|  |  |
| --- | --- |
|  | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ \_\_Информатика и Системы управления\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

КАФЕДРА \_\_\_Информационная безопасность\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

***К КУРСОВОЙ РАБОТЕ***

***НА ТЕМУ:***

***Бот для корпоративного мессенджера Slack: анализ финансового рынка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

Студент \_\_ИУ8-32\_\_\_\_\_\_\_ **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_Головин Н.В.\_\_\_\_\_\_**

(Группа) (Подпись, дата) (И.О.Фамилия)

Руководитель курсовой работы **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_Колесников А.В.\_\_**

(Подпись, дата) (И.О.Фамилия)

Консультант **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

(Подпись, дата) (И.О.Фамилия)

*20 г.*

**Министерство науки и высшего образования Российской Федерации**

**Федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**«Московский государственный технический университет имени Н.Э. Баумана**

**(национальный исследовательский университет)»**

**(МГТУ им. Н.Э. Баумана)**

УТВЕРЖДАЮ

Заведующий кафедрой ИУ8

(Индекс)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ М.А. Басараб

(И.О.Фамилия)

« \_\_\_\_\_ » \_\_\_\_\_\_\_\_\_\_\_\_ 20 \_\_\_\_ г.

**ЗАДАНИЕ**

**на выполнение курсовой работы**

по дисциплине \_Алгоритмические языки\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Студент группы \_\_ИУ8-32\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_Головин Николай Васильевич\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Фамилия, имя, отчество)

Тема курсовой работы: ***Бот для корпоративного мессенджера Slack: анализ финансового рынка*** \_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Направленность КР (учебная, исследовательская, практическая, производственная, др.)

\_\_\_\_\_\_\_\_\_\_Практическая\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Источник тематики (кафедра, предприятие, НИР) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

График выполнения работы: 25% к \_\_\_ нед., 50% к \_\_\_ нед., 75% к \_\_ нед., 100% к \_\_\_ нед.

***Задание*** \_\_\_\_ создание на основе корпоративного мессенджера Slack бота, позволяющего получать данные о финансовом рынке \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

***Оформление курсовой работы:***

Расчетно-пояснительная записка на \_\_\_\_\_ листах формата А4.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата выдачи задания « \_\_\_ » \_\_\_\_\_\_\_\_\_\_\_\_ 2018\_\_ г.

**Руководитель курсовой работы**  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Колесников А.В.\_\_\_\_\_

(Подпись, дата) (И.О.Фамилия)

**Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_Головин Н.В.\_\_\_\_**

(Подпись, дата) (И.О.Фамилия)

Примечание: Задание оформляется в двух экземплярах: один выдается студенту, второй хранится на кафедре.

Бот для Slack

Цель работы заключается в создании на основе корпоративного мессенджера Slack бота, позволяющего сократить время решения каких-либо задач, лежащих перед конечным пользователем.

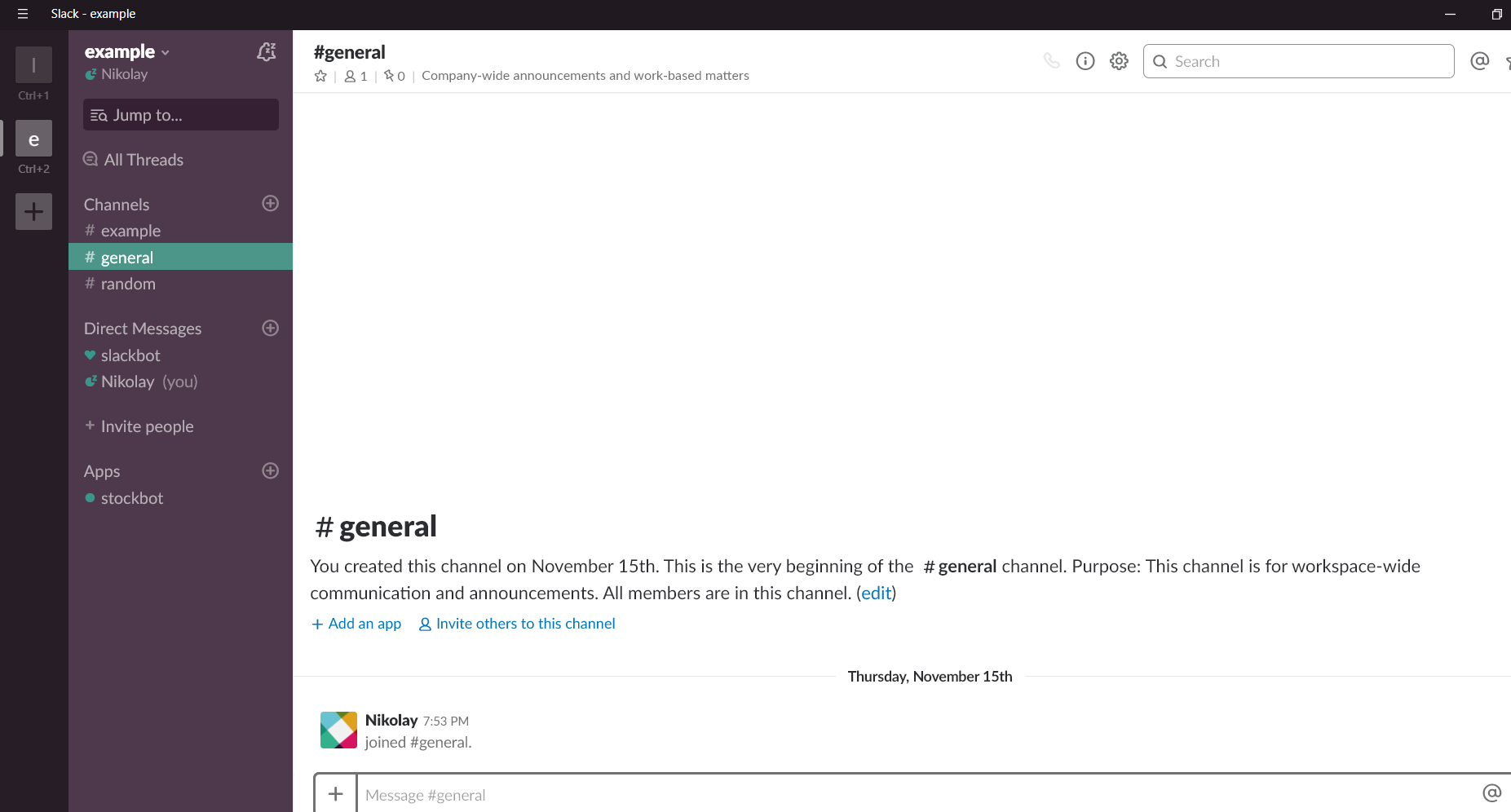
В данном отчете я привожу пример создания такого бота, который позволяет узнать биржевые цены на акции компаний, путем непосредственного обращения к этому боту.

Надо отметить, что использование бота не требует никаких специфических знаний (помимо знания английского языка), т.к. при его создании использовалось машинное обучение, позволяющее определять контекст обычного предложения. В результате обращение к боту не отличается от обращения к человеку, бот так же «понимает» собеседника.

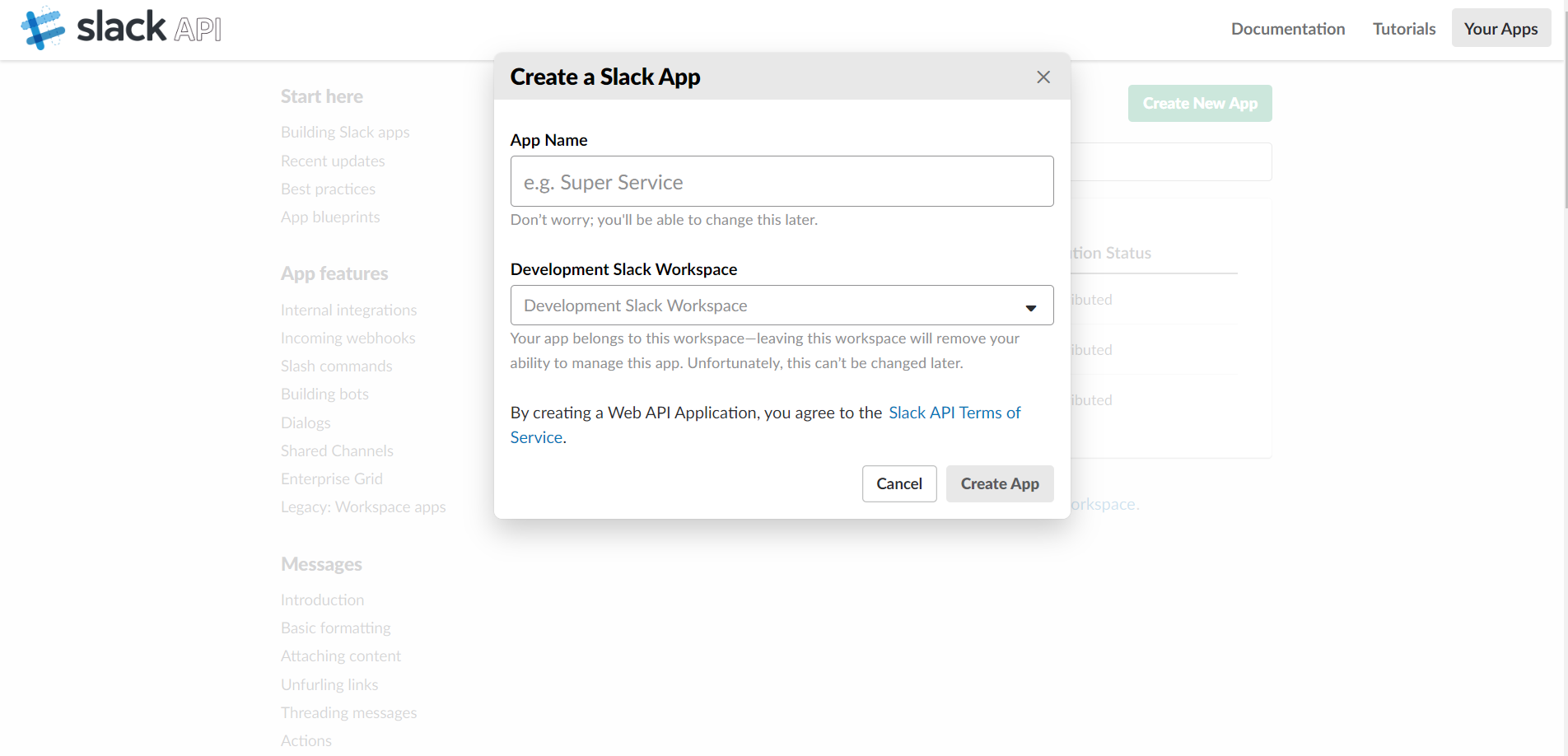
Данный бот не предусматривает какие либо ограничения ОС, не предусмотренные мессенджером Slack, т.к. он непосредственно его использует. Стоит отметить, что Slack доступен для загрузки практически для всех популярных платформ, как например Windows, Linux, MacOS, IOS, Android.

Разработка

Прежде всего нам необходимо создать рабочую среду:



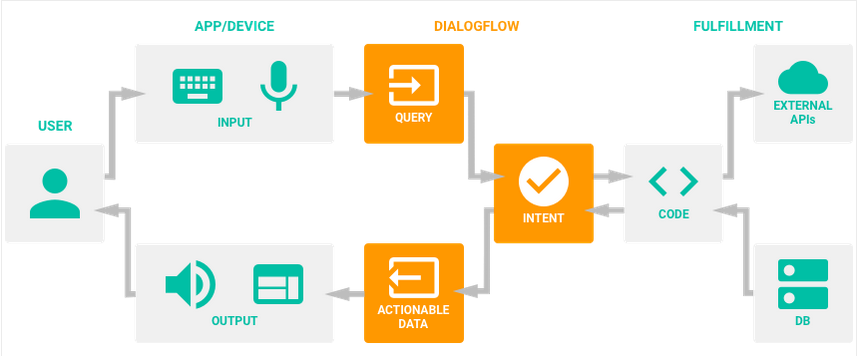
Далее создаем бота в нашем созданной рабочей среде:



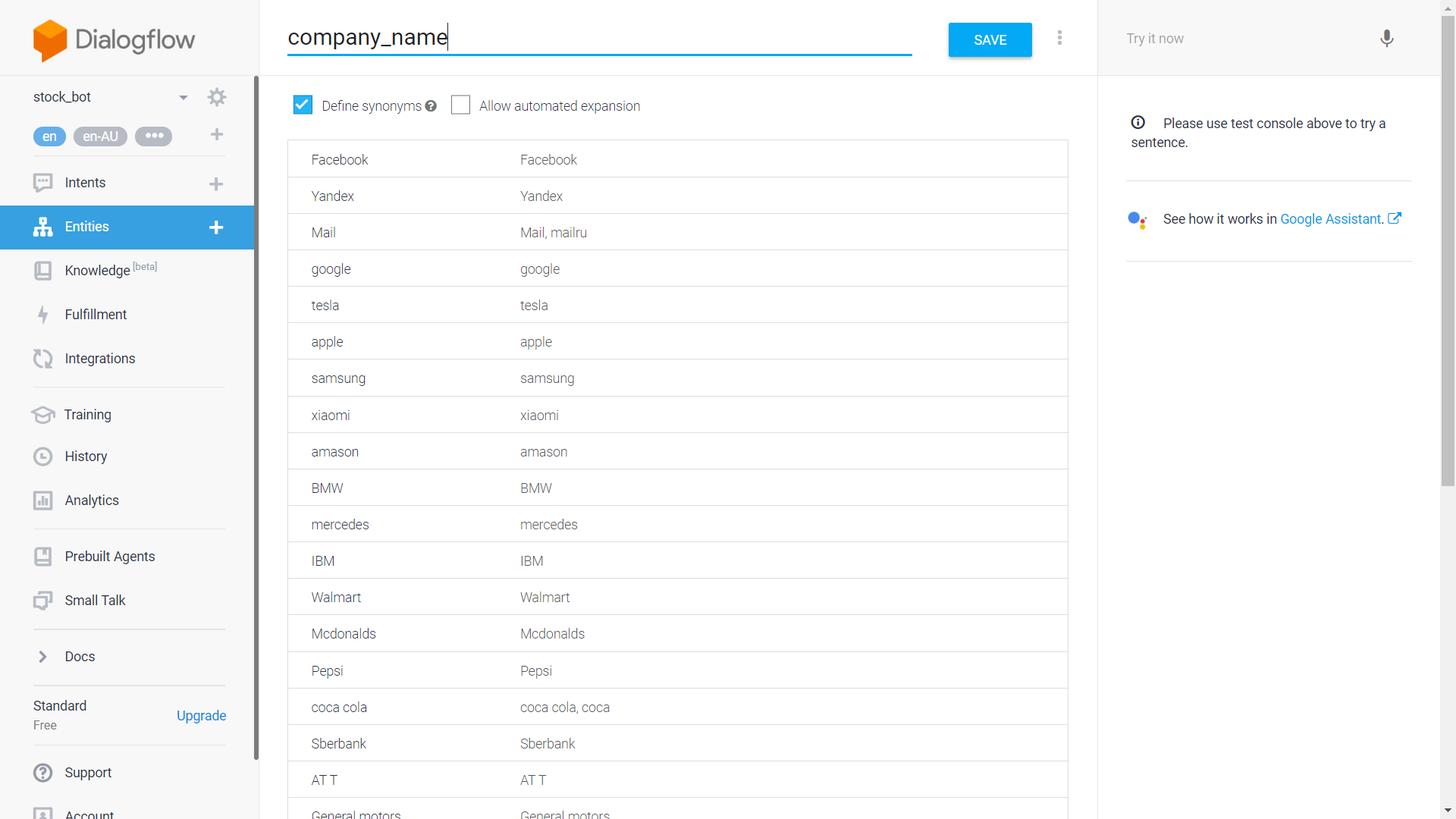
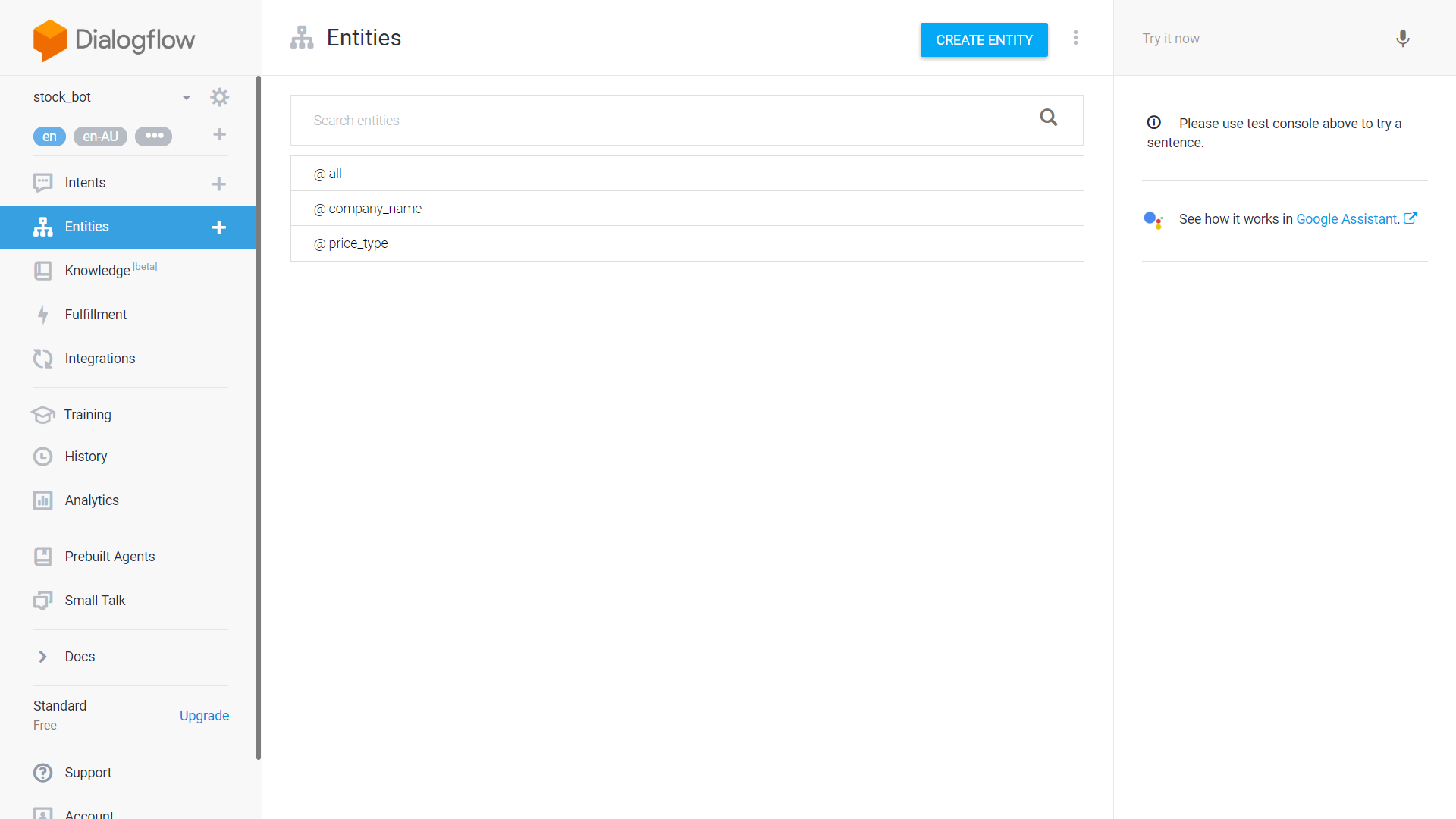
Формально бот создан, но он пока ничего не умеет. Чтобы он начал понимать наши запросы, надо запрограммировать его отдавать пользовательские запросы на аутсорс. В нашем случае таким «аутсорсом» будет выступать сервис от Google под названием DialogFlow. Его суть заключается в том, что этот сервис ориентируется на понимании человеческой речи с помощью машинного обучения. DialogFlow работает, создавая агентов − модули [NLU](https://en.wikipedia.org/wiki/Natural_language_understanding) (Natural Language Understanding). Они преобразуют естественные пользовательские запросы в управляемые данные.

Это преобразование происходит, когда введённое пользователем сообщение соответствует команде, которую ожидает программа.

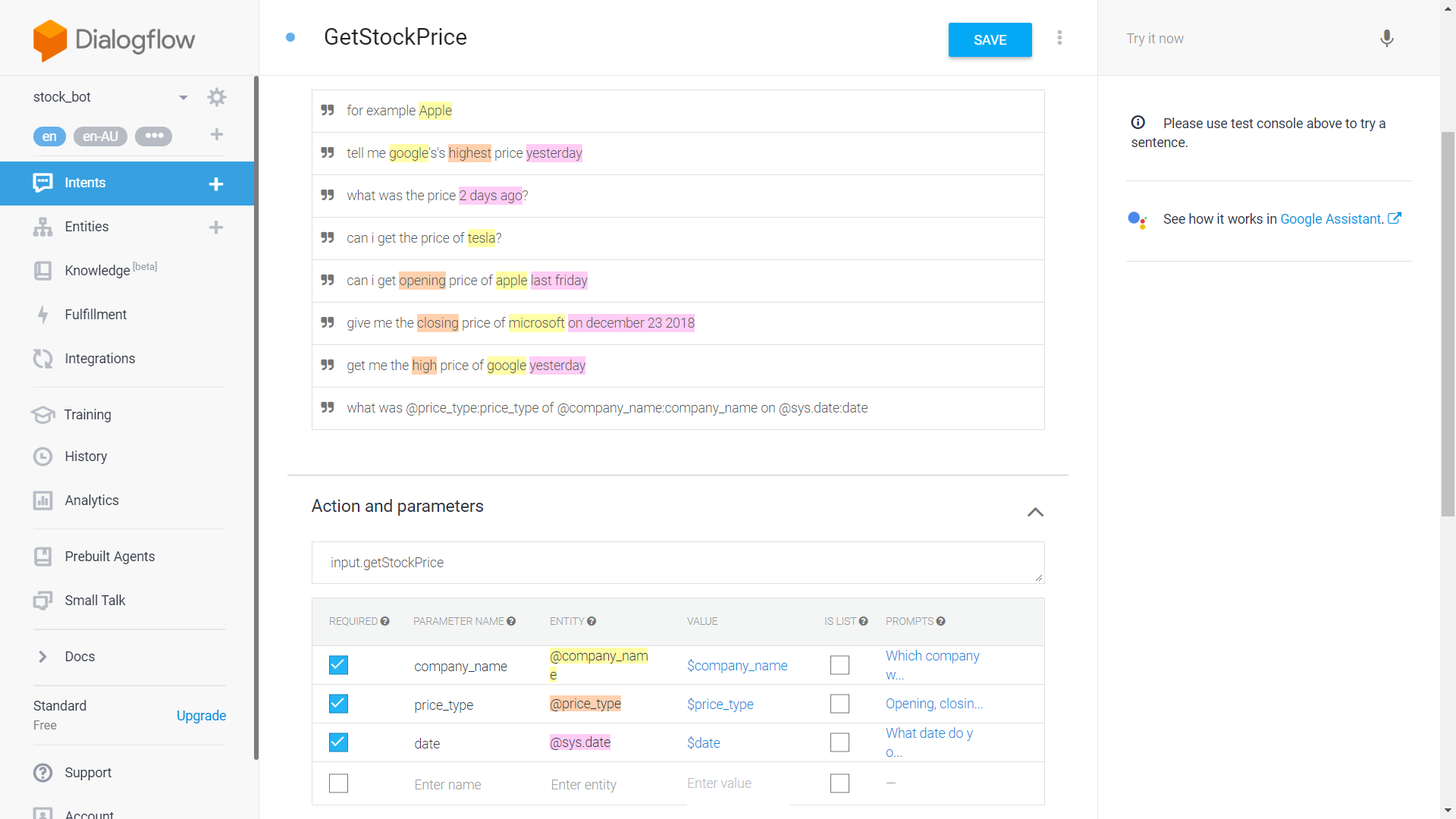
Например, вы говорите чат-боту, что вам холодно. Агент должен знать, что ему делать с этой информацией. Если он знает, что при холоде нужно дать сигнал вентилятору выключиться, он произведёт это действие.



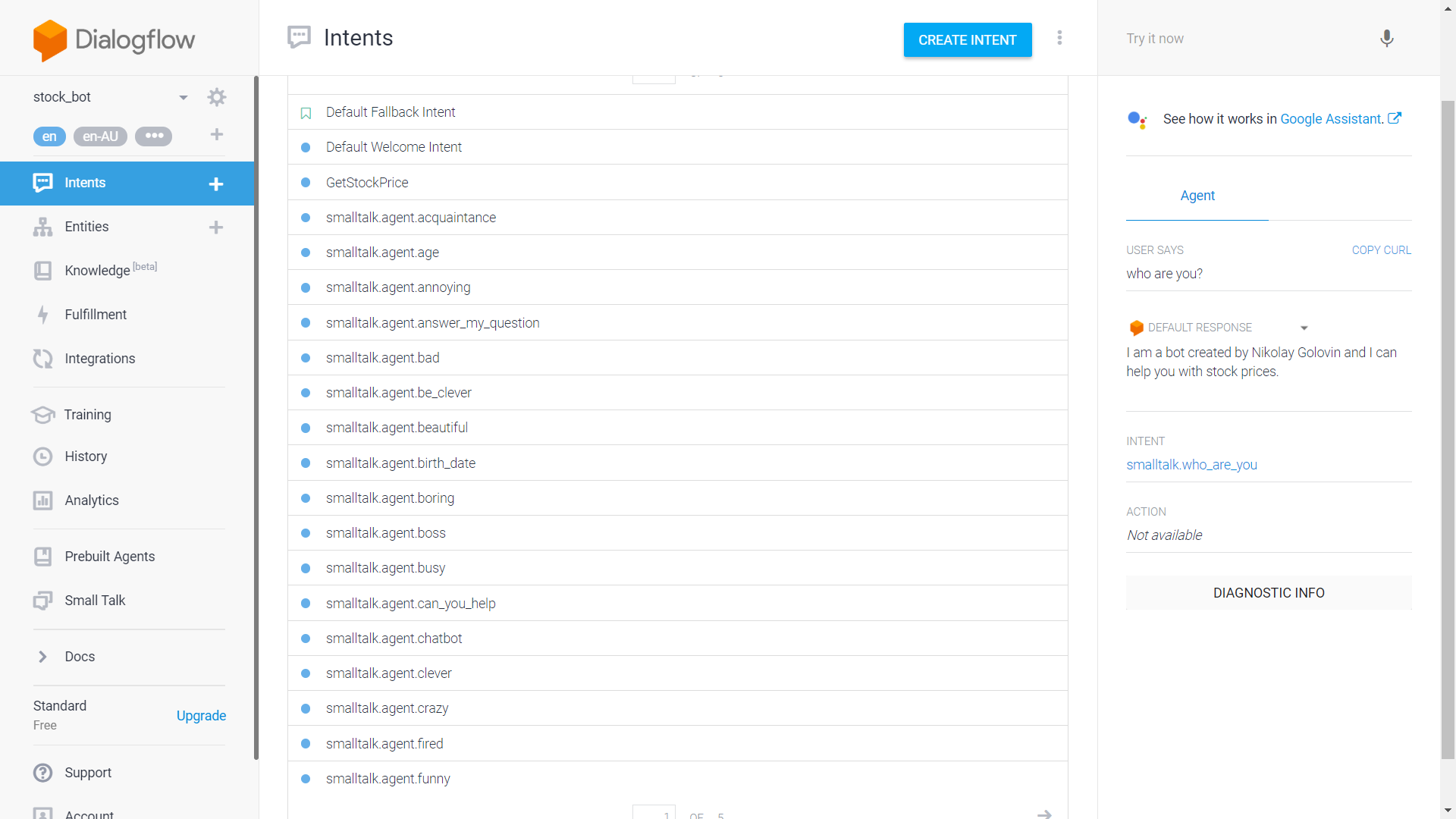
Итак, мы создали агента под названием stock\_bot. Как мы будем обучать его понимать человеческую речь? Сначала нам надо определиться с объектами с которыми мы будем иметь дело. Это название компании, тип цены (максимальная, минимальная и т.д.), и время. Они инициализируются во вкладке Entities (сущности). У меня они выглядят так:



Дальше нам необходимо настроить какие-то действия с ними. На помощь нам приходит вкладка Intents. Она отвечает за понимание написанного пользователем, и в зависимости от этого предпринимает какие-либо действия, будь то ответ заранее запрограммированной фразой, распознавание Entities (сущностей) или передача запроса в какие-либо сторонние API (рассмотрим такой случай позже). Для примера рассмотрим, как устроено «главное» у меня намерение getStockPrice:



Здесь мы видим тренировочные фразы – фразы максимально похожие на те, которые пользователь будет нам писать. Слова в них раскрашены в разные цвета, которые показывают, что эти слова были распознаны как ключевые, то есть несут полезную информацию, на основе которой нам и надо дать ответ. К примеру, здесь есть фраза «tell me google’s highest price yesterday», где безошибочно было определено, что нужно дать информацию по Компания: Google; какую цену хотим: highest; за какое время: yesterday. Кстати, чуть ниже как раз и определены те самые сущности, которые мы ищем в запросе. Таких «намерений» у меня получилось довольно много – около 80 штук. Они в основном отвечают за «мелкие» фразы, например «привет» или «как дела».



Мы поняли что написал пользователь, что нам дальше делать с этой информацией? Нам нужно откуда-то достать информацию об акциях, причем извне. Для этого в DialogFlow есть вкладка Fulfillment, позволяющая подключать различные сторонние API куда можно отправлять и принимать запросы. Я использовал сервис Intrinio, который как раз и создан для обработки и анализа финансового рынка через API.

Fulfillments работает на основе бэкэнд сервиса Гугл Firebase. Он много что умеет, но мы будем его для использовать лишь для отправки запроса на сервер Intrinio и получения ответа. Чтобы отправить запрос был написан код на JavaScript:

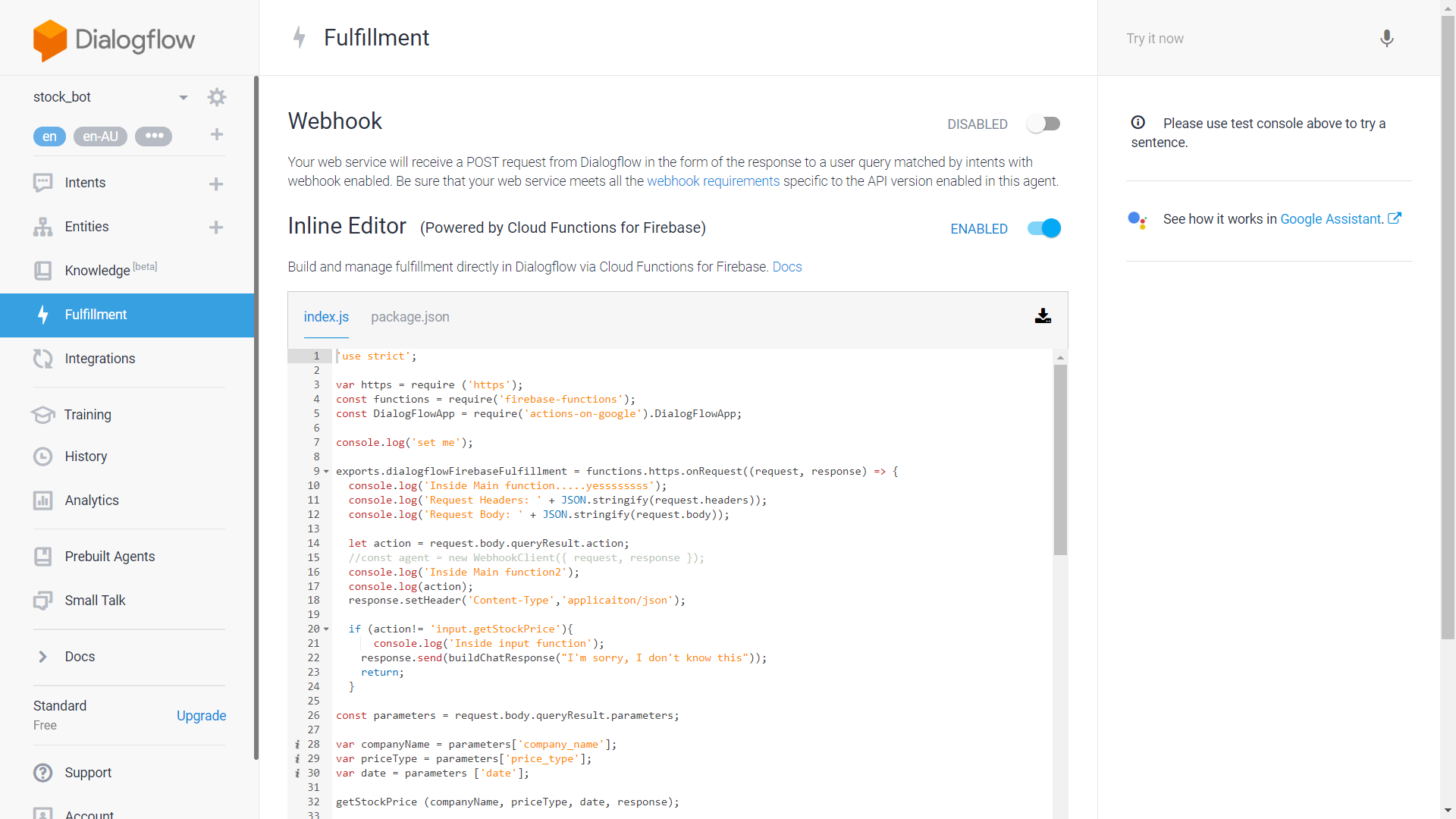
1. **var** https = require ('https');
2. **const** functions = require('firebase-functions');
3. **const** DialogFlowApp = require('actions-on-google').DialogFlowApp;
5. console.log('set me');
7. exports.dialogflowFirebaseFulfillment = functions.https.onRequest((request, response) => {
8. console.log('Inside Main function..');
9. console.log('Request Headers: ' + JSON.stringify(request.headers));
10. console.log('Request Body: ' + JSON.stringify(request.body));
12. let action = request.body.queryResult.action;
13. //const agent = new WebhookClient({ request, response });
14. console.log('Inside Main function2');
15. console.log(action);
16. response.setHeader('Content-Type','applicaiton/json');
18. **if** (action!= 'input.getStockPrice'){
19. console.log('Inside input function');
20. response.send(buildChatResponse("I'm sorry, I don't know this"));
21. **return**;
22. }
24. **const** parameters = request.body.queryResult.parameters;
26. **var** companyName = parameters['company\_name'];
27. **var** priceType = parameters['price\_type'];
28. **var** date = parameters ['date'];
30. getStockPrice (companyName, priceType, date, response);
32. });
34. **function** getStockPrice (companyName, priceType, date, CloudFnResponse) {
36. console.log('In Function Get Stock Price');
38. console.log("company name: " + companyName);
39. console.log("price type: " + priceType);
40. console.log("Date: " + date);

43. **var** tickerMap = {
44. "apple" : "AAPL",
45. "microsoft" : "MSFT",
46. "ibm" : "IBM",
47. "google" : "GOOG",
48. "facebook" : "FB",
49. "snapchat" : "SNAP",
50. "ford" : "F",
51. "yandex" : "YNDX"
52. };
53. **var** priceMap = {
54. "opening" : "open\_price",
55. "closing" : "close\_price",
56. "maximum" : "high\_price",
57. "high" : "high\_price",
58. "low" : "low\_price",
59. "minimum" : "low\_price"
60. };
62. **var** stockTicker = tickerMap[companyName.toLowerCase()];
63. **var** priceTypeCode = priceMap[priceType.toLowerCase()];
64. console.log ('pricetypecode' + priceTypeCode);
66. **var** pathString = "/historical\_data?ticker=" + stockTicker + "&item=" + priceTypeCode +
67. "&start\_date=" + date +
68. "&end\_date=" + date;
70. console.log ('path string:' + pathString);
72. **var** username = "d15b59ddbb42b5de650dd1490f7bece2";
73. **var** password = "df3cb8f5888135efd57fac2febbaa0bd";
75. **var** auth = "Basic " + **new** Buffer(username +":" + password).toString('base64');
77. **var** request = https.get({
78. host: "api.intrinio.com",
79. path: pathString,
80. headers: {
81. "Authorization": auth
82. }
83. }, **function** (response) {
84. **var** json = "";
85. response.on('data', **function**(chunk) {
86. console.log("received JSON response: " + chunk);
87. json += chunk;

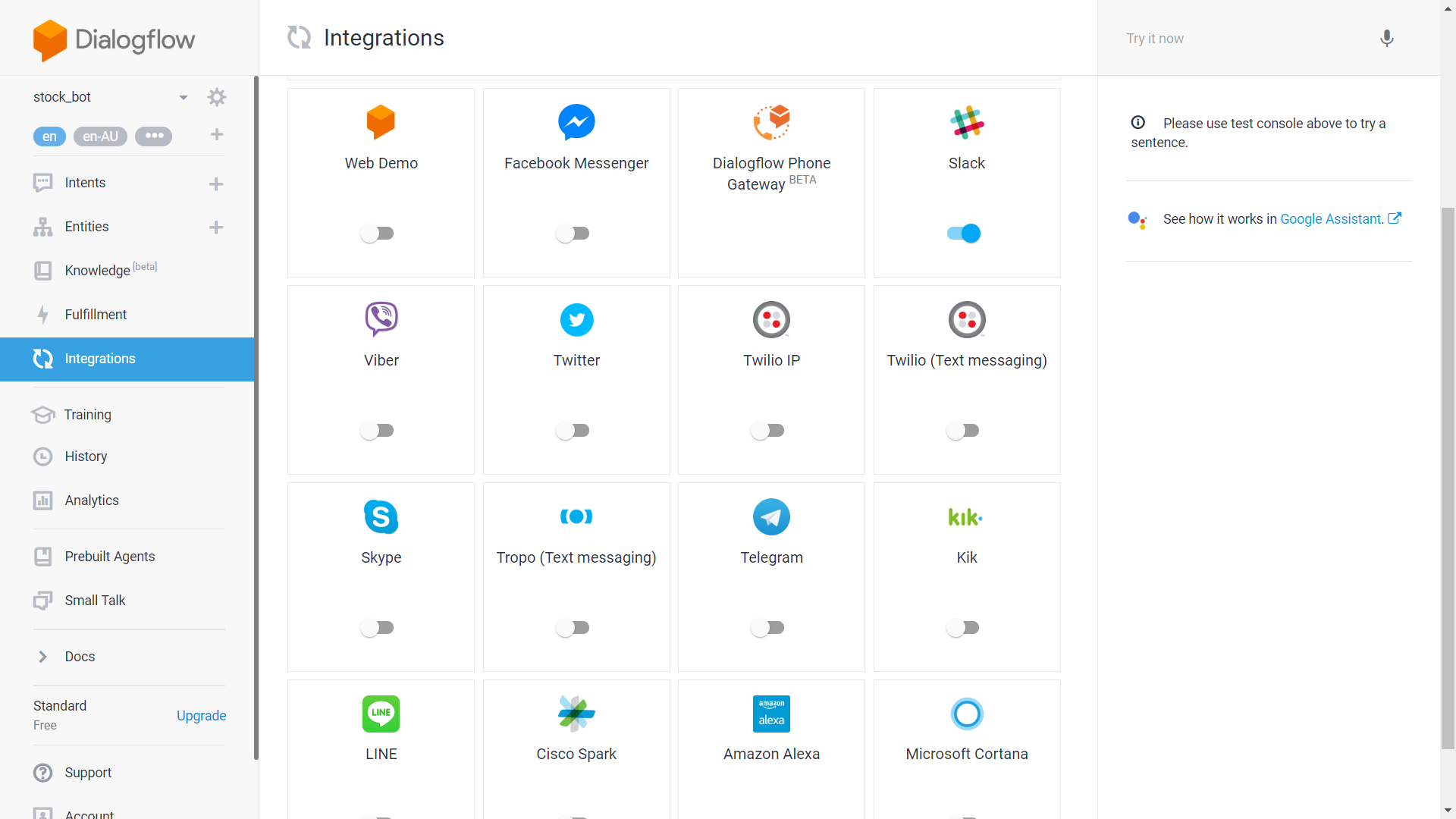
90. });
92. response.on('end', **function**(){
93. **var** jsonData = JSON.parse(json);
94. **var** stockPrice = jsonData.data[0].value;
96. console.log ("the stock price received is:" + stockPrice);
98. **var** chat = "The " + priceType + " price for " + companyName +
99. " on "  + date + " was " + stockPrice;
101. CloudFnResponse.send(buildChatResponse(chat));
103. });
105. });
107. }
109. **function** buildChatResponse(chat) {
110. **return** JSON.stringify({"fulfillmentText": chat});
111. }

Здесь считываются те самые сущности и передаются в Intrinio, которое возвращает значение по принятым параметрам.

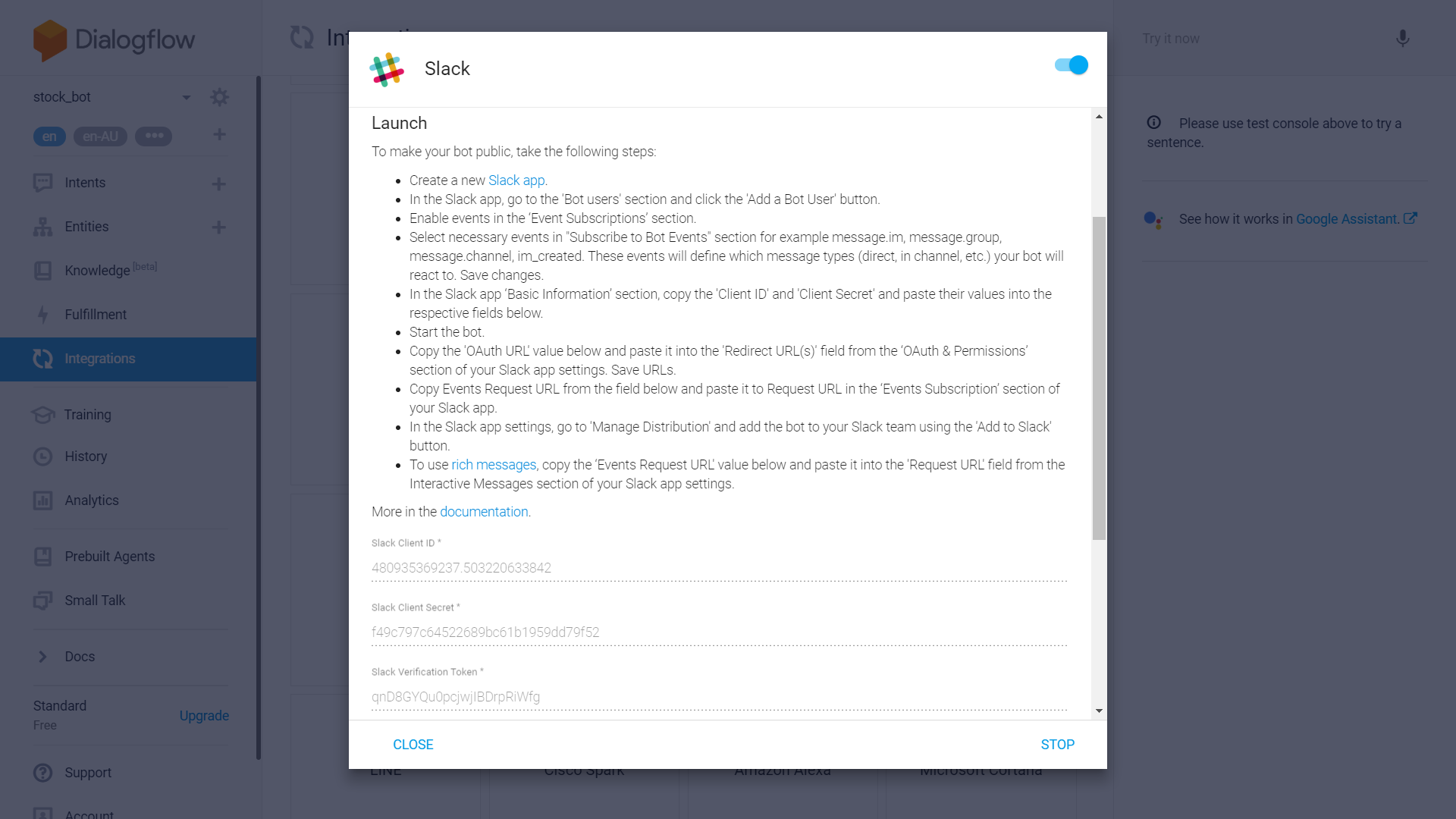
Как это выглядит в консоли DialogFlow:



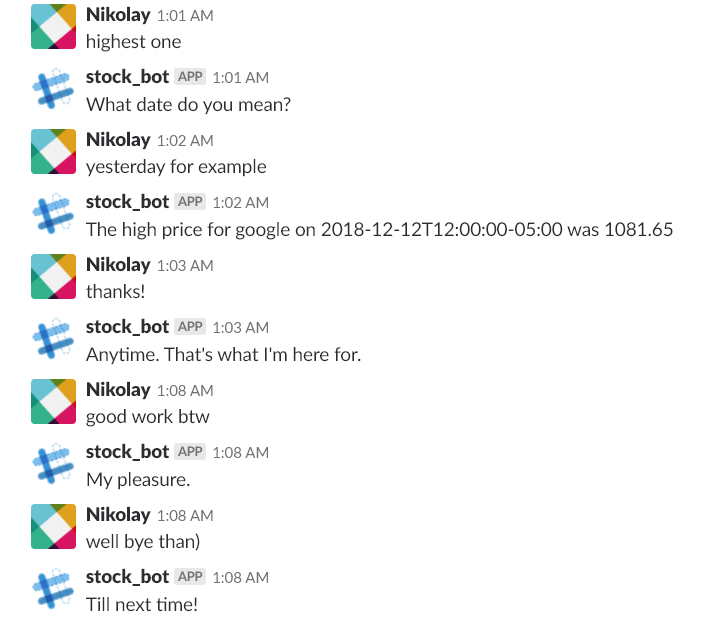
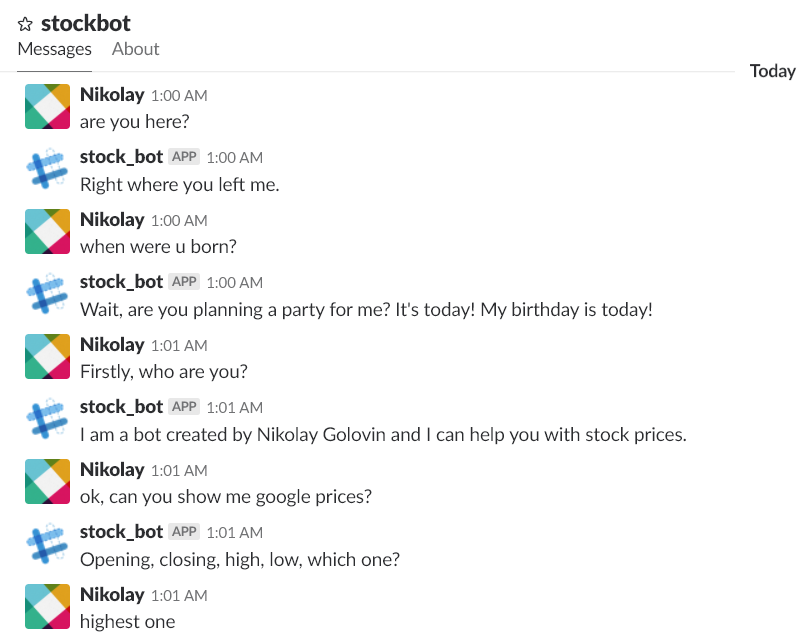
Итак, это все работает в консоли DialogFlow, дальше нужно подключить Slack. Здесь это делается весьма просто во вкладке Integrations, где можно подключить не только Slack, но и много еще популярных приложений.



О том как подключить здесь написано достаточно подробно, нет смысла повторяться:



Мы выполнили все настройки, пока проверить что наш бот умеет:



Из этого диалога можно сделать несколько выводов. Первый – это то что бот отлично понимает контекст написанного и исходя из этого выбирает «правильные» заранее запрограммированные намерения, в результате диалог получается органичным. Второе – это то, что его не «сбить» неправильным написанием слов или построением предложений. Надо отметить, что качество ответов и распознавания фраз увеличивается с большим количеством примеров для тренировки и намерений, то есть бот при должной разработке сможет понимать гораздо более сложные предложения и давать более сложные ответы, не говоря о том, что в будущем можно будет добавить неограниченное количество различных функций, направленных на упрощение мира вокруг нас.