# RECIMĒ DOCUMENTATION

Recimē is an end-to-end AI chatbot solution to automate customer support and content distribution. It gives you the tools to build omni-channel chatbots quickly and easily and eventually measure growth and user engagement to save cost and increase customer satisfaction.

|  |  |  |
| --- | --- | --- |
| Plan | Price |  |
| Starter  Good for community and OS projects. Comes with community support and Recime branding. | $0.00  (monthly) | **Our plan** |
| Personal  Great for personal use and freelancers. Comes with chatbot re-branding ability and community support. | $19.99  (monthly) | Upgrade |
| Business  Tailored for businesses of all sizes. Includes enough API calls and 5 credits / month support. | ---Contact |  |
| Enterprise  Everything you need to scale our chatbot at maximum performance. D+1 / 4hr support option available. | --- Contact |  |

# Developer Features

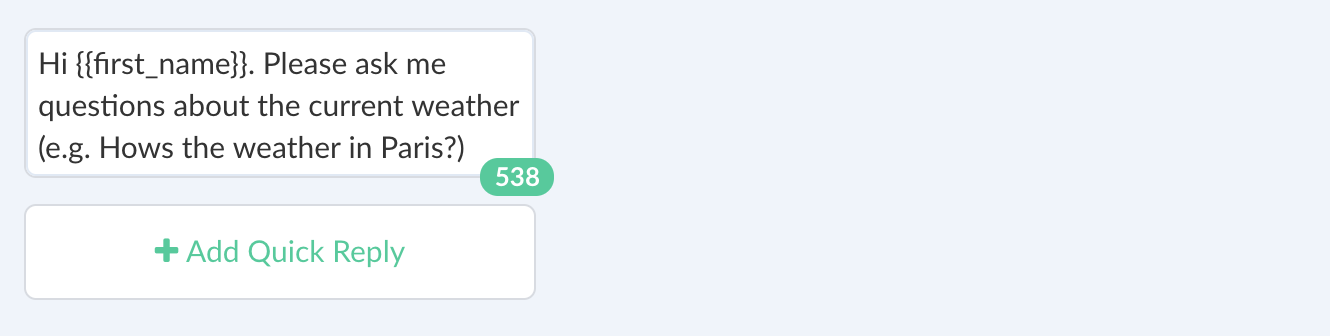
## User Variables

User variables are assigned by the underlying platform (e.g. Facebook) or by the user.

As our bot is connected to Facebook. It will assign the following variables:

**\*** first\_name **\*** last\_name

You can access a user variable using the double braces syntax in the following way:



It is also possible to set our own user variables that is available throughout the life-cycle of our bot.

A quick way of doing that is by using the script block. Copy and paste the following snippet into our script block to set “color”:

exports**.**handler **=** (context, done) **=>**

{

context**.**vars**.**set("color", "red");

done();

};

## JSON API

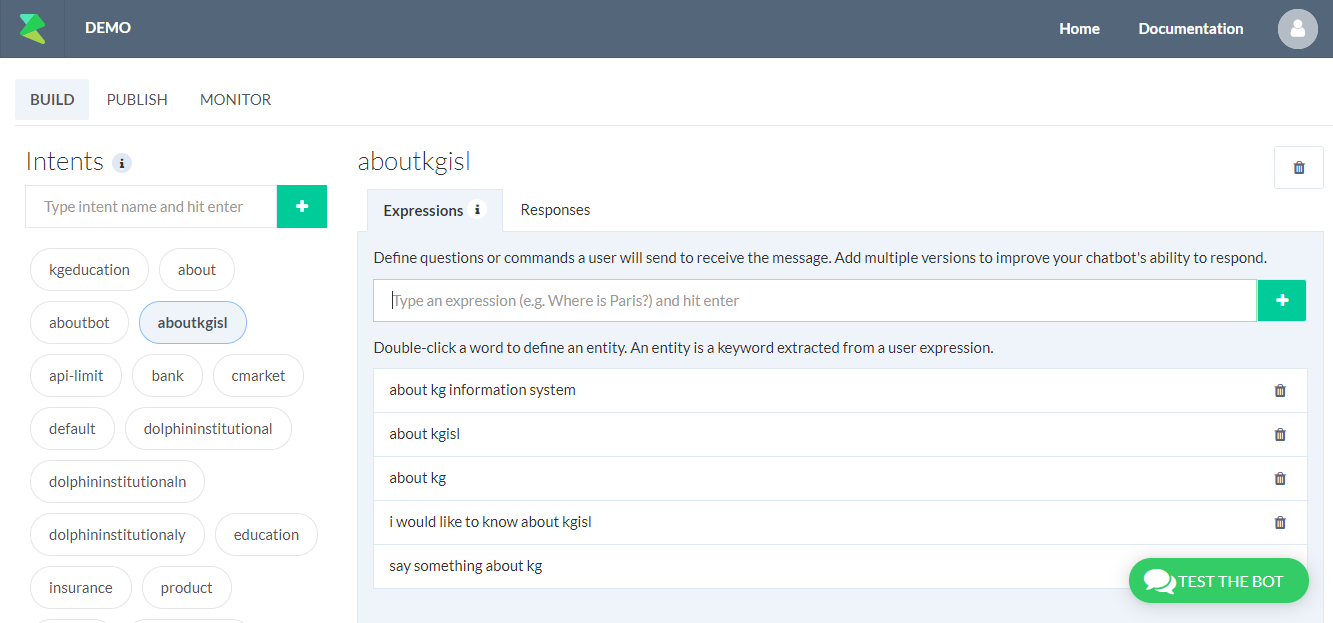
The plugin allows you to create server-side integrations or define our conversation logic based on dynamic content. You can make the following HTTP requests:

* GET
* POST
* PATCH
* DELETE
* PUT

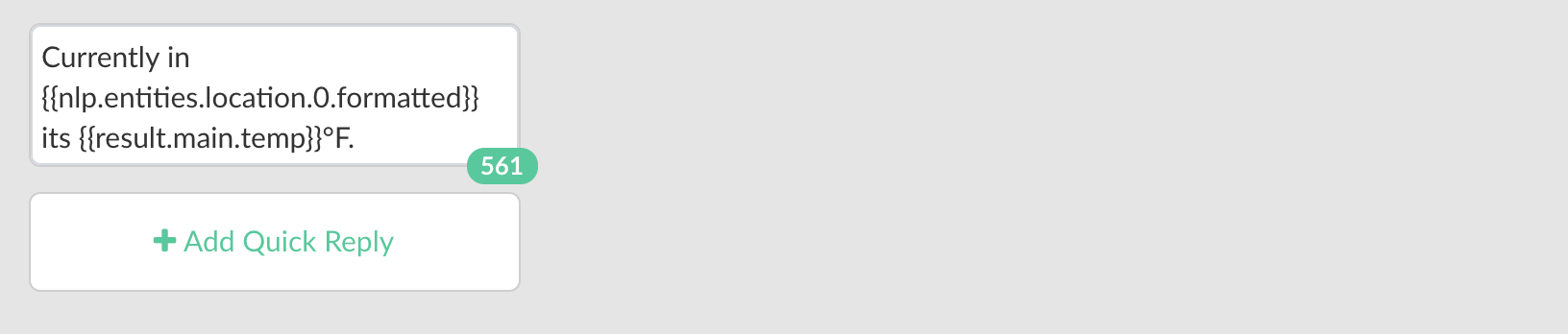
In addition, you can pass query string parameters, POST body, and headers to address a various type of custom implementation. It is possible to use user variables as parameters using the double braces syntax to pass user input and NLP response. Here is an example of how to use JSON API in order to get weather information:



Here is defined as an entity from the “Define Expressions” tab:



The response is saved in {{result}} varaible which is used from the text element to print out the current weather information in the following way:

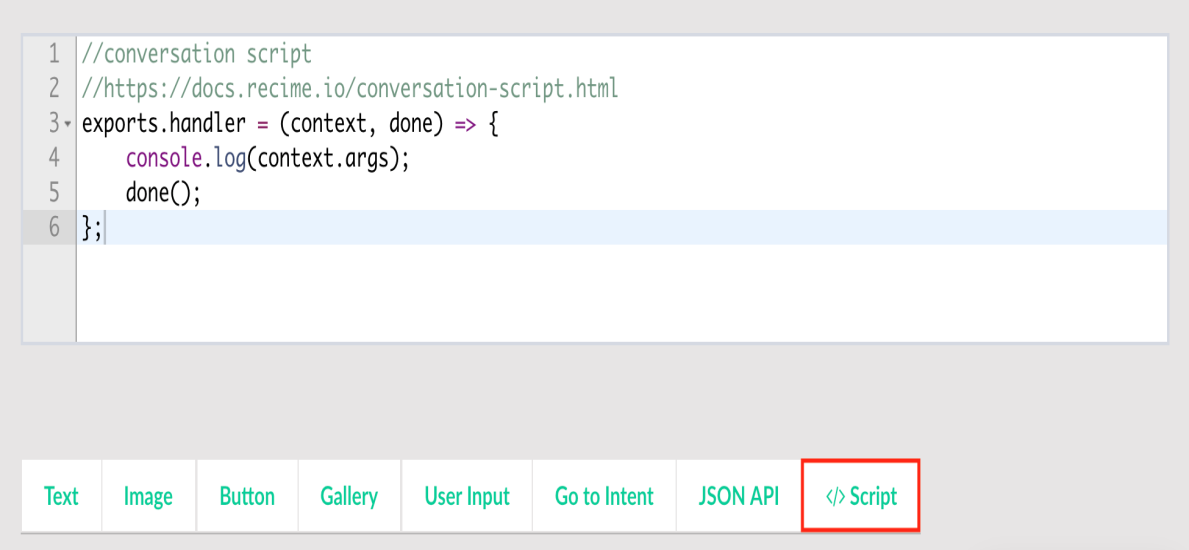


JSON API allows you to implement the following use-cases:

1. Retrieve dynamic content.
2. Create server-side integrations, where it will send data to an endpoint whenever a conversation reaches a certain point.
3. Trigger an event.

## Script Block

Script Block allows you to quickly write custom logic inside the conversation builder. Click on script button in the toolbar as shown below to insert a script block:



The entire conversation context is available in the script block. This will let you write custom logic based on user input, entities and variables. The example below shows how to pull data from a weather API and then set it as a context variable for use in the builder:

**//**script block

**//**https:**//**docs**.**recime**.**io**/**script**-**block**.**html

**import** request **from** 'request';

**//** get a free API key **from** openweathermap.org

const appId **=** 'PASTE\_OUR\_API\_KEY';

exports**.**handler **=** (context, done) **=>** {

request({

url : 'https://api.openweathermap.org/data/2.5/weather',

qs : {

lat : context**.**nlp**.**entities**.**location[0]**.**lat,

lon : context**.**nlp**.**entities**.**location[0]**.**lng,

appid: appId,

units: 'imeprial' },

json : true

}, (err, reponse, body)**=>**{

**if** (err){

console**.**log(err); }

**//**debug

console**.**log(body);

**//** save

context**.**vars**.**set("weatherData", body);

done(); });};

context parameter have the following properties:

|  |  |
| --- | --- |
| Property Name | Description |
| args | [User input](https://docs.recime.io/message-object.md) |
| nlp | Contains entities and intents for an expression. |
| vars | Set or get vars to use in the current context. |

You can use variables inside the builder in the following way using the double braces syntax:



Script block can be useful in the following scnearios:

* Retrieving data from our existing API that you want inject in our conversation context.
* Construct dynamic response based on user input and entities.

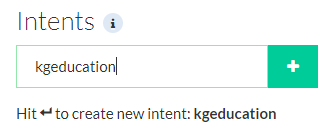
The following libraries are available in the **Script Block** runtime:

* [node core](https://nodejs.org/api/modules.html#modules_core_modules)
* [request](https://github.com/request/request)
* [recime-bot-extension](https://github.com/Recime/recime-bot-extension)
* [recime-keyvalue-store](https://github.com/Recime/recime-keyvalue-store)

## Capturing User Location

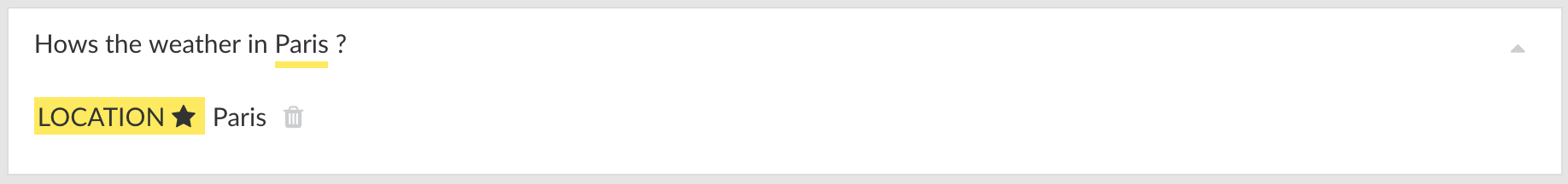
Facebook, Viber, and Telegram allow the user to send a location to our bot. It is sometimes useful to capture user’s location in order to book an appointment, order a pizza, etc.

When user sends a education names, it triggers a kgeducation intent or goes to the default intent. Create location intent by typing “location” into search box in the following way:



## Entities

An entity represents a term or object in the user's expression that provides clarification or specific context for a particular intent.



# Building a Bot

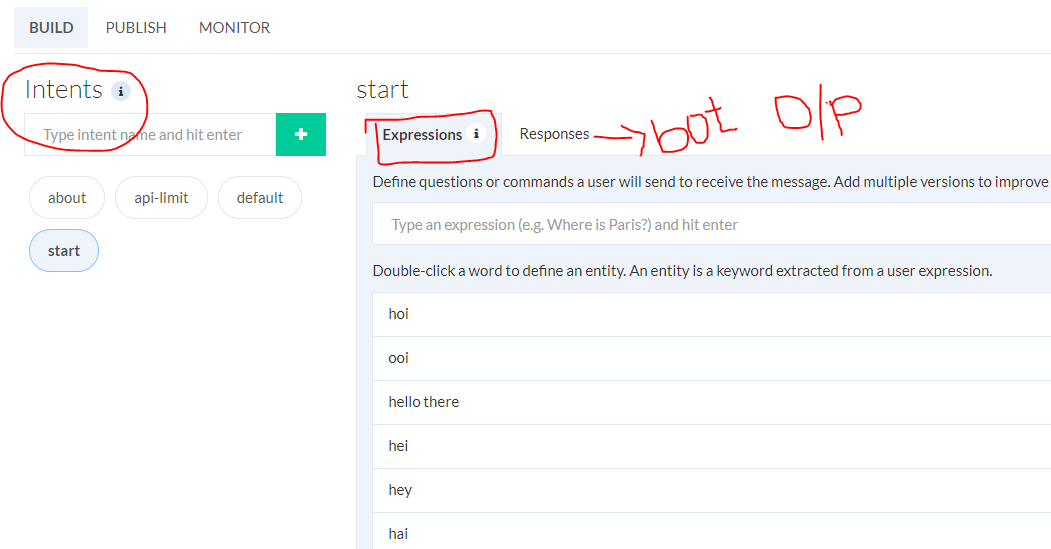
Step1:

Create Recime account using <https://console.recime.io/sign-in>

Step2:

Click create new bot button -> Give Title, description, Icon, and platform where we will use this bot (i.e webpage/messenger/telegram/slack etc.) ->Submit details.

Step3: Create intents (Expected questions from user) for our use case.



Step4:

Create response based on users questions (Intent).

We have many response types in Recime.io bot,

* Text
* Image
* Card
* Gallery
* Generic Template (Facebook)
* Article (WeChat)
* Audio
* Video
* User Input
* Go to Intent
* User Variable
* JSON API

Sample bot



