Sistema generador de recibos de pago

Edgar Ivan Contreras Lopez
5 de junio de 2018

1. Resumen

Sistema que se encarga de generar archivos con información de pagos online estos tienen como datos principales: codigo, producto y total. Estos datos servirán para identificar el recibo de pago. De igual manera se generar documentos random de recibos cuya finalidad podría ser el la de poder comprobar la correcta funcionalidad de páginas web a través de la ejecución del sistema generador de recibos de pago.

2. Palabras clave

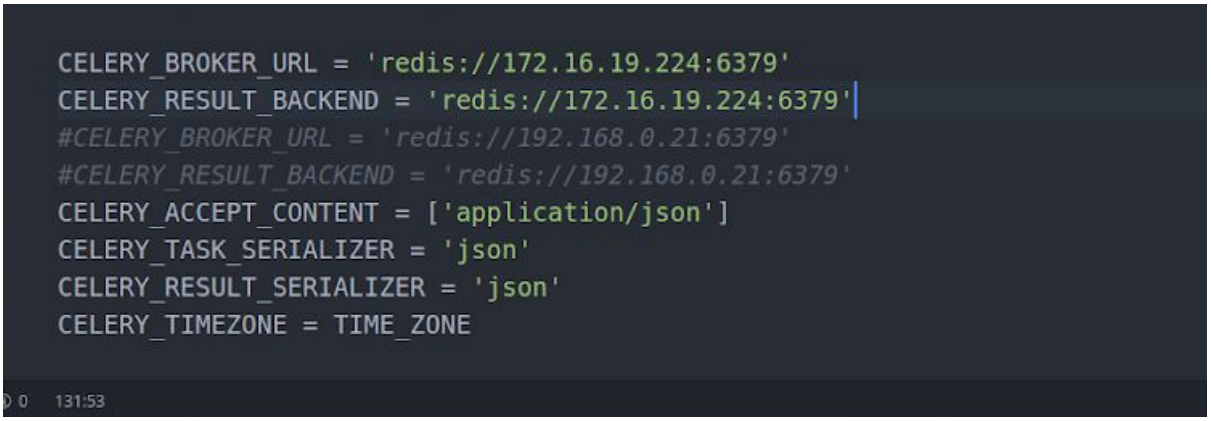
redis, celery, server, beat, sistema de compra, recibos.

3. Ejecución de pruebas

La configuración de installed_APPS se encuentra en la parte superior del archivo. Esta contiene los nombres de todas las aplicaciones Django que están activadas en esta instancia de Django, aqui iran todas nuestras aplicaciones creadas con anterioridad, en. Además en esta parte es donde realizamos las configuraciones de la ip para poder conectarnos con nuestro servidor en este caso ingresamos la ip de la otra computadora que es 172.16.19.224.

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'appDatos',
]
```

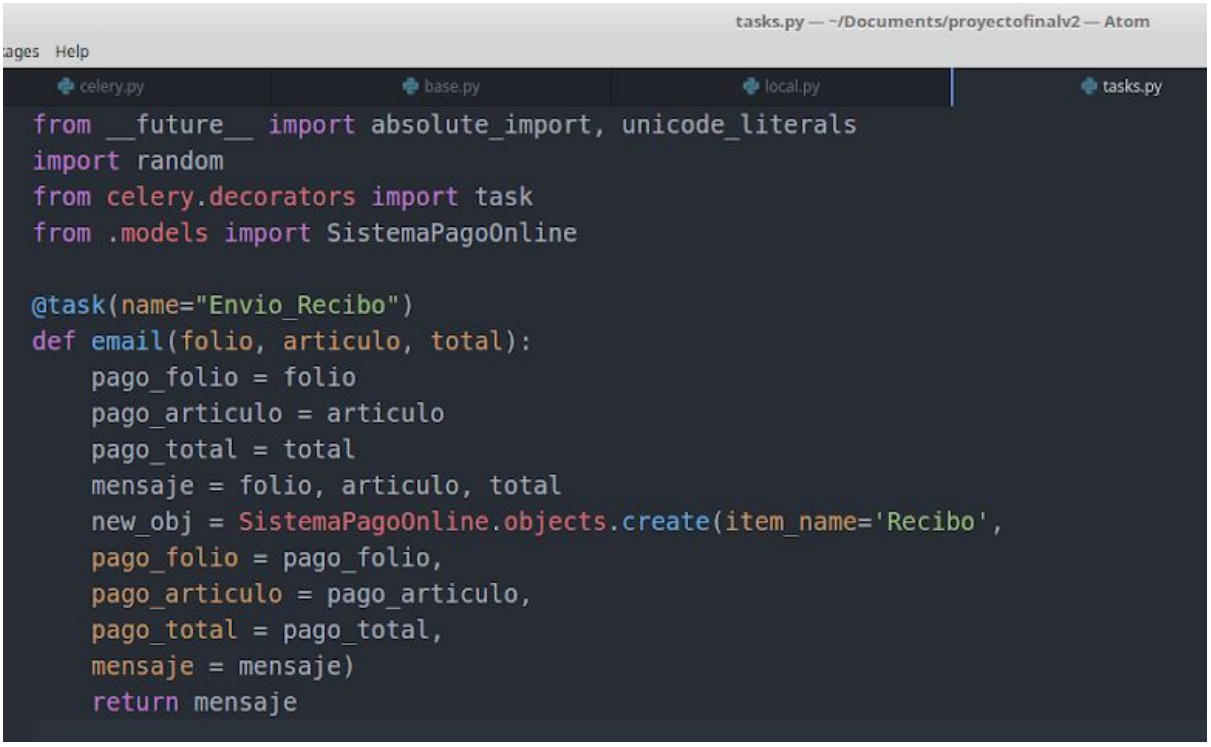
settings.py



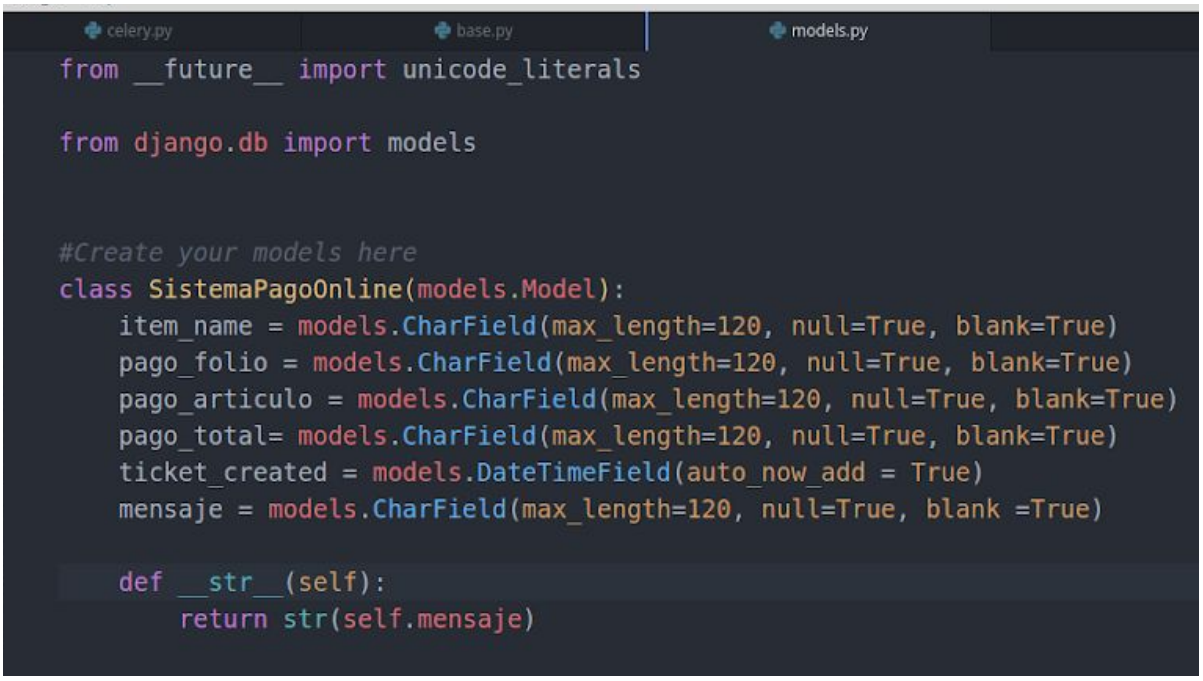
Settings.py

admin.py

En el archivo de task.py vamos a generar la información que va a contener el documento, y el documento tendrá la siguiente información: pago_folio, pago_artículo, pago_total, mensaje, new_obj. Y esa información se verá reflejada en el sistema online, previamente instalado y abierto.



Task.py



Models.py

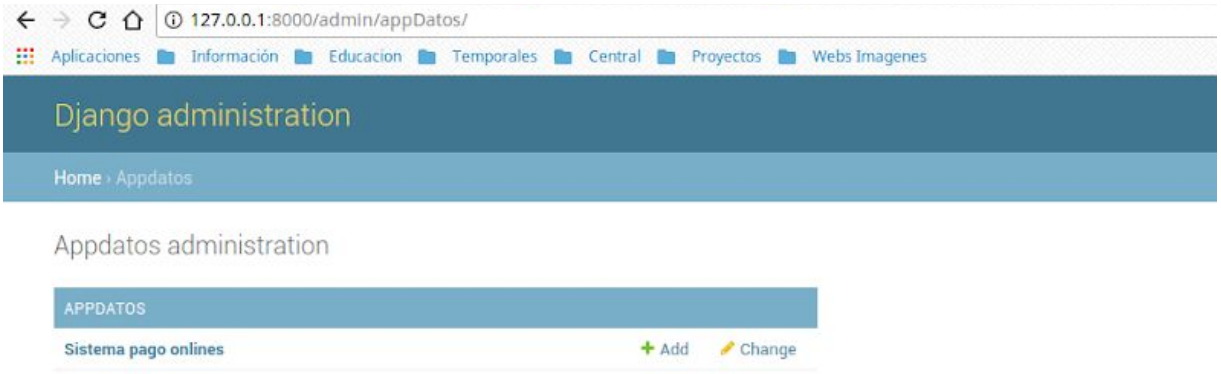
Celery es una aplicación que nos permite crear tareas de trabajo asíncronas gestionadas por un gestor de colas que está basada en el envio de mensajes de manera distribuida. Se focaliza en operaciones en tiempo real pero también soporta la calendarización de tareas, es decir, puede lanzar tareas que se tengan que ejecutar en un momento determinado o de manera periódica. En el archivo de celery

vamos a definir el tiempo que se tiene que generar, en este caso se va a generar documento en 5, 10 y 15 segundos estos datos se pueden editar pero estos 1 son los valores predeterminados. Cada que pase el tiempo predeterminado lo que se hará será generar archivos con los datos previamente ingresados

```python
from __future__ import absolute_import, unicode_literals
import os
from celery import Celery
from celery.schedules import crontab

#Configurar con el mismo ambient que wsgi.py
os.environ.setdefault("DJANGO_SETTINGS_MODULE", "proyecto.settings")
app = Celery('proj')
app.config_from_object('django.conf:settings', namespace = 'CELERY')
app.autodiscover_tasks()

#tareas agendadas
app.conf.beat_schedule = {
    'app-every-5-seconds':{
        'task':'Envio_Recibo',
        'schedule':5.0,
        'args':('9172642325','Recibido','$10000')},
    'app-every-10-seconds':{
        'task':'Envio_Recibo',
        'schedule':10.0,
        'args':('2454363637','No_Recibido','$500')},
    'app-every-15-seconds':{
        'task':'Envio_Recibo',
        'schedule':15.0,
        'args':('6346574745','Recibido','$7800')},
    }

@app.task(bind=True)
def debug_task(self):
    print('Request: {0!r}'.format(self.request))
#Tarea que es @app Unidad 3
```

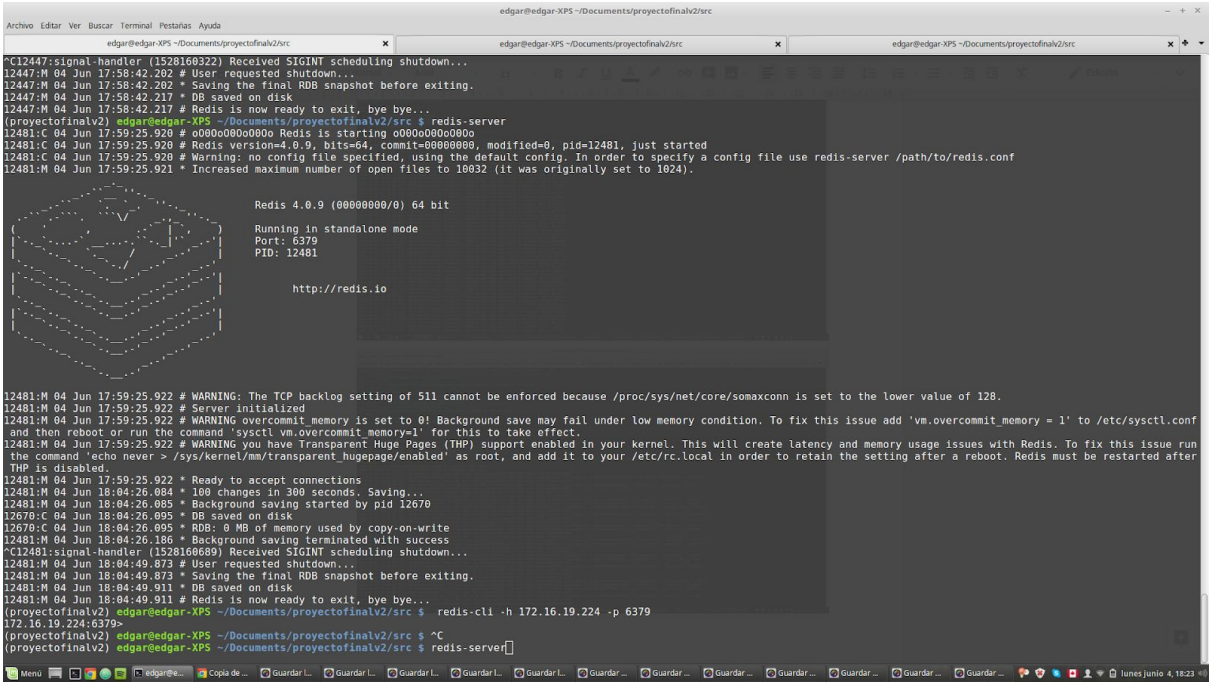
Celery.py

En este paso ingresamos al panel de administrador y observamos que se creó el modelo de manera correcta. Ingresamos a la parte de sistema pago online a ver si se registraron de manera correcta los valores en el modelo, y veremos que así es los datos se generaron de manera correcta.



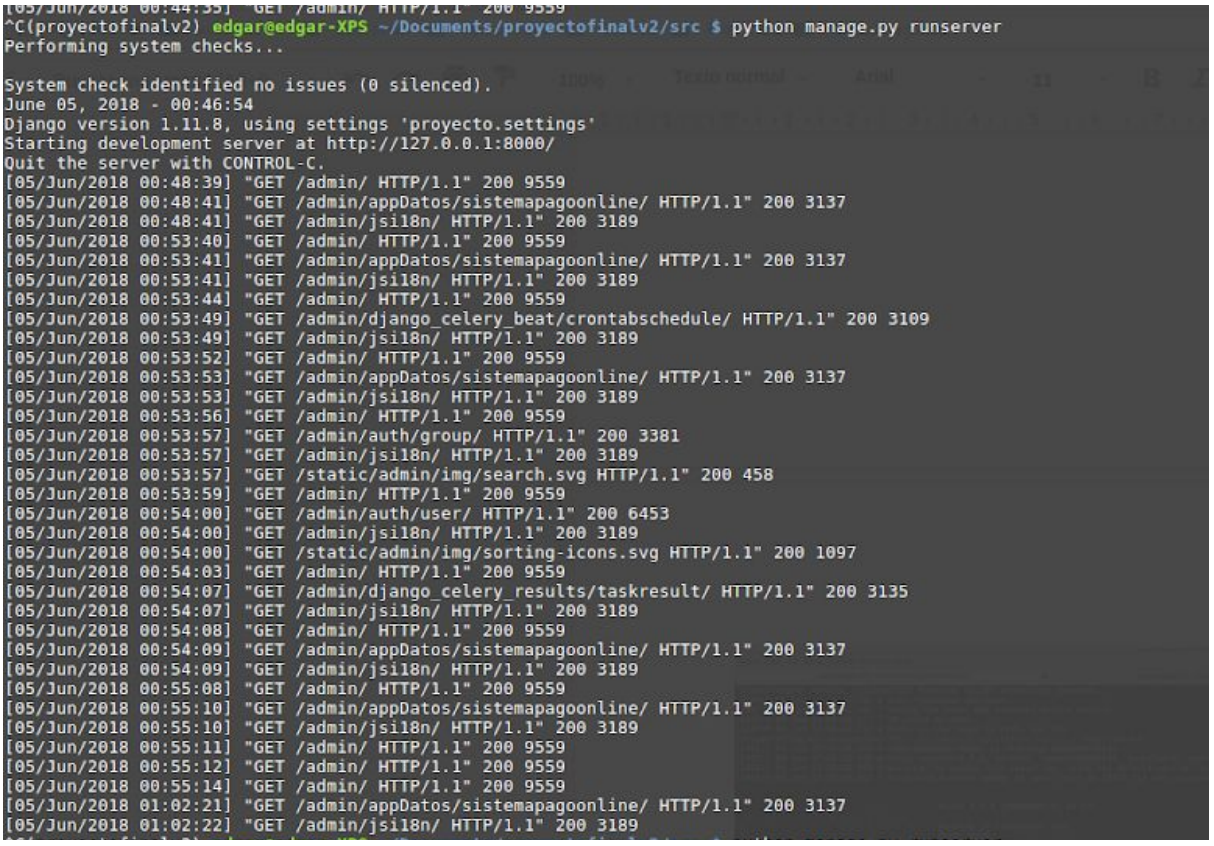
Django-admin

Primeramente, vamos a iniciar desde otra computadora que será nuestro nodo de redis, en esta parte levantamos nuestro servidor de redis. Para poder lograr la conexión con el otro computador.



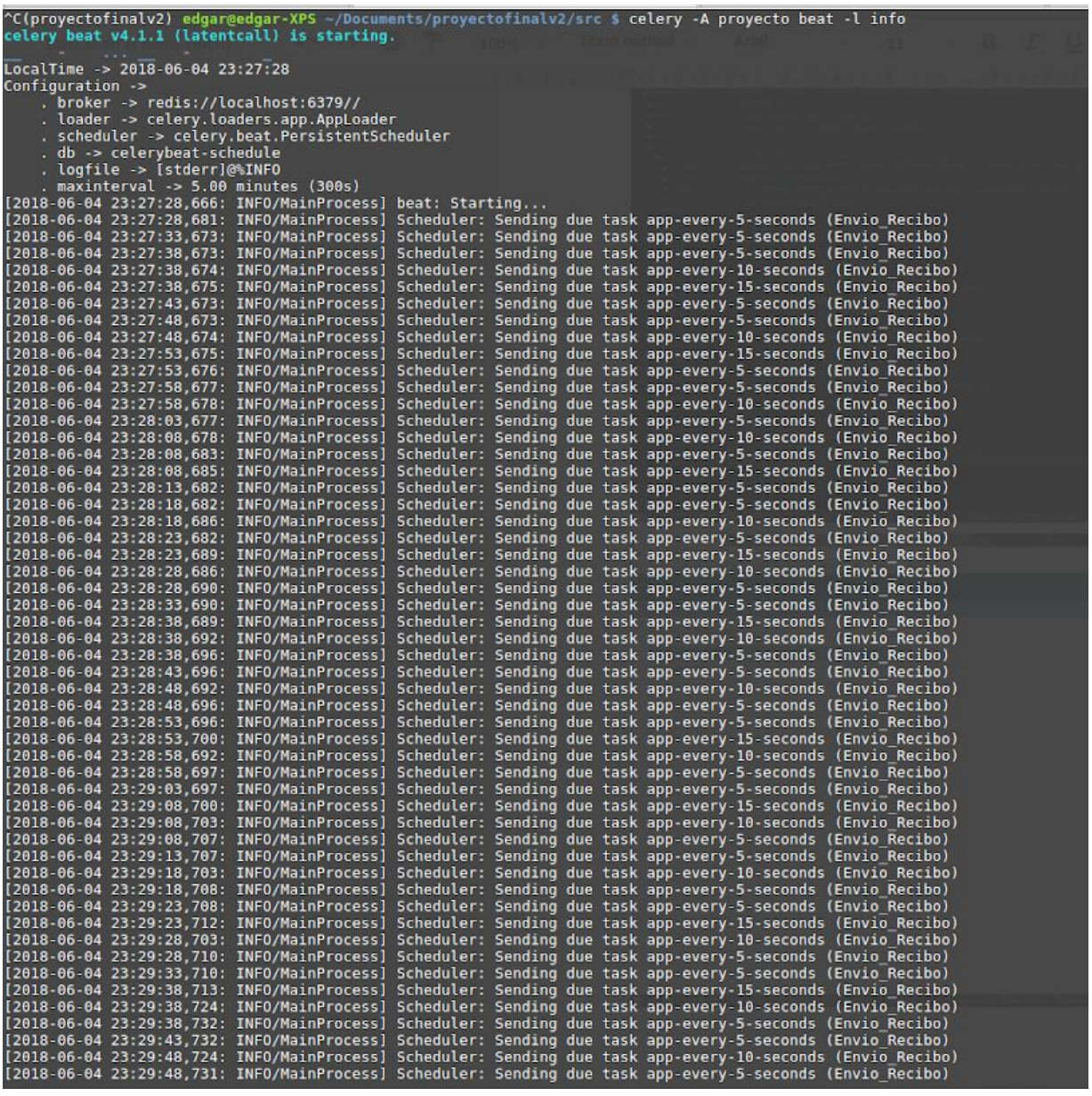
redis-server

Posteriormente vamos a iniciar nuestro server con el cual podremos revisar el correcto funcionamiento de nuestro programa debido y posteriormente en nuestro entorno de conexión entre otro computador.



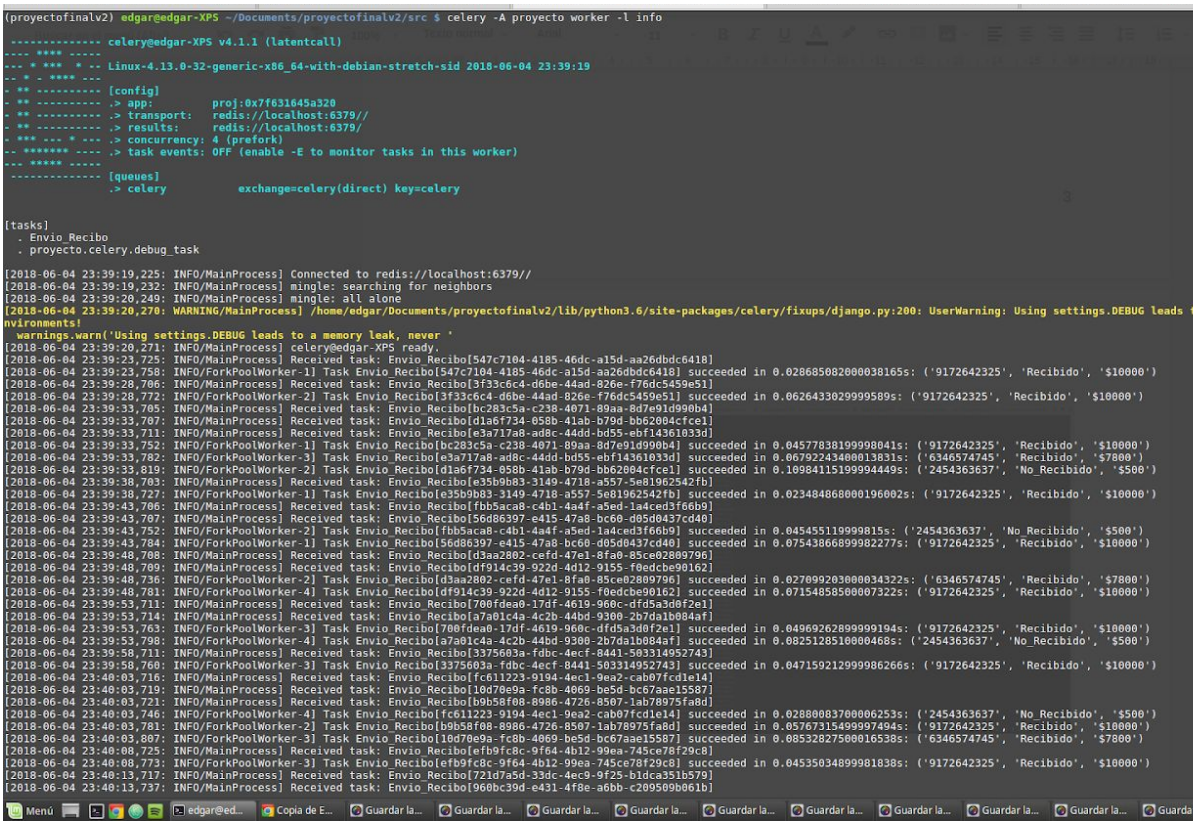
python manage.py runserver

Reiniciamos nuestro celery beat, lo volvemos a levantar y comenzará mandar las tareas, eso quiere decir que ya establece la conexión con redis, y de esta formas sabemos que los datos se envían de una manera adecuad al otro computador.



Beat

En el worker realizamos las mismas operaciones tumbamos y levantamos worker miraremos como se empiezan a ejecutar las tareas que se mandaron en el celery beat.



Worker

5. Conclusión

El proyecto se puede implementar en una gran cantidad de actividades como unas de ellas es la de hacer pruebas para poder comprobar el correcto funcionamiento de los sistemas de pago de las páginas web. Debido a que el sistema puede generar datos de manera aleatoria y esto se podría utilizar en beneficio del usuario