

# ÌNDICE

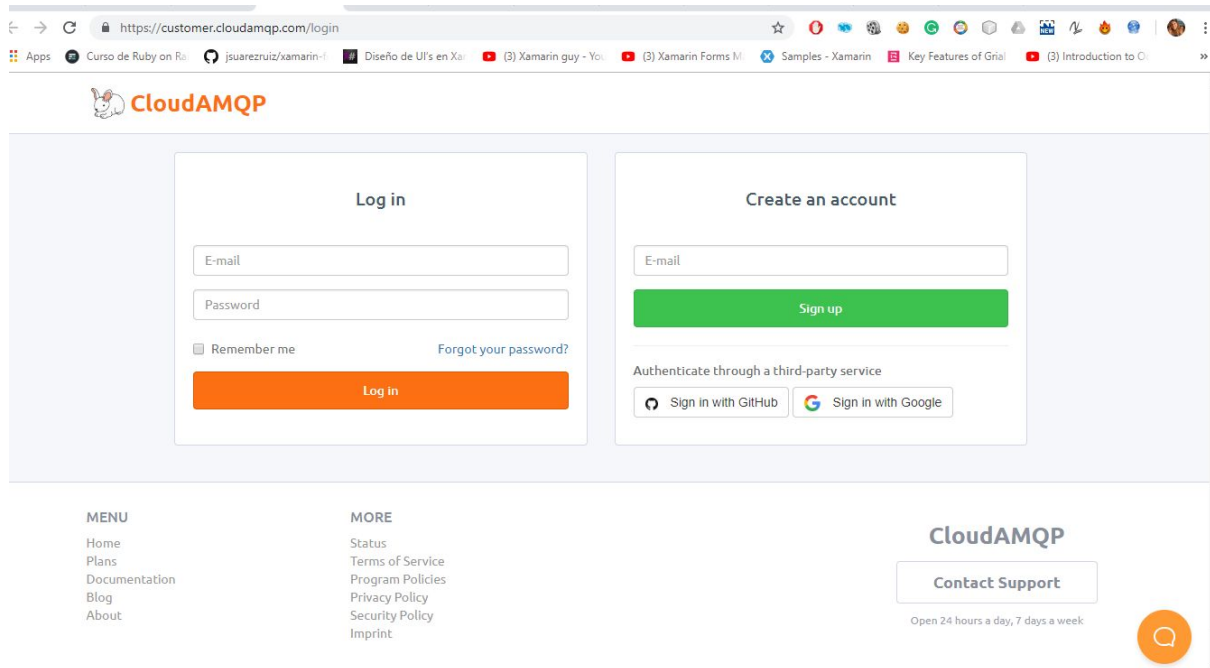
<b>ÌNDICE</b>	<b>0</b>
<b>Creaciòn de la QUEUE EN CloudAMQP</b>	<b>1</b>
<b>Proyecto Publisher</b>	<b>6</b>
Cambios en el còdigo	7
CONCLUSION	11
<b>Proyecto Subscribe</b>	<b>11</b>
<b>Tutoriales que serguì:</b>	<b>12</b>

# Creación de la QUEUE EN CloudAMQP

Lo que se va a usar es: RabbitMQ

Ir a <https://customer.cloudamqp.com/instance/create>

Nos creamos una cuenta, yo utilizo Sign in with Google



The screenshot shows the CloudAMQP login and account creation interface. The page has a light blue header with the CloudAMQP logo. Below the header, there are two main sections: 'Log in' and 'Create an account'. The 'Log in' section contains fields for 'E-mail' and 'Password', a 'Remember me' checkbox, a 'Forgot your password?' link, and an orange 'Log in' button. The 'Create an account' section contains an 'E-mail' field, a green 'Sign up' button, and a section for 'Authenticate through a third-party service' with buttons for 'Sign in with GitHub' and 'Sign in with Google'. At the bottom, there is a 'MENU' section with links to Home, Plans, Documentation, Blog, and About. A 'MORE' section lists links to Status, Terms of Service, Program Policies, Privacy Policy, Security Policy, and Imprint. On the right, there is a 'CloudAMQP' logo, a 'Contact Support' button, and a note 'Open 24 hours a day, 7 days a week'. A small orange chat bubble icon is visible in the bottom right corner.

Luego se especifica un name y un Tags:

PlanRegionConfigure (Dedicated plans only)Confirm

Select a plan and name - Step 1 of 4

Name

PersonasIntance

Plan

Little Lemur (Free)


Tags

Personas

Tags are used to separate your instances between projects. This is primarily used in the project listing view for easier navigation and access control.

Tags allow admins to [manage team members](#) access to different groups of instances.

Plan



Little Lemur

See the [plan page](#) to learn about the different plans.

Cancel

Select Region

Data Center: Virginia


Missing company address Please [fill in your address](#) if you want to subscribe to a paid plan

PlanRegionConfigure (Dedicated plans only)Confirm

Select a data center and type - Step 2 of 4

Data center

US-East-1 (Northern Virginia)




« Back

Cancel

Confirm

### Confirm new instance - Step 4 of 4

**Plan**



Little Lemur

Total: \$0/month

Name: PersonasIntance


Provider: Amazon Web Services

Region: US-East-1 (Northern Virginia)

Backend: RabbitMQ

Tags: Personas

Quedò creada la instancia

 List all instances ▾ vsseoane@gmail.com ▾

**Instances**

✓ Instance successfully created ✕

Name	Host	Plan	Datacenter	Actions
app	toad	Lemur	Amazon Web Services US-East-1 (Northern Virginia)	<a href="#">Edit</a> <a href="#">RabbitMQ Manager ↗</a>

**Personas**

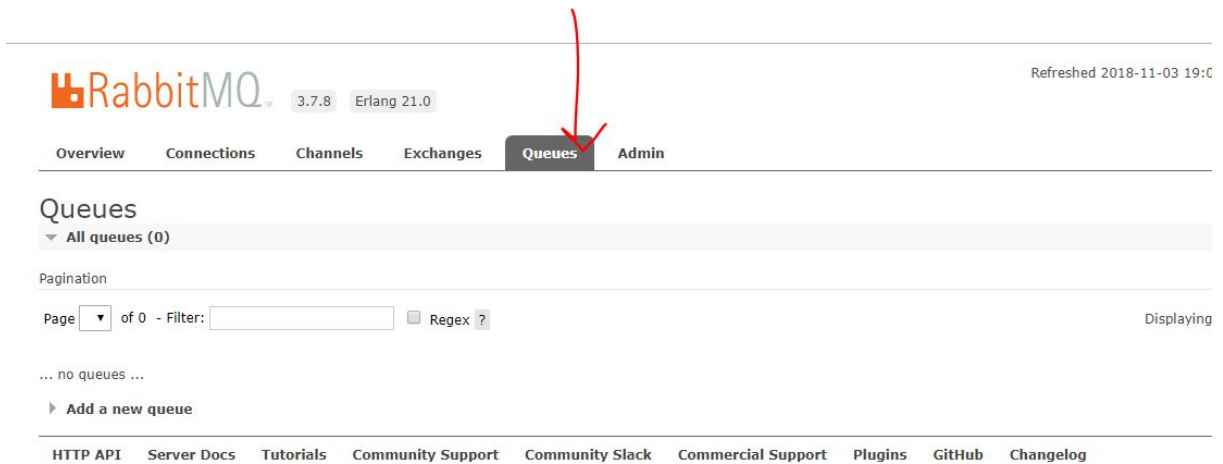
Name	Host	Plan	Datacenter	Actions
<a href="#">PersonasIntance</a>	toad	Lemur	Amazon Web Services US-East-1 (Northern Virginia)	<a href="#">Edit</a> <a href="#">RabbitMQ Manager ↗</a>

Hacemos click en :

**Personas**

Name	Host	Plan	Datacenter	Actions
<a href="#">PersonasIntance</a>	toad	Lemur	Amazon Web Services US-East-1 (Northern Virginia)	<a href="#">Edit</a> <a href="#">RabbitMQ Manager ↗</a>

Ir al tab Queues



Seleccionamos Add a new queue

The image shows the 'Add a new queue' form in the RabbitMQ Admin UI. The form is titled 'Add a new queue' and has a dropdown menu showing 'All queues (0)'. Below the title, there is a 'Pagination' section with 'Page 1 of 0 - Filter: [input] [checkbox] Regex ?' and 'Displaying'. Below this, it says '... no queues ...'. The form itself has a 'Name' field with a red asterisk indicating it is required. Below the name field, there is a 'Durability' dropdown menu with 'Durable' selected. Below the durability dropdown, there is an 'Auto delete' dropdown menu with 'No' selected. Below the auto delete dropdown, there is an 'Arguments' field with an equals sign and a dropdown menu with 'String' selected. Below the arguments field, there is a list of options: 'Add Message TTL ? | Auto expire ? | Max length ? | Max length bytes ? | Overflow behaviour ? | Dead letter exchange ? | Dead letter routing key ? | Maximum priority ? | Lazy mode ? | Master locator ?'. At the bottom of the form, there is an 'Add queue' button.

▼ Add a new queue

Name:  \*

Durability:

Auto delete:

Arguments:  =

Add  ? |  ? |  ? |  ? |  ?

? |  ? |  ?

? |  ?

Si hacemos click en el nombre de la Queue muestra unas mètricas

## Queues

▼ All queues (1)

Pagination

Page  of 1 - Filter:  ☐ Regex ?

Overview			Messages			Message rates			+/-
Name	Features	State	Ready	Unacked	Total	incoming	deliver / get	ack	
personas	<input type="text" value="D"/> <input type="text" value="HA"/>	<input type="text" value="idle"/>	0	0	0				

▼ Add a new queue

Name:  \*

Durability:

Auto delete:

Arguments:  =

Add  ? |  ? |  ? |  ? |  ?

? |  ? |  ?

? |  ?

## Queue personas

▼ Overview

Queued messages  ?

Message rates  ?

Currently idle

Details

Features	State
Policy	HA
Operator policy	HA
Effective policy definition	expirés: 2419200000 ha-mode: all ha-sync-mode: automatic queue-mode: lazy
Consumers	0
Consumer utilisation	0%
Messages	0
Message body bytes	0B
Process memory	14kB
Total	0
Ready	0
Unacked	0
In memory	0
Persistent	0
Transient, Paged Out	0

Vamos a instalar Bunny, es un cliente de ruby rabbitMQ que nos conecta con rabbitMAnager

gem install bunny

<https://github.com/ruby-amqp/bunny>

Ahora hacemos dos proyectos en ruby mine, para conectarlos mediante esta cola:

Por un lado hacemos rails new publisher y rails new publisher

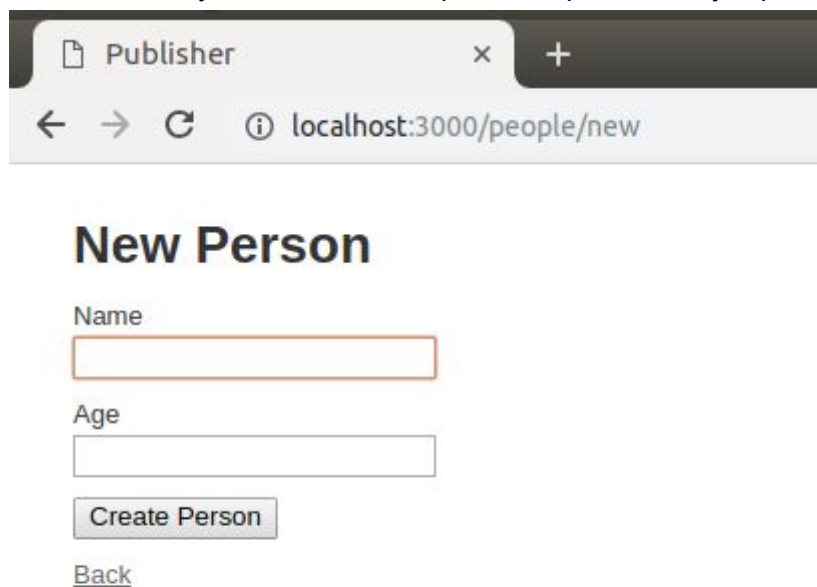
La idea va a ser crear una persona en el proyecto de publisher y pasarsela a subscriber

## Proyecto Publisher

Para eso en el proyecto de publisher creamos el objeto Persona:

rails scaffold g Person name:string age:integer

Entrar a la url y crear un usuario para ver que todo haya quedado bien



The screenshot shows a web browser window with a single tab titled 'Publisher'. The address bar displays 'localhost:3000/people/new'. The page content features a heading 'New Person' in a large, bold font. Below the heading are two text input fields: the first is labeled 'Name' and the second is labeled 'Age'. Underneath these fields is a button labeled 'Create Person' and a link labeled 'Back'.

Ahora se instala la gema

gem install bunny

Para verificar que se haya instalado bien Bunny, en la consola poner lo siguiente:

## Verifying your installation

Verify your installation with a quick irb session:

```
irb -rubygems
:001 > require "bunny"
=> true
:002 > Bunny::VERSION
=> "2.10.0"
```

```
vero@vero-VirtualBox:~/Escritorio/ProbandoRabbit/Publisher$ irb -rubygems
/usr/lib/ruby/2.5.0/irb/init.rb:280: warning: LoadError: cannot load such file -- ubygems
irb(main):001:0> requiere "bunny"
Traceback (most recent call last):
  2: from /usr/bin/irb:11:in `<main>'
  1: from (irb):1
NoMethodError (undefined method `requiere' for main:Object)
Did you mean?  require
irb(main):002:0> require "bunny"
=> true
irb(main):003:0> Bunny::VERSION
=> "2.12.0"
irb(main):004:0>
```

Y a partir de lo que voy a hacer ahora lo saqué de

[http://rubybunny.info/articles/getting\\_started.html](http://rubybunny.info/articles/getting_started.html)

## Cambios en el código

Se modifican 3 archivos:

- person.rb
- publisher\_job.rb
- publisher\_service.rb

En el modelo



```
person.rb
1 class Person < ApplicationRecord
2   after_create :notify_creation
3
4   def notify_creation
5     PublisherJob.perform_later self
6   end
7 end
8
```

```
class Post < ApplicationRecord
  after_create :notify_creation

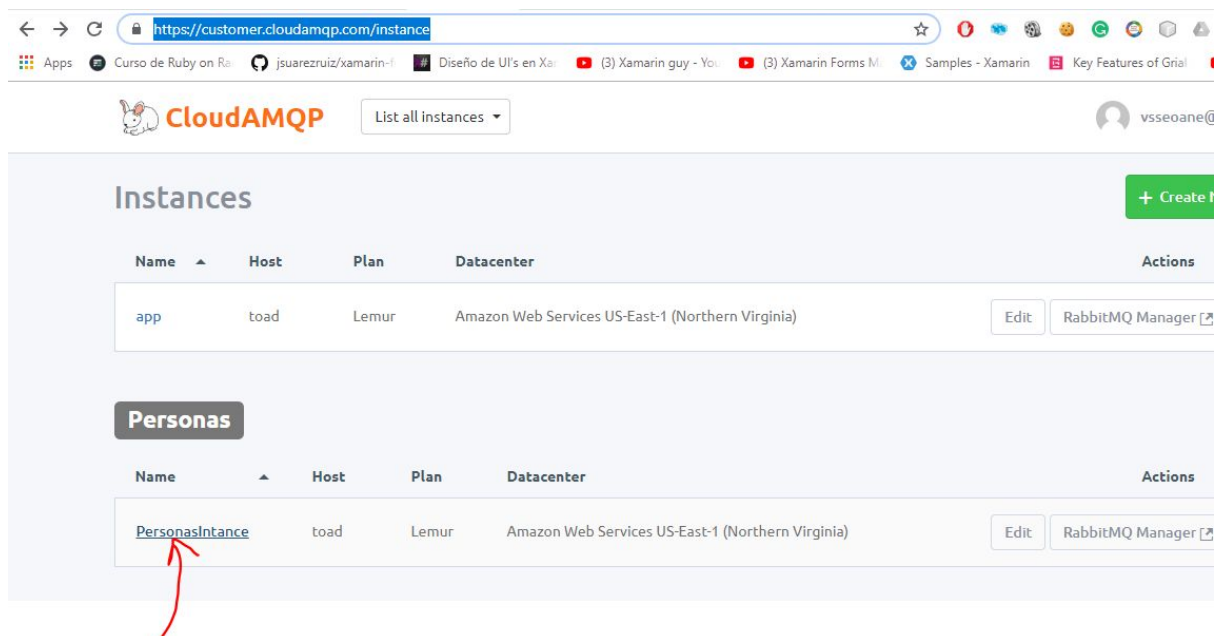
  def notify_creation
    PublisherJob.perform_later self
  end
end
```

Se crea una carpeta dentro de app llamada “services”, luego dentro se crea un archivo llamado “publisher\_service” y allí se pone el siguiente código:



En vez de posts poner el nombre de la queue en nuestro caso “personas”. Además hay que cambiar el usuario y la password por la que diga en los detalles de la cola:


Para verlo:

Click en:



Allí aparecen los datos que se necesitan


PersonasIntance ▾
 vsseoane@gmail.com ▾



Details

Host(s)

toad.rmcloudamqp.com (Load balanced)  
toad-01.rmcloudamqp.com

User & Vhost

uolpaoq

Password

JA03xWlWdw0S4Ds\_QlueJP5\_kQxn6iRo Rotate password

AMQP URL

amqp://uolpaoq:JA03xWlWdw0S4Ds\_QlueJP5\_kQxn6iRo@toad.rmcloudamqp.com/uolpaoq

MQTT details


Open connections

0 of 20

When you've reached the maximum concurrent connections further connections will be prohibited. You can connect again when you're under the limit.

Unfortunately when you've reached the maximum concurrent connections you can't access the management interface either

Active Plan



Little Lemur

Upgrade Instance

```

1  class PublisherService
2
3  def self.publish(message)
4    Rails.logger.warn "estoy en publisher_services y estoy en el publish y el mensaje es "
5    Rails.logger.warn message
6    queue = channel.queue('personas', durable: true)
7    channel.default_exchange.publish(message.to_json, routing_key: queue.name)
8  end
9
10 def self.channel
11   @channel ||= connection.create_channel
12 end
13
14 def self.connection
15   @connection ||= Bunny.new(
16     host: "toad.rmcloudamqp.com",
17     port: 5672,
18     ssl: false,
19     vhost: "uolpaoq",
20     user: "uolpaoq",
21     pass: "JA03xWlWdw0S4Ds_QlueJP5_kQxn6iRo",
22     heartbeat: :server, # will use RabbitMQ setting
23     frame_max: 131072,
24     auth_mechanism: "PLAIN"
25   ).start
26 end
27
28 end

```

```
class PublisherService
```

```

def self.publish(message)
  Rails.logger.warn "estoy en publisher_services y estoy en el publish y el
mensaje es "
  Rails.logger.warn message
  queue = channel.queue('personas', durable: true)
  channel.default_exchange.publish(message.to_json, routing_key: queue.name)

```

9

```

end

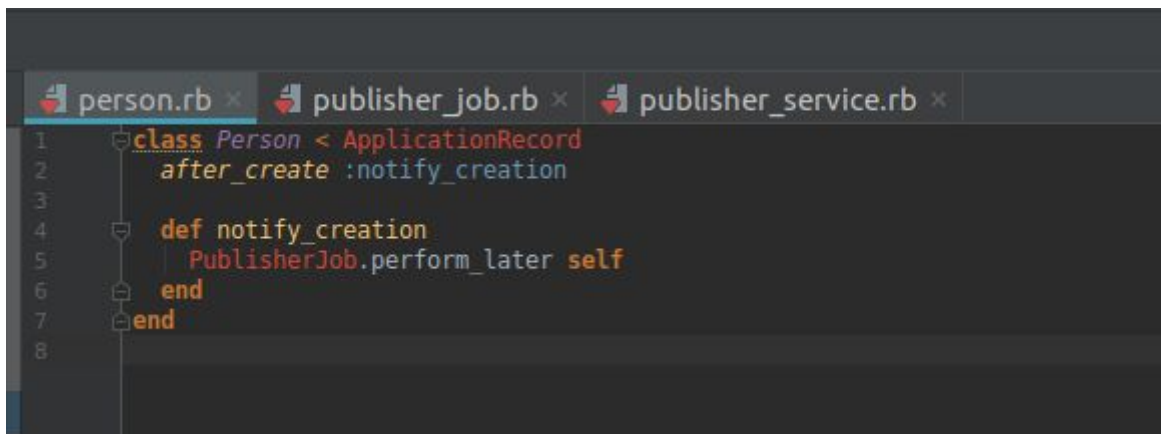
def self.channel
  @channel ||= connection.create_channel
end

def self.connection
  @connection ||= Bunny.new(
    host: "toad.rmq.cloudamqp.com",
    port: 5672,
    ssl: false,
    vhost: "uolpaoq",
    user: "uolpaoq",
    pass: "JA03xWlWdw0S4Ds_QlueJP5_kQxn6iRo",
    heartbeat: :server, # will use RabbitMQ setting
    frame_max: 131072,
    auth_mechanism: "PLAIN"
  ).start
end

end

```

Ahora se crea una carpeta llamada “jobs” dentro de app y se agrega un archivo llamado “publisher\_job.rb”  
Y se escribe el siguiente código:



```

1 class PublisherJob < ApplicationJob
2   after_create :notify_creation
3
4   def notify_creation
5     PublisherJob.perform_later self
6   end
7 end
8

```

```

class PublisherJob < ApplicationJob
  queue_as :default

  def perform(message)
    begin
      PublisherService.publish(message)
    rescue StandardError => err
      puts "Error publishing message: #{message}, error: #{err}"
    end
  end
end

```

end

end

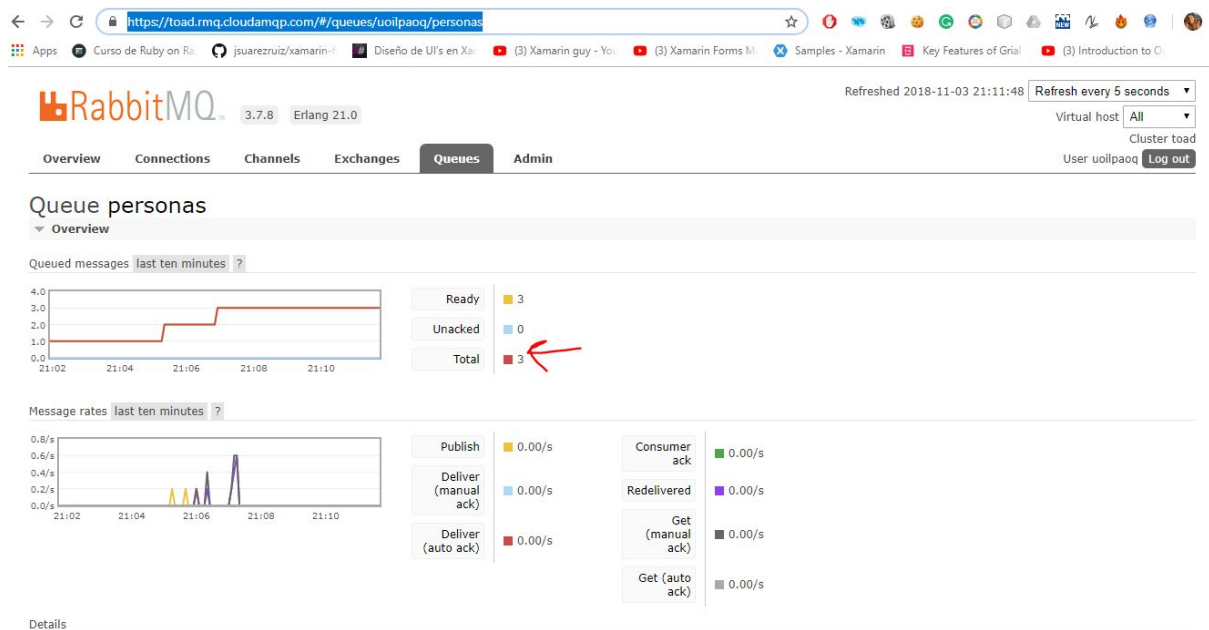
Para ver si llegó hay que ir a

<https://toad.rmq.cloudamqp.com/#/queues/uoilpaq/personas>

siendo **uoilpaq** el nombre del usuario

siendo **personas** el nombre de la queue

Donde se puede ver los mensajes que le llegan:



## CONCLUSION

Lo que acabamos de hacer es crear el proyecto Publisher, que lo que hace es que cuando crea una persona, luego ejecuta un job que es el que coloca en la cola el mensaje, el “self” es como si fuera el this, por lo que estaría enviando por la cola a la persona recién creada

# Proyecto Subscribe

En GemFile se agrega la gema bunny

```
Gemfile
1 source 'https://rubygems.org'
2
3 git_source(:name => :github) do |repo_name|
4   repo_name = "#{repo_name}/#{repo_name}" unless repo_name.include?("/")
5   "https://github.com/#{repo_name}.git"
6 end
7
8 gem 'bunny', '~> 2.12.0'
9
10 # Bundle edge Rails instead: gem 'rails', github: 'rails/rails'
11 gem 'rails', '~> 5.0.7'
12 # Use sqlite3 as the database for Active Record
13 gem 'sqlite3'
14 # Use Puma as the app server
15 gem 'puma', '~> 3.0'
16 # Use SCSS for stylesheets
17 gem 'sass-rails', '~> 5.0'
18 # Use Uglifier as compressor for JavaScript assets
19 gem 'uglifier', '~> 1.3.0'
20 # Use CoffeeScript for .coffee assets and views
21 gem 'coffee-rails', '~> 4.2'
22 # See https://github.com/rails/execjs#readme for more supported runtimes
23 # gem 'therubyracer', platforms: :ruby
24
25 # Use jquery as the JavaScript library
26 gem 'jquery-rails'
27 # Turbolinks makes navigating your web application faster. Read more: https://github.com/turbolinks/turbolinks
28 gem 'turbolinks', '~> 5'
29 # Build JSON APIs with ease. Read more: https://github.com/rails/jbuilder
30 gem 'jbuilder', '~> 2.5'
31 # Use Redis adapter to run Action Cable in production
32 # gem 'redis', '~> 3.0'
33 # Use ActiveModel has_secure_password
34 # gem 'bcrypt', '~> 3.1.7'
```

bundle install

Se crea una carpeta dentro de app que se llama services, y luego un archivo llamado subscribe\_service.rb

Y dentro el siguiente código:

```
class SubscriberService
```

```
  def self.receive()
    q = channel.queue("posts", durable: true)
    q.subscribe(block: true) do |delivery_info, properties, payload|
      puts "Received #{payload}, message properties are #{properties.inspect}"
    end
  end
end
```

```
  def self.channel
    @channel ||= connection.create_channel
  end
end
```

```
  def self.connection
    @connection ||= Bunny.new(
      host: "toad.rmq.cloudamqp.com",
```

```
    port: 5672,  
    ssl:false,  
    vhost: "nvpvbhwr",  
    user: "nvpvbhwr",  
    pass: "vm15IVPwsXPWXlQe17Z7gy-c7c3Iso1v",  
    heartbeat: :server, # will use RabbitMQ setting  
    frame_max: 131072,  
    auth_mechanism: "PLAIN"  
  ).start  
end  
  
end
```

Hay que asegurarse de cambiar lo que esta en rojo

## Tutoriales que sèrguì:

<https://bitbucket.org/pablovilas/pub-sub-example/src>

<https://www.rabbitmq.com/getstarted.html>

[http://rubybunny.info/articles/getting\\_started.html](http://rubybunny.info/articles/getting_started.html)

<https://github.com/ruby-amqp/bunny>