

BOTS Charlatanes

Emiliano Dalla Verde Marcozzi

Twitter @edvm

Email edvm@fedoraproject.org



Que vamos a ver

- Tutorial breve de **Wit.ai**
- **Conversation** de IBM Watson
- Ponerlo en producción con Django

Que pasa si logramos que el computador pueda:

- Entender
- Analizar
- Obtener significado

Por ejemplo, del **TEXTO** que generamos?

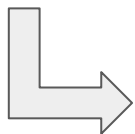
Las interfaces con texto están de moda

- Comentarios en redes sociales, bots, etc
- Whats up
- Twitter
 - Incluso se puede pedir soporte a empresas ←
- IRC, los clásicos no pasan de moda ;-)

Poder **ENTENDER** al
USUARIO

NLP / Natural Language Processing

Es el campo de estudio que se enfoca en la interacción entre el lenguaje humano y los computadores.



NLU / Natural Language Understanding

Es un subtópico de NLP que trata de solucionar cosas aún más complicadas

Algunas aplicaciones

- Crear resúmenes de textos
- Entender sentimientos, clientes enojados, alegres, etc
- Categorizar texto automáticamente (por ej: congreso de la Nación)
- Traducir entre idiomas
- Obtener la **intención** de un usuario, ej: *quiero una pizza*
- Clasificar las preguntas que envía el Congreso Nacional al Poder Ejecutivo

Tutorial breve y acelerado de **Wit.ai**

<https://wit.ai/docs>

Lean la documentación y en
10 minutos seguimos...

Crear nueva app en Wit.ai

wit.ai



Apps

Docs

Explore

Help



Create a new App

edvm / beerstore

Beer store test app

Language

Spanish

BETA

Some entities may not work as expected. [Help us extend it!](#)



Open

Your data will be open to the community



Private

Your data will be private and accessible only by you and the developers you decide to share your app with.


Import your app from a backup

Browse...

+ Create App

Crear nueva app en Wit.ai

wit.ai

+ Apps Docs Explore Help 

 edvm / beerapp / 


 Inbox

 Understanding

 Stories ^{beta}

 Actions

 Logs

 Settings

BEERAPP

+ Create a story



Create your bot's first story

It's time to build beerapp! To start, we need to create stories. A story is an example of a conversation that people will have with your bot.

Click  to start your first story.

5 Minute Quickstart

Recipes and Guides

API Documentation

Feature Roadmap

Export App

Press  to chat with your bot



Crear primer story en Wit.ai

wit.ai

+ Apps Docs Explore Help 

edvm / beerapp / ●

Inbox

Understanding

 Stories ^{beta}

⚡ Actions

📁 Logs

⚙️ Settings

BEERAPP

All Stories

Hola

+ Create a story

Hola

Cancel

Save Story



Hola

What intent does this story handle? e.g greeting, question...



intent

+ Add a new entity

saludo

Create option "saludo"



User says



Bot sends



Bot executes



Jump



Bot sends

If your bot is ready to send a direct response (like "Hello!") to the user, click on **Bot Says**.


5 Minute Quickstart

Recipes and Guides

API Documentation

Feature Roadmap

Export App

Press  to chat with your bot



Crear primer story en Wit.ai

wit.ai

+ Apps Docs Explore Help 

edvm / beerapp / ●

Inbox

Understanding

 ^{beta} Stories

⚡ Actions

📁 Logs

⚙️ Settings

BEERAPP

All Stories

Hola

+ Create a story

Hola

Cancel

Save Story



Hola

intent

saludo

x

+ Add a new entity

x

Hola! Bienvenido a PyConAr Beer Bar 2016! :)



+ Variable

Set quick replies



User says



Bot sends



Bot executes



Jump



Bot Says

Type the bot's response text. You can click on **Add Variable**, if you would like the bot to use dynamic data.

5 Minute Quickstart

Recipes and Guides

API Documentation

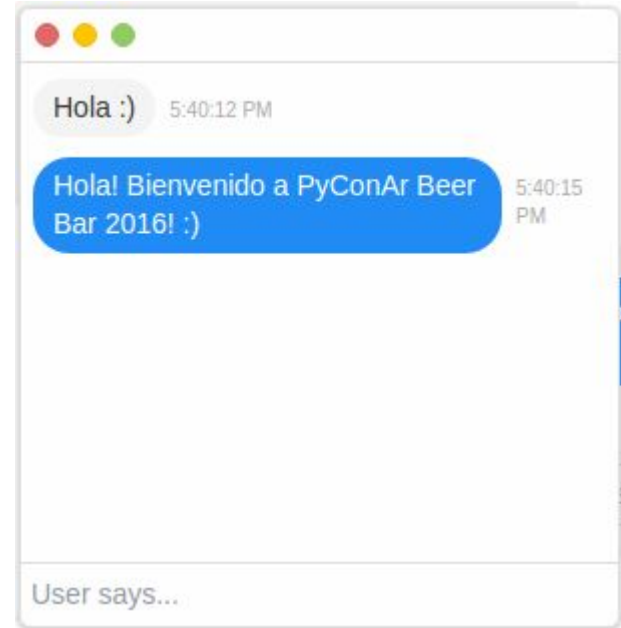
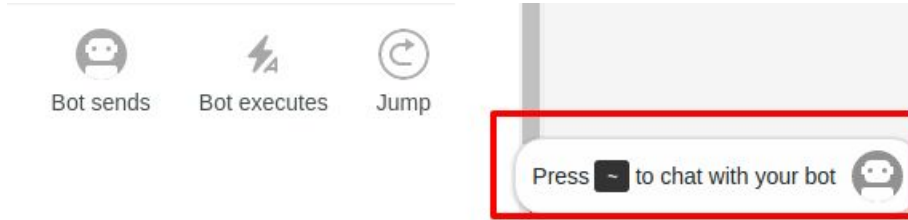
Feature Roadmap

Export App

Press  to chat with your bot



Crear primer story en Wit.ai



Agregando mas formas de saludar

wit.ai


+ Apps Docs Explore Help 

 edvm / beerapp / 

 Inbox

 Understanding

 Stories ^{beta}

 Actions

 Logs

 Settings

Test how your bot understands a sentence

You can train your bot by adding more examples

Hey, como andas?

intent

saludo

+ Add a new entity

✓ Validate

Your app uses 1 entity

Name

Search Strategy 

Values

intent →

User-defined entity

trait

free-text

keywords

saludo

Press  to chat with your bot



Probando desde Python

`pip install wit`

Ojo, es **'wit'**, no **'pywit'**

Probando desde Python

```
from wit import Wit

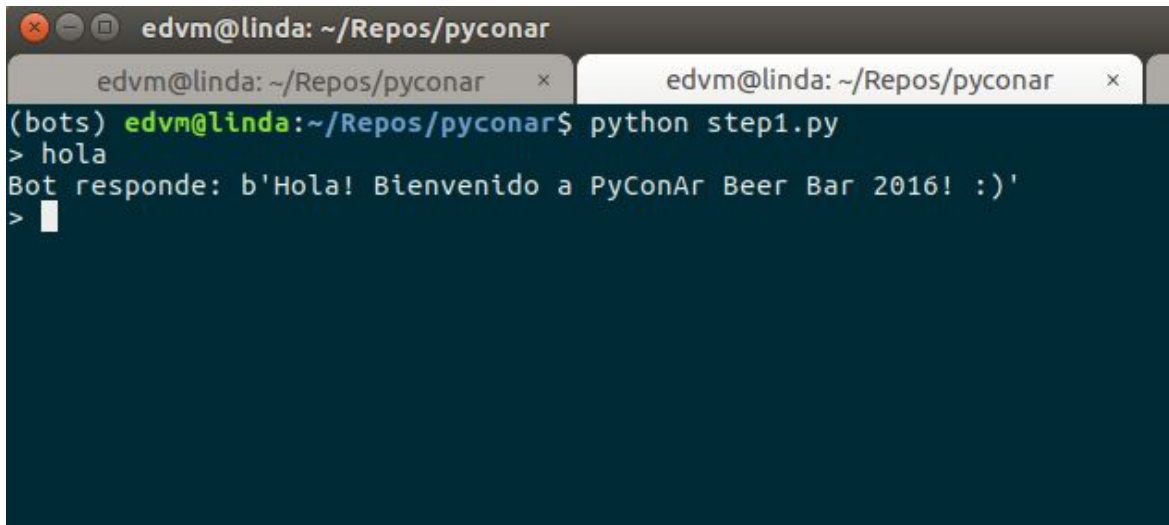
def send(request, response):
    print('Bot responde: {}'.format(response['text']))

actions = {
    'send': send,
}

ACCESS_TOKEN = 'too_secret'

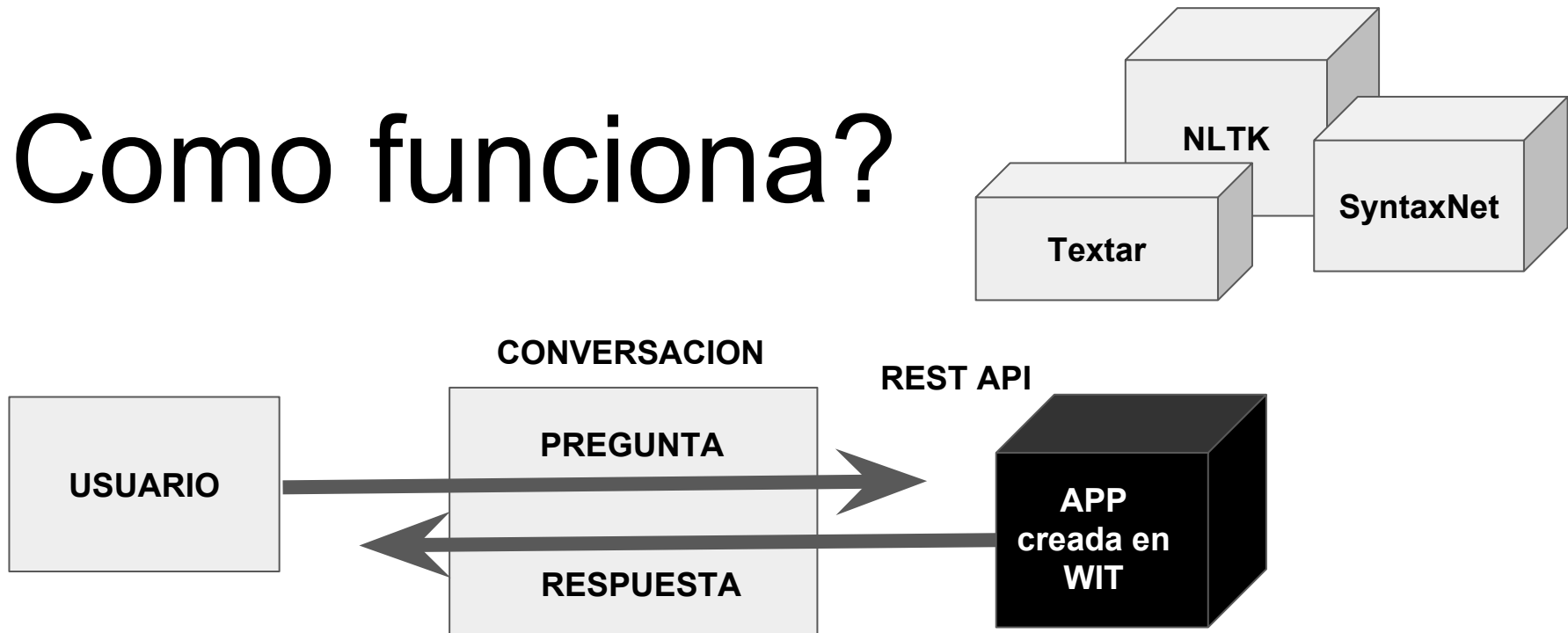
client = Wit(
    access_token=ACCESS_TOKEN,
    actions=actions
)

client.interactive()
```

A terminal window with a dark background and light-colored text. The window title is 'edvm@linda: ~/Repos/pyconar'. The prompt is '(bots) edvm@linda:~/Repos/pyconar\$'. The user enters 'python step1.py'. The prompt changes to '>'. The user enters 'hola'. The prompt changes to 'Bot responde:'. The output is 'b\'Hola! Bienvenido a PyConAr Beer Bar 2016! :)\''. The prompt changes to '>' and a cursor is visible.

```
(bots) edvm@linda:~/Repos/pyconar$ python step1.py
> hola
Bot responde: b'Hola! Bienvenido a PyConAr Beer Bar 2016! :)'
> 
```

Como funciona?



Agregando una segunda intención

wit.ai

+ Apps Docs Explore Help 

edvm / beerapp / ●

Inbox

Understanding

 ^{beta} Stories

⚡ Actions

📁 Logs

⚙️ Settings

BEERAPP

All Stories

Hola ●

Quiero una cerveza ●

+ Create a story

Quiero una cerveza

Cancel

Save Story

×



Quiero una cerveza

×

intent

+ Add a new entity

What intent does this story handle? e.g greeting, question...

beber|

Create option "beber"



User says



Bot sends



Bot executes



Jump



Bot sends

If your bot is ready to send a direct response (like "Hello!") to the user, click on **Bot Says**.

5 Minute Quickstart

Recipes and Guides

API Documentation

Feature Roadmap

Export App

Press  to chat with your bot



Agregando una segunda intención

wit.ai

+ Apps Docs Explore Help 

edvm / beerapp / ●

Inbox

Understanding

Stories ^{beta}

Actions

Logs

Settings

BEERAPP

All Stories

Hola ●

Quiero una cerveza ●

+ Create a story

+ Add a new entity

Rubia

Negra

Roja

Perfecto! De que tipo?

+ Variable

Rubia

Negra

Roja



Rubia



tipo_birra

Rubia



+ Add a new entity



Bot sends

If your bot is ready to send a direct response (like "Hello!") to the user, click on **Bot Says**.

5 Minute Quickstart

Recipes and Guides

API Documentation

Feature Roadmap

Export App



User says



Bot sends



Bot executes




Jump

Press  to chat with your bot



Agregando una segunda intención

wit.ai

+ Apps Docs Explore Help 

edvm / beerapp / ●

Inbox

Understanding

Stories ^{beta}

Actions

Logs

Settings

BEERAPP

All Stories

Hola

Quiero una cerveza

+ Create a story

Rubia

Negra

Roja



Roja



tipo_birra

Roja



+ Add a new entity

```
{
  tipo_birra,
}
```

get_beer_type(context, entities)



tipo_birra

Perfecto! Marche una cerveza
{tipo_birra} bien helada!



+ Variable

Set quick replies



Bot executes

If your bot needs to perform an action on the context (for instance to store the extracted entities, or do a calculation, or call an external API, etc.), click **Bot executes**.

5 Minute Quickstart

Recipes and Guides

API Documentation

Feature Roadmap

Export App

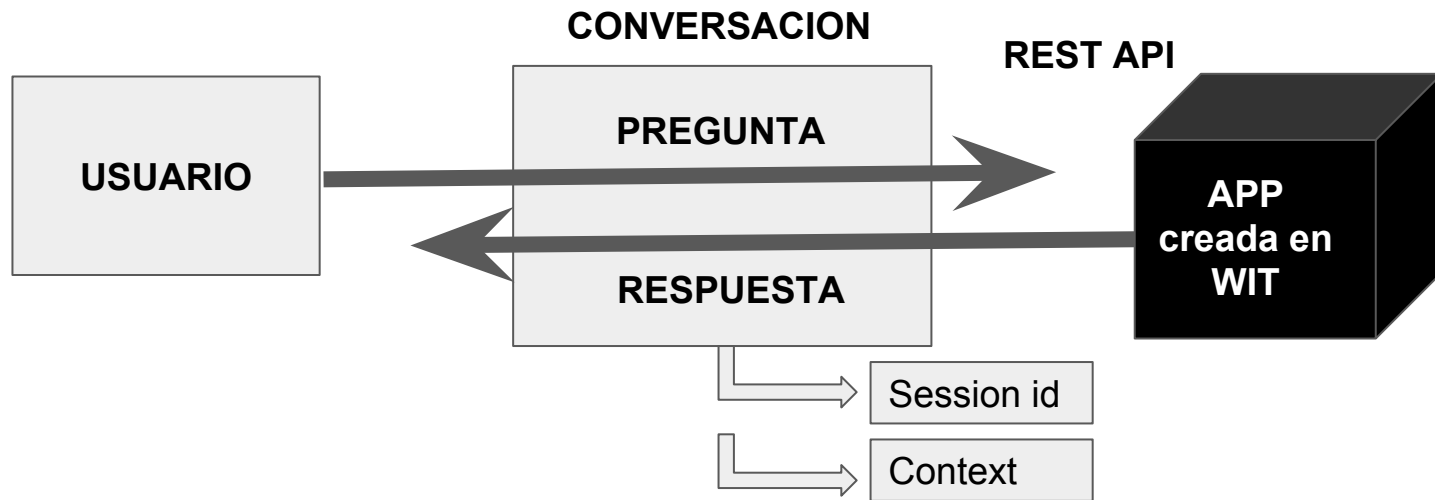
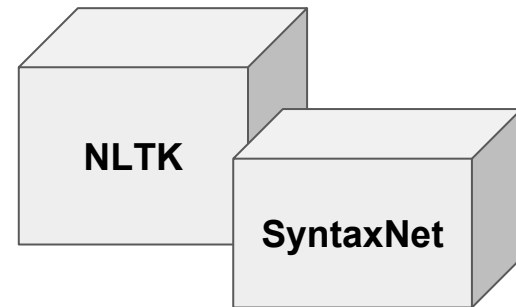
Press  to chat with your bot



```
2 from utils import first_entity_value
3
4 def send(request, response):
5     print(response['text'])
6     if response['quickreplies']:
7         print(">>> Opciones: {}".format(response['quickreplies']))
8
9 def get_beer_type(request):
10     context = request['context']
11     entities = request['entities']
12
13     beer_type = first_entity_value(entities, 'tipo_birra')
14     if beer_type:
15         context['tipo_birra'] = beer_type
16
17     return context
18
19 actions = {
20     'send': send,
21     'get_beer_type': get_beer_type,
22 }
23
24 ACCESS_TOKEN = 'BYP2LLV64PDAUH2KPNRHN2T7ICZUGDHV'
25 client = Wit(access_token=ACCESS_TOKEN, actions=actions)
26 client.interactive()
```

```
32 (bots) edvm@linda:~/Repos/pyconar$ python
31 2.py
30 > hola
29 b'Hola! Bienvenido a PyConAr Beer Bar 2016
28 '
27 > quiero una cerveza
26 b'Perfecto! De que tipo?'
25 >>> Opciones: ['Rubia', 'Negra', 'Roja']
24 > Rubia
23 b'Perfecto! Marche una cerveza Rubia bien
22 da!'
21 > 
20
19
18
17
16
15
14
13
12
11
10
9
8
```

Como funciona?



Conversation de IBM Watson

(en un solo slide)

Chrome File Edit View History Bookmarks People Window Help

IBM Watson Conversation x Editing Designing IBM Wat...

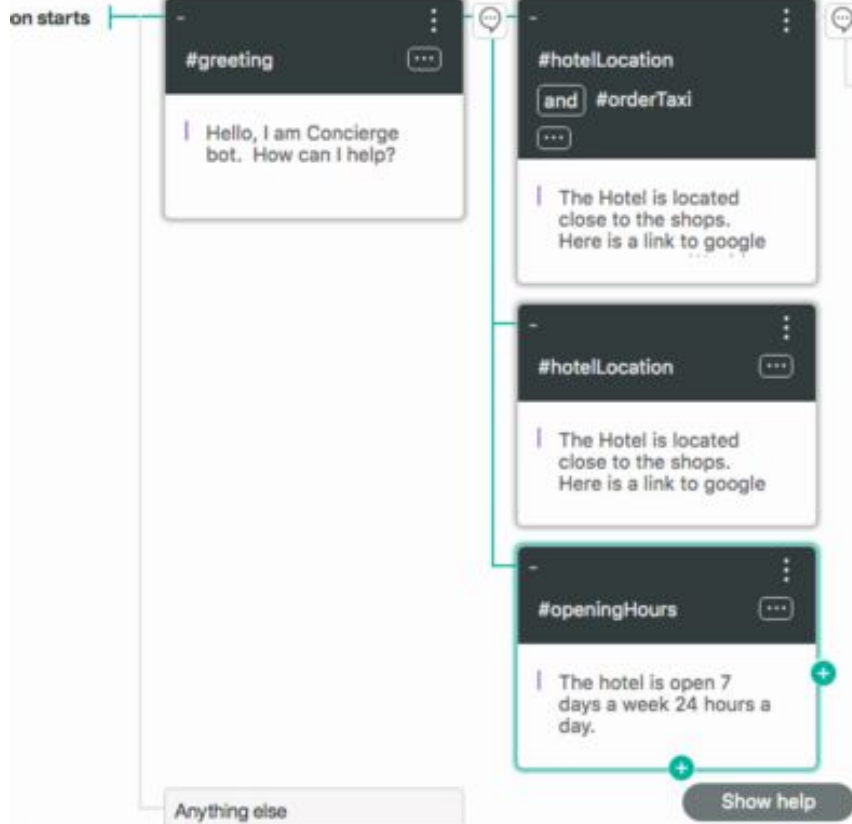
John

https://www.ibmwatsonconversation.com/us-south/49888290-9a3f-4fbf-93e5-ff4c4cde0045/workspaces/a6b3417f-8247-41b7-b583-f2a6ad688179/dia...

Apps Stuff Blogs

Other Bookmarks

Hotel Concierge Intents Entities Dialog



Try it out

Clear

Watson is done training

#greeting

Hello, I am Concierge bot. How can I help?

where is the hotel?

#hotelLocation

The Hotel is located close to the shops. Here is a link to google maps - xxxxx

can I book a taxi

#orderTaxi

I am sorry, I don't understand.

can I order a taxi from the hotel?

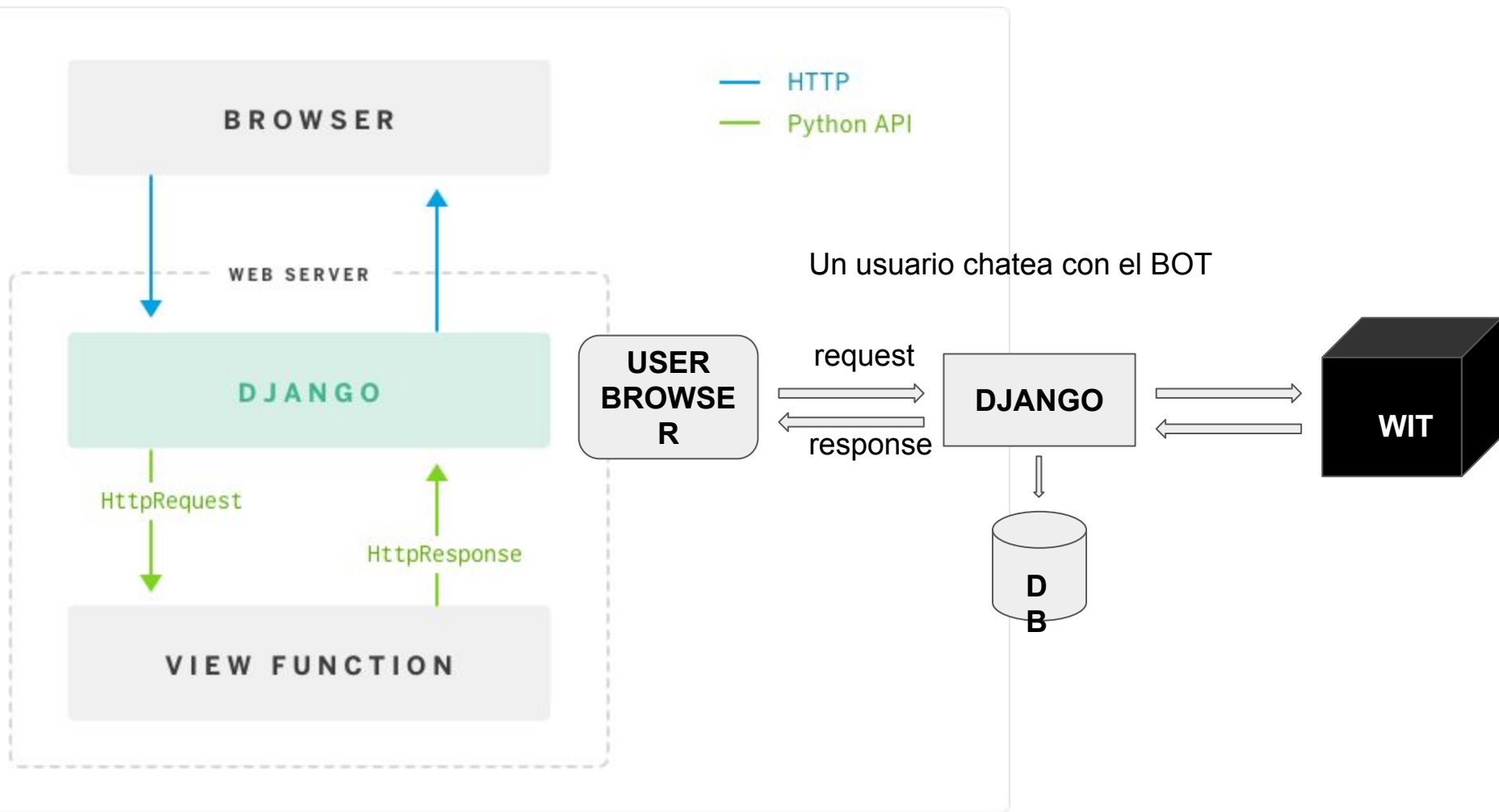
#orderTaxi

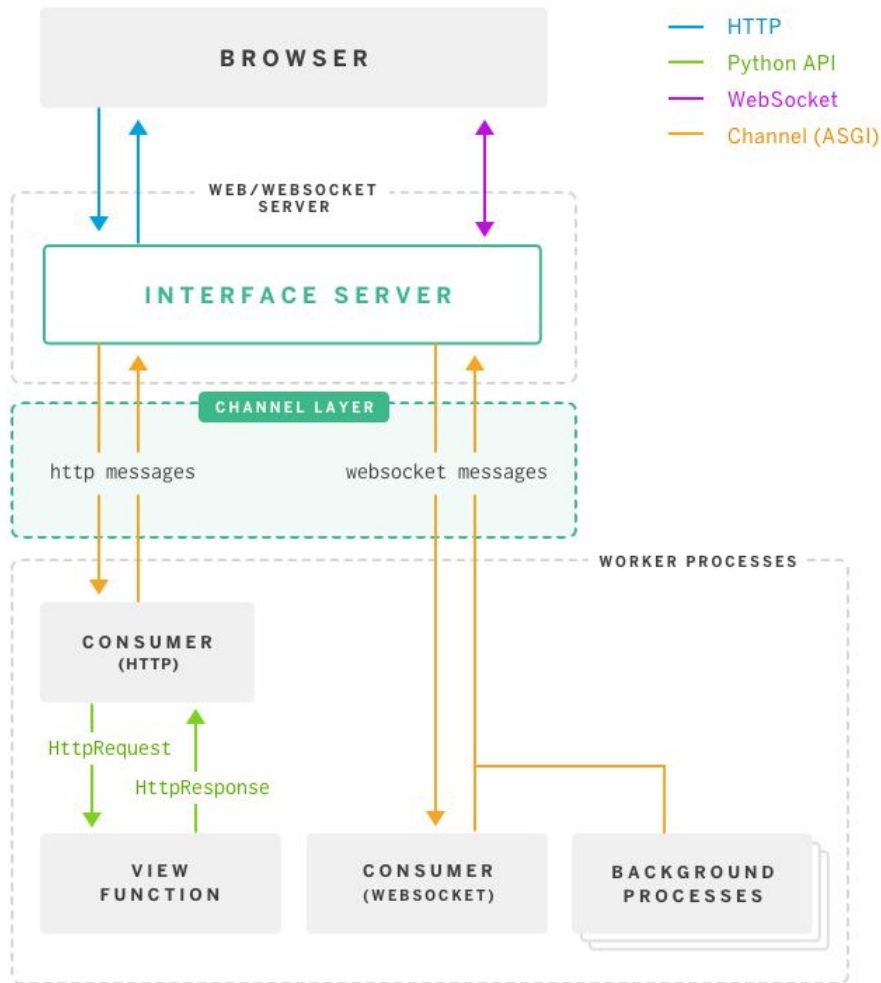
I am sorry, I don't understand.

Enter something to test your service

Poniendo esto en Producción

Con Django Channels

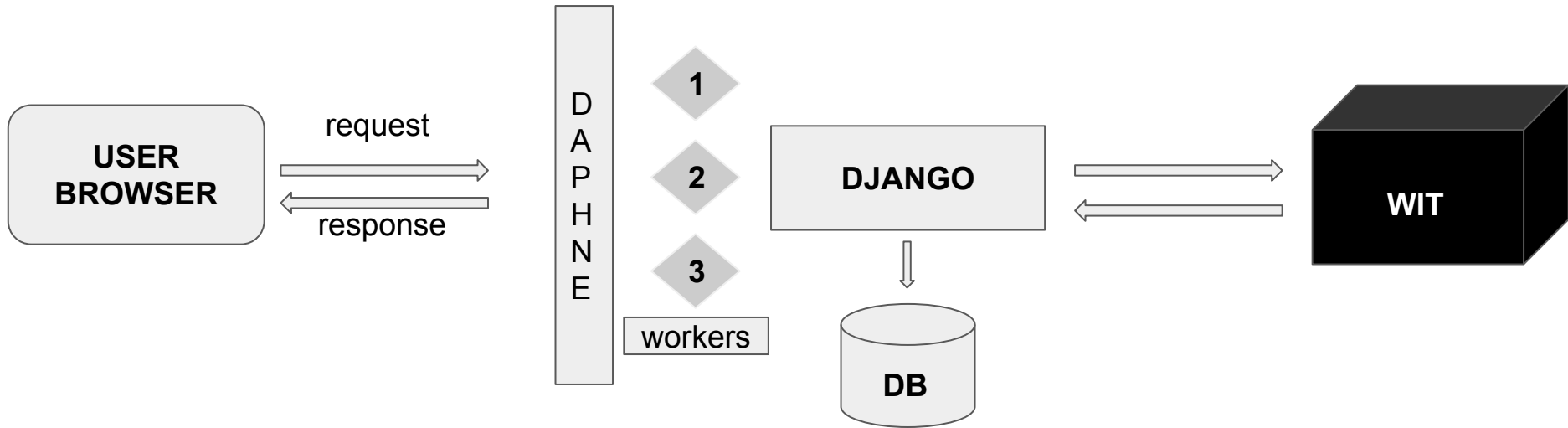




Los channels son **Colas de Tareas**, donde los **Productores pushean mensajes** que luego son entregados a los **Consumidores** que están escuchando en esos channels. [Ver mas](#)

PASOS

- pip install **channels**
- Agregar '**channels**' a **INSTALLED_APPS**
- Elegir un **channel layer** y configurarlo en **settings.py** variable **CHANNEL_LAYERS**
- Configurar **channel_routing**
- Configurar **asgi.py** (Asynchronous Server Gateway Interface)
- Correr Django con **Daphne** (Channel Interface Server)

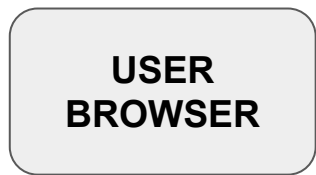


routing.py

```
Terminal File Edit View Search Terminal Help
1 from channels.routing import route
2
3 channel_routing = [
4     route('websocket.receive', 'pyconar.consumers.ws_bot'),
5 ]
6
```

consumers.py

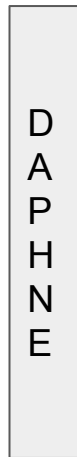
```
Terminal File Edit View Search Terminal Help
1 from pyconar.bot import ConversationBot
2 from channels.sessions import channel_session
3
4
5 @channel_session
6 def ws_bot(message):
7
8     bot = ConversationBot()
9     bot.ask(question=message)
10
11     message.channel_session['context'] = bot.context
12     message.reply_channel.send({'text': bot.response.as_json})
```



request



response

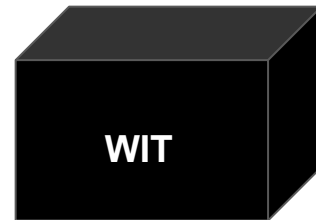
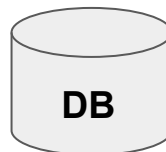


1

2

3

workers



FIN

Emiliano Dalla Verde Marcozzi
@edvm
edvm@fedoraproject.org