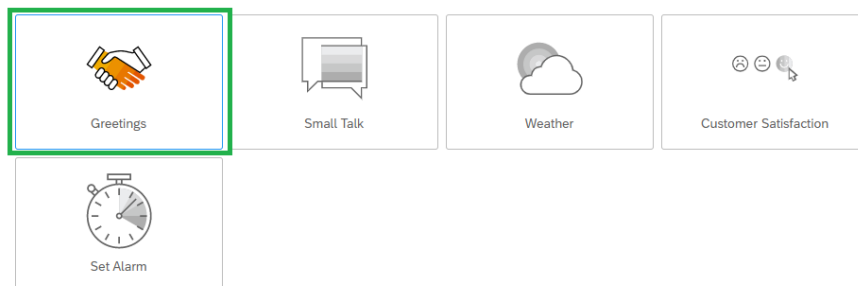


- Select **Perform Actions** and **Greetings** as the predefined skill for your bot.
- **Give the bot a name**, in my case I'll put *bestrunbot*.

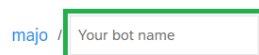
1 What do you want your chatbot to do?



2 Select predefined skills for your bot

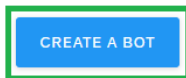


3 Create your bot



- Add a **description** and **topics** if you want to.
- Select **English** as the language.
- Select **Non-personal** in the **Data Policy**.
- **Store** the conversation data and,
- Select **Non-vulnerable**.
- Make your bot **Public** or **Private** as you wish.


Then click on **Create a bot**.




Now make sure you're in the **Intents** tan inside the **Train** tab. Let's **create an Intent** for Categories available with Products for sale.

- Give the intent a name, I'm naming it: **categoriesforsale** and create it.

- Add the following expression: **See Categories for Sale**.

No description provided 

 Intent settings

 1
 EN 1

★ TIPS

Hey there! I can help you to improve your bot, please follow the instructions to train your bot well!



+ ADD LANGUAGE

 FORK

Add an expression

IMPORT A CSV FILE

You have 2 expressions suggested to enrich your intent



All



☐ See Categories for Sale



Now let's go to the **Entities** tab. You can go back to the previous tab or click again or click in the **Train** tab.

- Create a **Restricted entity** and give the name of **category**.

Create a custom entity

majo / bestrunbot /

category

☒ **Restricted entity**

Useful if you have a strict list of words to detect and don't need automatic detection of entity.

#PIZZA

☐ cheese
 ☐ calzone
 ☐ pepperoni

or

☐ **Free entity**

Useful if you don't have a strict list of values and want machine learning to detect all possible values.

#PHONE-TYPE

☐ I want to buy an iPhone
☐ My Huawei is broken
 ☐ I'd like to have the new Samsung phone

A detected entity with a confidence score strictly greater than the matching strictness is kept and returned in the JSON. Our recommendation is a matching strictness between 50 and 70.

Matching strictness

0

90

100

Reset to default matching strictness

Cancel

CREATE

- Enter your entity just created and add the following categories as values:
 - Food
 - Beverages
 - Electronics

bestrunbot / entities / #CATEGORY RESTRICTED [FORK](#)


Entity settings

★ **TIPS**
Hey there! I can help you to improve your bot, please follow the instructions to train your bot well! →

List of values Tagged values Enrichments

Your entity training [HOW DOES IT WORK?](#)









Here, you can add entity values to improve the detection of your custom entity.



Search

List of values case-insensitive 20 per page

Add a new entity value [FETCH VIA SERVICE API](#) [IMPORT A CSV FILE](#)

<input type="checkbox"/>			
<input type="checkbox"/>	electronics		
<input type="checkbox"/>	beverages		
<input type="checkbox"/>	food		

Let's create another entity for products, but in this case, we are going to fetch the data from an external API service.

- Go back to **Entities** and click on **create** a new one.
- Put **product** as the name and select the **restricted** option.
- Click on **Fetch via service API**.

List of values case-insensitive 20 per page

Add a new entity value [FETCH VIA SERVICE API](#) [IMPORT A CSV FILE](#)

- Select **GET** and paste the following endpoint:
[https://services.odata.org/V3/\(S\(vnym1b3ehndm0p4fr0bdtbon\)\)/OData/OData.svc/Products/?\\$format=json](https://services.odata.org/V3/(S(vnym1b3ehndm0p4fr0bdtbon))/OData/OData.svc/Products/?$format=json) (this is a public OData service, if you want to explore it, feel free 😊)
- Click on **Fetch**

- ☒ **Replace**
to replace with the fetched entity values

☐ **Merge**
to add fetched entity values to the bot's existing entity values

You should be able to see the values imported like this:

bestrunbot / entities / #PRODUCTS RESTRICTED

Entity settings

★ TIPS
Hey there! I can help you to improve your bot, please follow the instructions to train your bot well!

List of values Tagged values Enrichments

Your entity training HOW DOES IT WORK?

Here, you can add entity values to improve the detection of your custom entity.

Search

List of values case-insensitive 20 per page

Add a new entity value

FETCH VIA SERVICE API IMPORT A CSV FILE

<input type="checkbox"/>	bread	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	coffee	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	cranberry juice	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	dvd player	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	fruit punch	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	havina cola	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	lcd hdtv	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	lemonade	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	milk	<input type="checkbox"/>	<input type="checkbox"/>

Now let's build the skills of the bot.

- Click on the **Build** tab and **Add a skill**

majo / bestrunbot DEVELOPMENT v1

No description provided

Add topics

Train Build Connect Monitor

Add

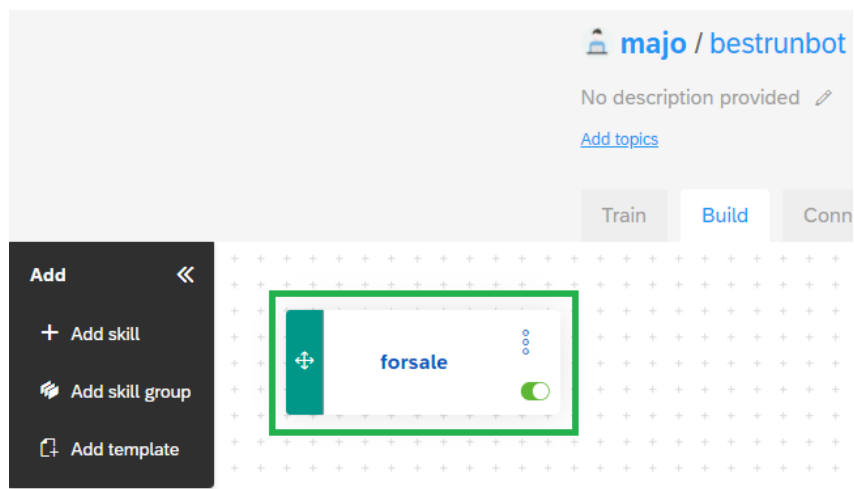
- + Add skill
- Add skill group
- Add template

- Give the skill a name, I'm naming it **forsale**.
- Select a **Business** skill and create it.

The 'Add Skill' dialog box is shown with the following fields and options:

- Name:** A text input field containing 'forsale'. A character count '248' is visible on the right. Below the field is the example text 'ex. book-flight, checkout'.
- Type:** Two buttons are present: 'Business' (which is highlighted with a green border) and 'Floating'.
- Activation:** A toggle switch is turned on (green).
- Title:** A text input field with a placeholder 'Type a short title for this skill's quick reply button...'. Above the field is a small flag icon.
- Buttons:** At the bottom right, there are 'Cancel' and 'Add' buttons. The 'Add' button is highlighted with a green border.

You'll see your skill created like this:



Now click on it and go to the **Triggers** tab.

- Add the **categoriesforsale** intent as a trigger for this skill.

bestrunbot / forsale BUSINESS < > ⚙️ 🔗 FORK

README.md **Triggers** Requirements Actions

The skill will be triggered only if:

🇬🇧 HOW DO TRIGGERS WORK?

If @categoriesforsale is-present 🗑️ +

+

Now let's go to the **Requirements** tab and,

- Add **category** as entity and give it the same name to save it in the memory.
- Click on *if #category is missing* to add a custom message fetching data from an external service.

bestrunbot / forsale BUSINESS < > ⚙️

README.md Triggers **Requirements** Actions

REQUIRED INFORMATION *The skill requires this information before executing Actions* 🔗 WHAT'S A REQUIREMENT?

🇬🇧

#category as category ⬆️

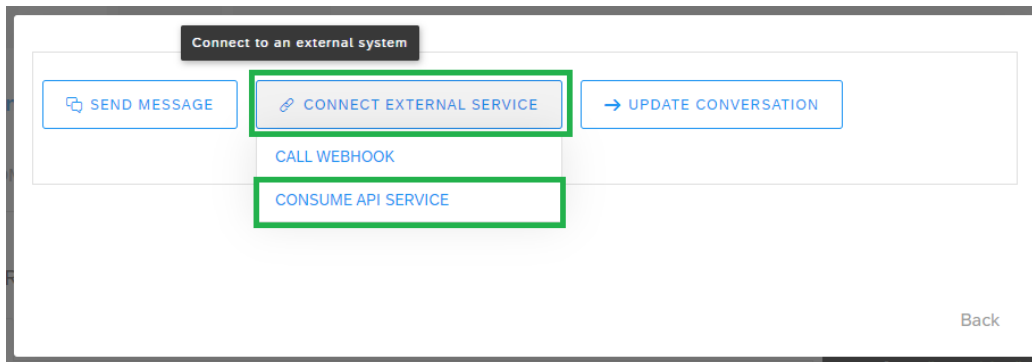
+ NEW REPLIES If #category is complete

+ NEW REPLIES **If #category is missing**

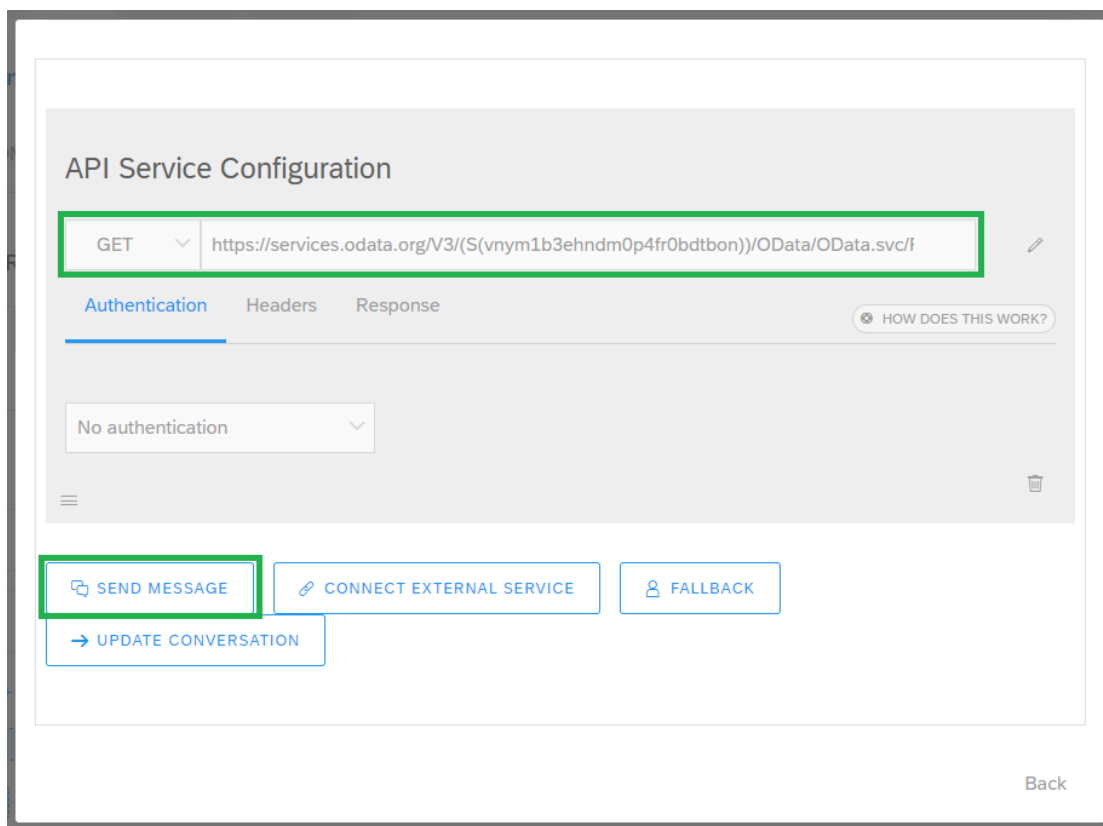
+ Add validators to handle errors 🗑️

+

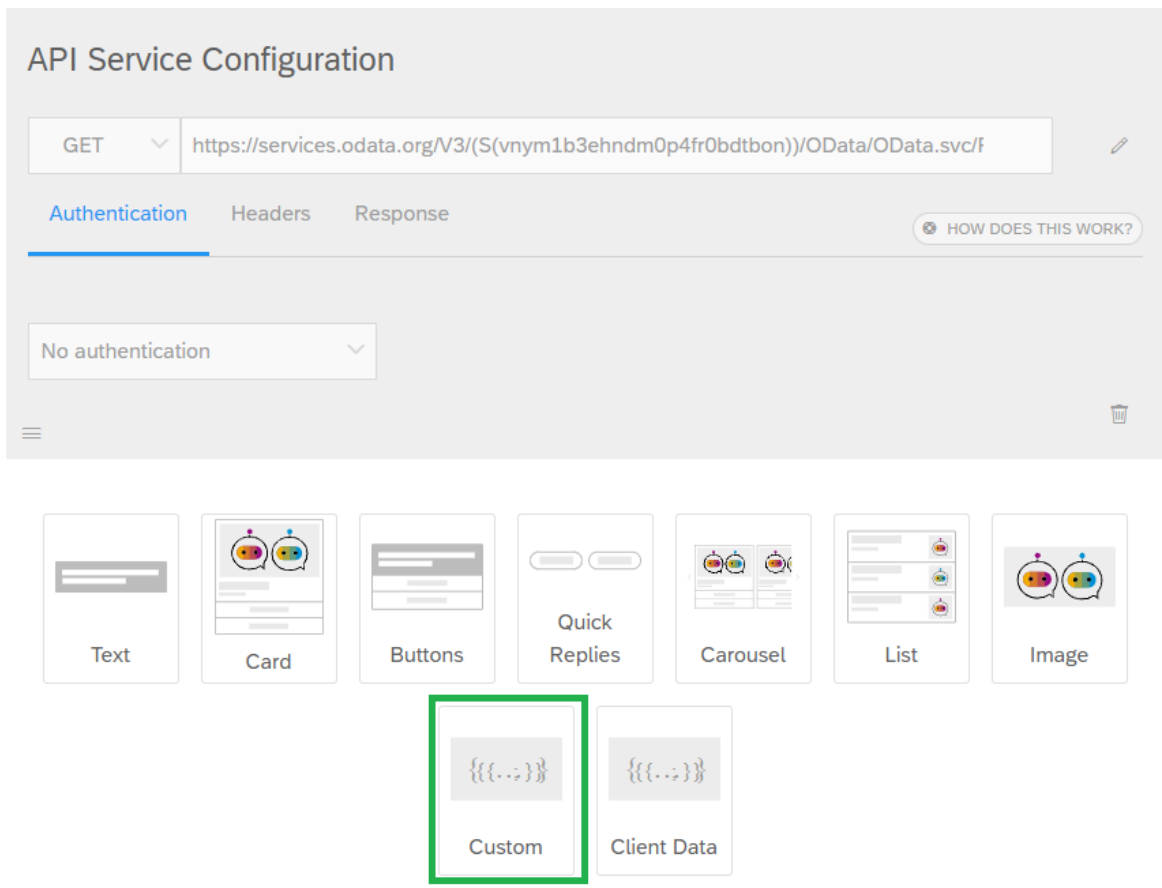
- Select **Connect External Service**.
- And select **Consume API Service**.



- Select a **GET** operation.
- Copy and paste this API endpoint:
[https://services.odata.org/V3/\(S\(vnym1b3ehndm0p4fr0bdtbon\)\)/OData/OData.svc/Categories/?\\$format=json](https://services.odata.org/V3/(S(vnym1b3ehndm0p4fr0bdtbon))/OData/OData.svc/Categories/?$format=json)



- Let's add a Script for a **Custom Message**.



- Select the **Message Type** as **Buttons** and,
- Add the following script:

```
{
  "type": "buttons",
  "delay": "",
  "content": {
    "title": " Please select the category you are interested in!",
    "buttons": [
      {{#eachJoin api_service_response.default.body.value}}
      {
        "title": "{{Name}}",
        "value": "{{Name}}",
        "type": "postback"
      }
    ]
  }
}
```

- And **save** it.

API Service Configuration

GET

Authentication Headers Response

No authentication

HOW DOES THIS WORK?

Message Type: Buttons

Response Script

```

1 {
2   "type": "buttons",
3   "delay": "",
4   "content": {
5     "title": "Please select the category you are interested in!",
6     "buttons": [
7       {{#eachJoin api_service_response.default.body.value}}
8       {
9         "title": "{{Name}}",
10        "value": "{{Name}}",
11        "type": "postback"
12      }{{/eachJoin}}
13     ]
14   }
15 }
16

```

HOW DOES THIS WORK? **SAVE**

Now let's add the product entity as a **new requirement** as we did with the category one.

Add a new requirement

Select **product** as the entity and put the same name on the right

#category as category

AND

#product as product

- Open it and click on *if #product is missing*

We are going to repeat some steps:

- Select **Connect External Service**.
- And select **Consume API Service**.
- Select a **GET** operation.
- Copy and paste this API endpoint:
[https://services.odata.org/V3/\(S\(vnym1b3ehndm0p4fr0bdtbon\)\)/OData/OData.svc/Products/?\\$filter=%20Categories/any\(Category:Category/Name%20eq%20%27{{memory.category.raw}}%27\)&&\\$format=json](https://services.odata.org/V3/(S(vnym1b3ehndm0p4fr0bdtbon))/OData/OData.svc/Products/?$filter=%20Categories/any(Category:Category/Name%20eq%20%27{{memory.category.raw}}%27)&&$format=json)

- Add a **text message** and paste the following script:

```
{{#if (length api_service_response.default.body.value)}} Here are {{pluralize 'product'
quantity=(length api_service_response.default.body.value) }}. {{/if}}
```

Note: this is to customize the message if we have more than one result, so it can be shown as a plural response.

- Add a **custom message**, select **List** as the message type and paste the following json:

```
{"type": "list",
  "content": {
    "elements": [
      {{#eachJoin api_service_response.default.body.value}}
      {
        "title": "{{Name}}",
        "imageUrl": "",
        "subtitle": "$ {{Price}}",
        "buttons": [{
          "title": "{{Name}}",
          "value": "{{Name}}",
          "type": "postback"
        }]
      } {{/eachJoin}}
    ]
  }
}
```

You should have it like this:

API Service Configuration

GET [https://services.odata.org/V3/\(S\(vnym1b3ehndm0p4fr0bdtbon\)\)/OData/OData.svc/](https://services.odata.org/V3/(S(vnym1b3ehndm0p4fr0bdtbon))/OData/OData.svc/)

Authentication Headers Response [HOW DOES THIS WORK?](#)

No authentication

+

`{{#if (length api_service_response.default.body.value)}} Here are {{pluralize 'product' quantity=(length api_service_response.default.body.value)}}. {{/if}}`

<> [Markdown syntax is disabled.](#)

+

HOW DOES THIS WORK?

SAVE

Message Type

List

Response Script

```
1 {"type": "list",
2   "content": {
3     "elements": [
4       {{#eachJoin api_service_response.default.body.value}}
5       {
6         "title": "{{Name}}",
7         "imageUrl": "",
8         "subtitle": "$ {{Price}}",
9         "buttons": [{
10          "title": "{{Name}}",
11          "value": "{{Name}}",
12          "type": "postback"
13        }]
14      } {{/eachJoin}}
15   ]
16 }
17
18
```

Now let's add a new skill for the checkout process.

- Click on the **Build** tab

- Add a **Business skill** and name it **checkout**.

Add Skill

Name
 247
ex: book-fight, checkout

Type

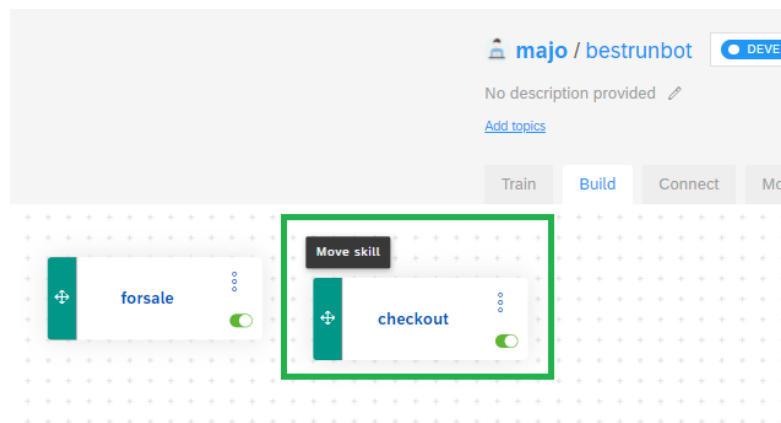
Business skills reflect the core purpose of your bot.

Activation
 Your skill is active. If you deactivate it, your bot will no longer use it when chatting. ☒

Title
 Your bot automatically uses this skill title when disambiguating during chat. If you don't specify one, your bot will display: `checkout`

Cancel


Move the skill by clicking in the cross and click it to train the skill:



- Go to the **Requirements** tab.
- Select the entity **Person** and put the same name to save it in the memory, we're going to ask for the user's full name.
- Click on *if #person is missing*.

REQUIRED INFORMATION The skill requires this information before executing Actions

 WHAT'S A REQUIREMENT?



#person

as

person

+ NEW REPLIES

If #person is complete

+ NEW REPLIES

If #person is missing

- Click on **send message**.
- Select **Text** as message type and,
- Copy and paste the following:
Could you please share with me your full name?
- **Save it.**

Could you please share with me your full name? 594

SAVE

☐ Enable Markdown syntax

Set a specific delay before sending next message

Now let's do the same to ask the user his/her email and location (address). Important fact: as you could see we didn't create an entity for person, email and location. This is because they are golden entities already predefined in SAP Conversational AI, and there are tons more.

- Click on the plus "+" to add new requirements

#person as person

+ NEW REPLIES If #person is complete

+ EDIT REPLIES If #person is missing

+ Add validators to handle errors

Add a new requirement

+

After adding email and location as requirements for the checkout you should see them like this:

AND #person as person

AND #email as email

+ NEW REPLIES If #email is complete

+ NEW REPLIES If #email is missing

+ Add validators to handle errors

AND #location as location

+ NEW REPLIES If #location is complete

+ NEW REPLIES If #location is missing

+ Add validators to handle errors

+

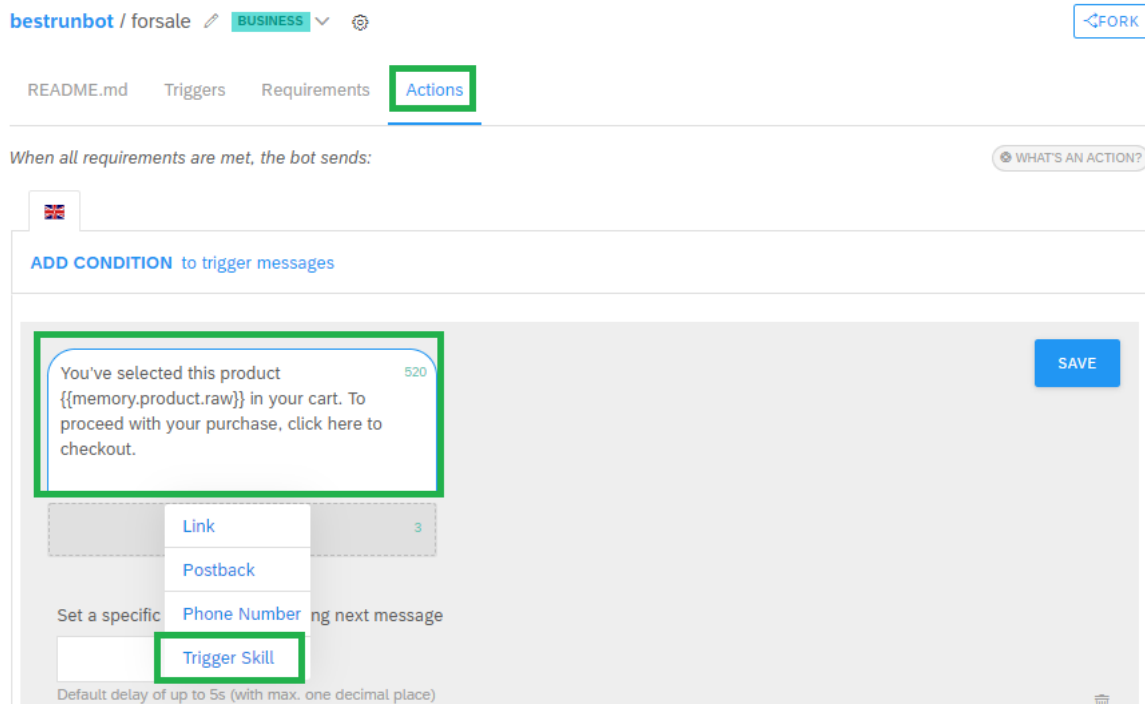
Now let's put the questions for each of them.

- Click on *if #email is missing* and paste the following as a **text** message:
And your email as well?
- **Save it**

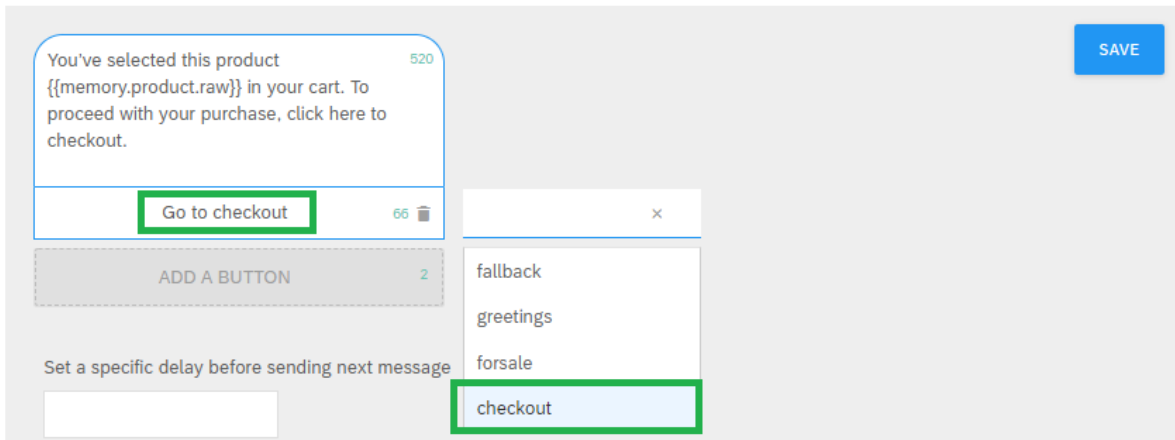
- Click on *if #location is missing* and paste the following also as text message:
Now I need your shipping address, could you please write it down?
- **Save** it.

Now let's go back to the **forsale** skill

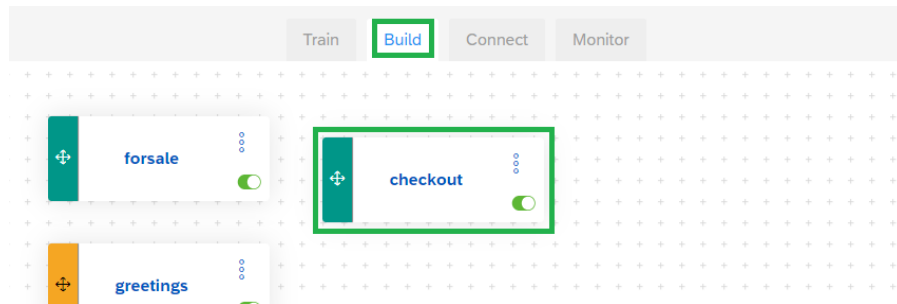
- Click on the **Build** tab
- Click on the **forsale** skill
- Go to the **Action** tab inside the skill and,
- Click on **Add new message group**
- Now click on **send message**
- Select **Buttons** as your message type and,
- Copy and paste the following:
You've selected this product `{{memory.product.raw}}` in your cart. To proceed with your purchase, click here to checkout.
- Click to **add a button** and select **Trigger Skill**



- In the **button title** put: Go to checkout
- And select **checkout** as the skill to trigger





Now let's complete for now the **checkout** skill. Let's go to the **Build** tab and click on the **checkout** skill.



- Go to the **Actions** tab inside the **checkout** skill
- Click on **add a new message group**
- Select **text** as message type
- And paste the following:
 Alright {{memory.person.raw}}, you've selected the product: {{memory.product.raw}}.
 We'll send you an email to {{memory.email.raw}} with the order confirmation which will
 be delivered to your shipping address: {{memory.location.formatted}}. Thank you for using
 BestRunMarket Bot! :)

When all requirements are met, the bot sends:

 WHAT'S AN ACTION?



ADD CONDITION to trigger messages

+

Alright {{memory.person.raw}}, you've selected the product: {{memory.product.raw}}. We'll send you an email to {{memory.email.raw}} with the order confirmation which will be delivered to your shipping address: {{memory.location.formatted}}. Thank you for using BestRunMarket Bot! :)


Note: We are going to leave the exercise 'til this point. However, you can use this as an example on how to build a commercial bot and continue integrating it with a backend server app or external service such as a payment motor to charge the order to the customer.

Now let's add the intent for asking *categories for sale* in the greetings' skill. Let's go to the **Build** tab and click on the **greetings** skill.

Train **Build** Connect Monitor


+

forsale




+

checkout



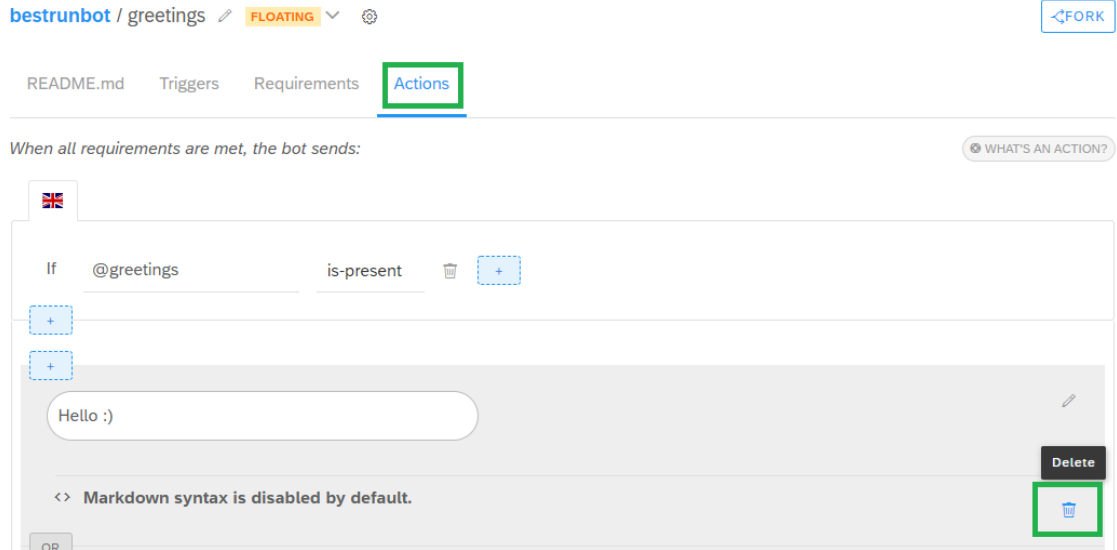
+

greetings

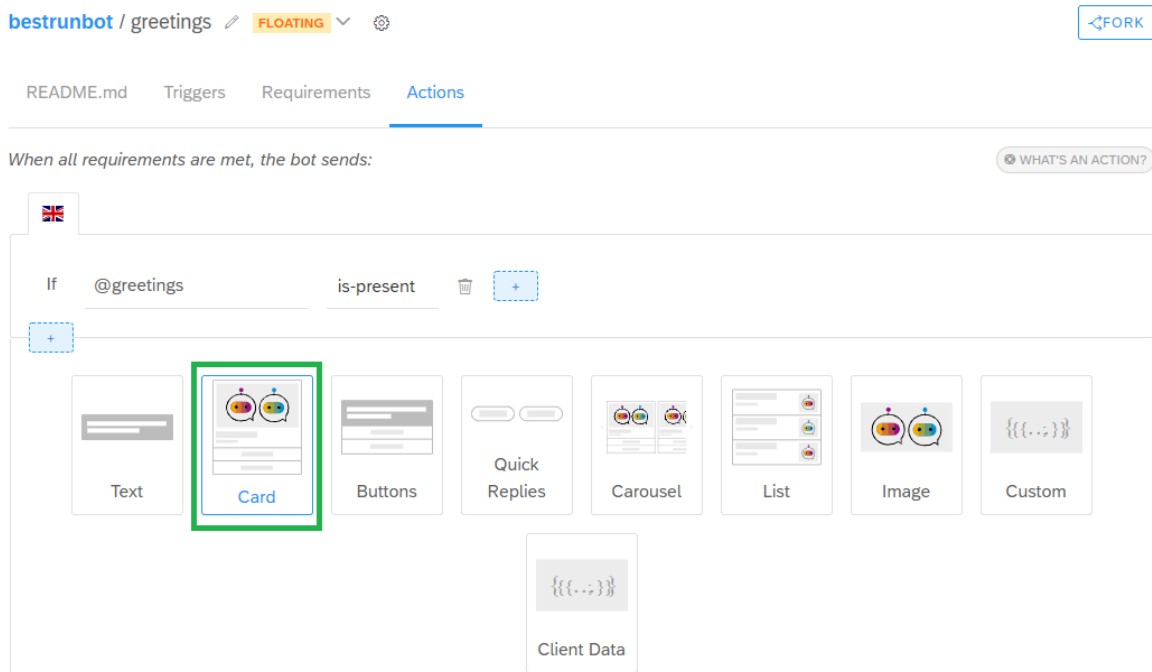


Go to the **Actions** tab. Notice that this skill and these responses are already configured because at the beginning we selected greetings as a predefined skill for the bot.

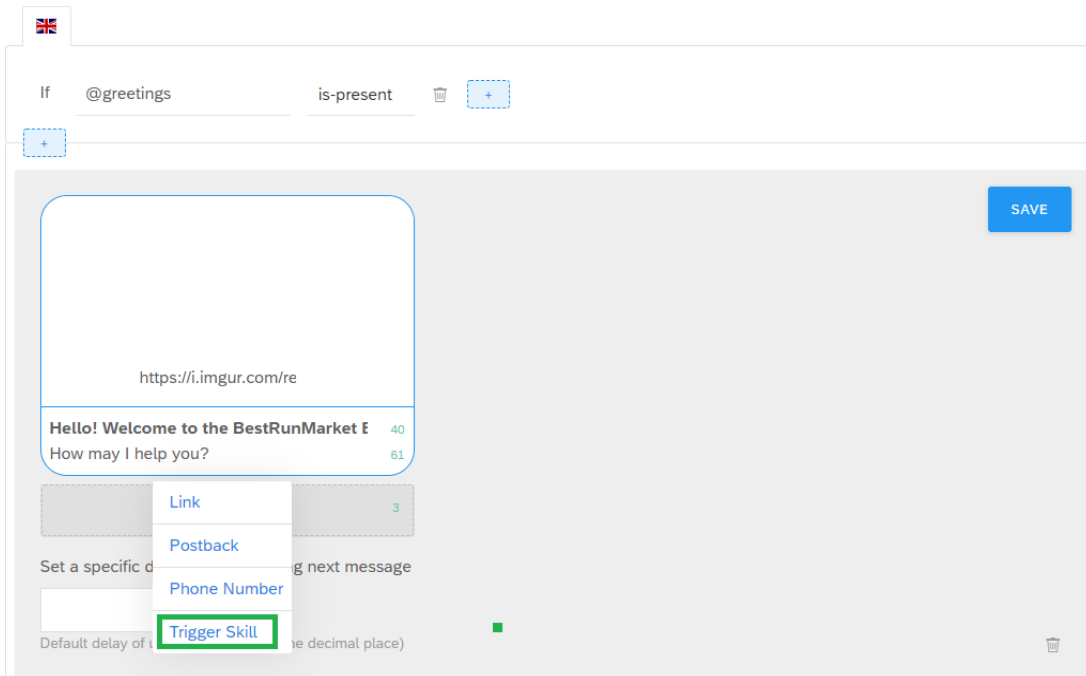
- **Delete** all the responses, we're going to put a new one.



- Click on **send message** and,
- Select **Card** as message type.



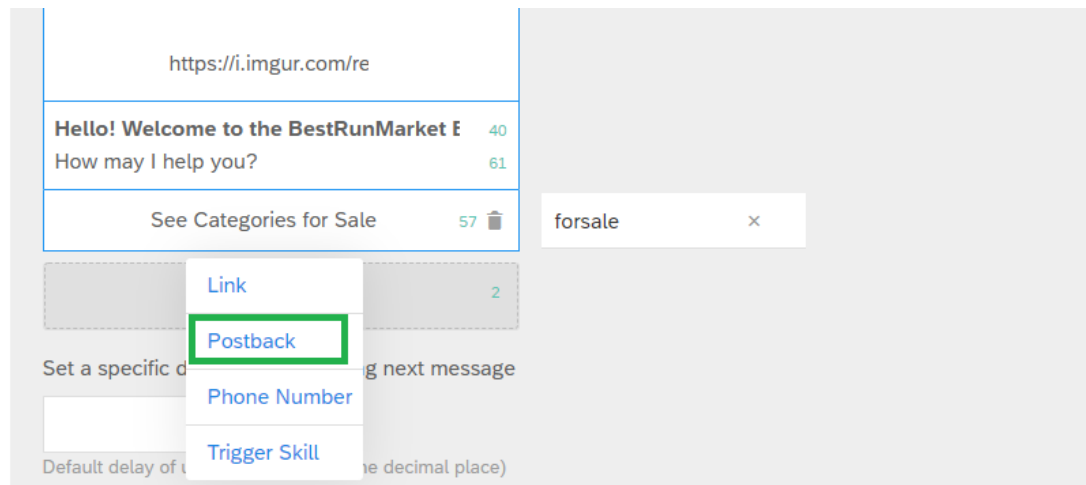
- As **image url**, put the following url with the BestRunMarket logo:
<https://i.imgur.com/reQirgK.png>
- In the **title** put: **Hello! Welcome to the BestRunMarket Bot!**
- **Subtitle:** **How may I help you?**
- Click to add a button and select **Trigger Skill**



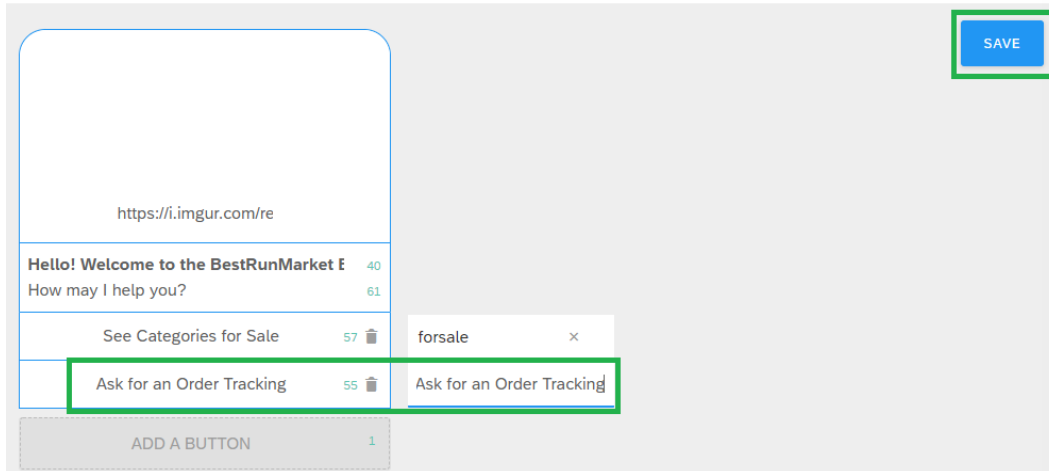
- In the **button title** put: [See Categories for Sale](#)
- And select the **forsale** skill.

Now let's add another button:

- Click to add another button and select **Postback**



- In the **button title** and **Postback value** put the same: [Ask for an Order Tracking](#) (we're going to leave it 'til here, using the intent *see categories for sale* as the main intent for this exercise... however this is an example of how we can put more options for the user to select and interact with the bot).
- **Save** it.

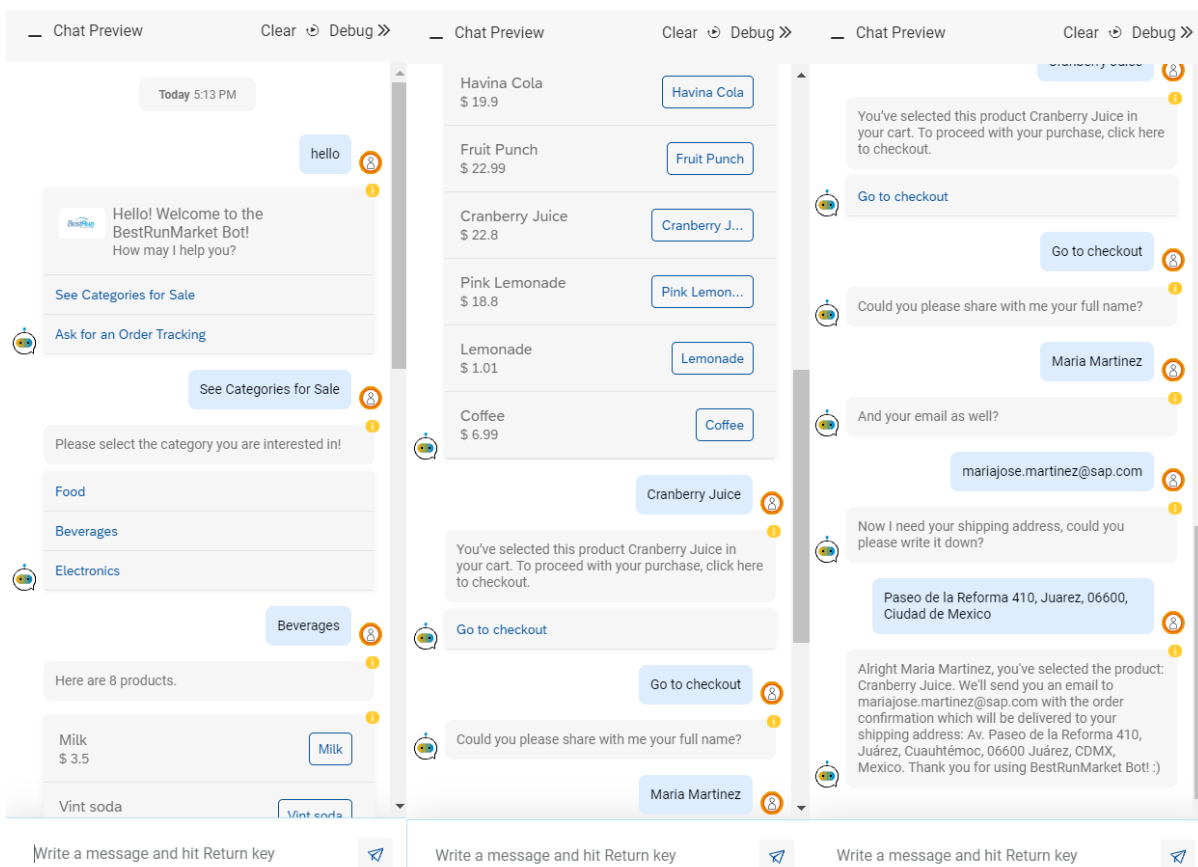


Now let's test the bot.

- Click on **Chat Preview**.



And test the bot as following:



To see the bot's memory, click on any yellow *i* in the conversation flow. For example, by clicking in the last one, you'll be able to see the whole entities saved in the bot's memory, like this:



```
47 }
48 },
49 "conversation": {
50   "id": "test-1601849579149",
51   "language": "en",
52   "memory": {
53     "category": {
54       "confidence": 0.99,
55       "raw": "Beverages",
56       "value": "beverages"
57     },
58     "product": {
59       "confidence": 0.99,
60       "raw": "Cranberry Juice",
61       "value": "cranberry juice"
62     },
63     "person": {
64       "confidence": 0.99,
65       "raw": "Maria Martinez",
66       "fullname": "Maria Martinez"
67     },
68     "email": {
69       "confidence": 0.99,
70       "raw": "mariajose.martinez@sap.com",
71       "domain": "sap.com",
72       "tag": null,
73       "local": "mariajose.martinez"
74     },
75     "location": {
76       "confidence": 0.55,
77       "raw": "Paseo de la Reforma 410, Juarez,
78         06600, Ciudad de Mexico",
79       "country": "mx",
80       "lng": -99.17075919999999,
81       "city": "Juárez",
82       "formatted": "Av. Paseo de la Reforma 410
83         , Juárez, Cuauhtémoc, 06600 Juárez,
84         CDMX, Mexico",
85       "street_number": "410",
86       "place": "ChIJiamSeUn_0YURot59vwT3Qts",
87       "state": "CDMX",
88       "type": "street_address",
89       "postal_code": "06600",
90       "lat": 19.4249823,
91       "street_name": "Av. Paseo de la Reforma"
92     },
93     "skill": "checkout",
94     "skill_occurrences": 4,
95     "skill_id": "5d659788-ca02-4810-bad8
96       -ed0ed02391fb"
97   },
98   "logs": {
99     "input": "Paseo de la Reforma 410, Juarez,
100       06600, Ciudad de Mexico",
101     "logs": [
```

Hope you'd enjoyed this hands-on exercise! 😊