STRIPE

Global API Key
Server-side language
Let’s install libraries into virtualenv

pip install stripe

I will explain it with a django example. By the first i need create a new application for payment.

urls.py lets add follow lines:

```
# payment/urls.py
from django.urls import path, include
from .views import payment_test, process, payment_canceled, payment_completed,
stripe_webhook

app_name = 'payment'

urlpatterns = [
        path('payment/process/', process , name='process'),
        path('payment/payment_completed/', payment_completed ,
name='payment_completed'),
        path('payment/payment_canceled/', payment_canceled ,
name='payment_canceled'),
        path('payment/stripe/webhook/', stripe_webhook, name='stripe-webhook'),
]
```

each of those paths corresponds for some functions for example:

- **payment_canceled**-print a html page with informations about payment error

```
def payment_canceled(request):
        context = {
        'title': 'Payment Canceled!',
        'redirect_url': '/',
        'msg': f"Payment Canceled!",
        'contact_Paula_Serrano': "paulaserranoeducacao@gmail.com"
        }

        return render(request, 'index.html', context)
```

- **payment_completed**-print a html page with information about payment success

```
def payment_completed(request):
        context = {
        'title': "Payment successful!" ',
        'success': True,
        'redirect_url': '/',
        'msg':f"Payment successful!" ,
        'contact_Paula_Serrano':"paulaserranoeducacao@gmail.com"
        }
        return render(request, 'index.html', context)
```

- **process-** is logic to payment

```
def process(request):
        items_to_pay = []

        success_url =
request.build_absolute_uri(reverse('payment:payment_completed'))
        cancel_url =
request.build_absolute_uri(reverse('payment:payment_canceled'))

        #teste real payment
        session_data = {
        'payment_method_types': ['card'],
        #'client_reference_id': '1',
        #'customer': customer.id,
        'line_items': [
        {
                'price_data': {
                'currency': 'eur', #Euro
                'product_data': {
                'name': "Title item",
                },
                'unit_amount': 200,  # price in cents=2
                },
                'quantity': 1,
        },
        ],
        'mode': 'payment',
        'payment_intent_data': {
        'setup_future_usage': 'off_session'
        },
        'success_url': success_url,
        'cancel_url': cancel_url,
        }

        session = stripe.checkout.Session.create(**session_data)

        return redirect(session.url, code=303)
```

- **stripe_webhook-**notifications

```
@csrf_exempt
def stripe_webhook(request):
        payload = request.body
        sig_header = request.headers.get("Stripe-Signature")

        try:
        event = stripe.Webhook.construct_event(
        payload, sig_header, settings.STRIPE_ENDPOINT_SECRET
        )
        except ValueError as e:
        print("Ошибка ValueError:", str(e))  # Логи о проблемах с парсингом
        return JsonResponse({"error": "Invalid payload"}, status=400)
        except stripe.error.SignatureVerificationError as e:
        print("Ошибка подписи Stripe:", str(e))  # Логи о неверной подписи
        return JsonResponse({"error": "Invalid signature"}, status=400)

        if event["type"] == "checkout.session.completed":
        session = event["data"]["object"]
        customer_details = session.get("customer_details")
        customer_email = customer_details.get("email")
        payment_intent_id = session.get("payment_intent")

        if payment_intent_id:
        charges = stripe.Charge.list(payment_intent=payment_intent_id).get("data", [])
        if charges:
                receipt_url = charges[0].get("receipt_url")
                if customer_email and receipt_url:
                subject = "Your Receipt from Our Site"
                message = (f"Thanks for using our site. You can get your receipt
clicking on link: \n {receipt_url}")
                send_mail(
                subject,
                message,
                settings.EMAIL_HOST_USER,
                [customer_email])
        else:
        print(f"New evente: {event['type']}")

        return JsonResponse({"status": "success"}, status=200)

def payment_test(request):
        return redirect('payment:process')
```