django-daraja Documentation

Release 0.0.1

Martin Mogusu

May 27, 2023

Contents

This is a django library that interacts with the MPESA Daraja API. Source code can be found https://github.com/martinmogusu/django-daraja.git MPESA Daraja documentation can be found at https://developer.safaricom.co.ke

CHAPTER 1

Installation

To install the library, run:

```
pip install django_daraja
```

CHAPTER 2

Documentation

2.1 Introduction

This is a django library to interact with the Safaricom MPESA Daraja API (https://developer.safaricom.co.ke)

The source code can be found at https://github.com/martinmogusu/django-daraja.git

2.1.1 Installation

To install the package, run

```
$ pip install django_daraja
```

2.1.2 Example

An example, to send an STK push prompt to customer phone, then display response message

Listing 1: views.py

```
from django_daraja.mpesa.core import MpesaClient

def index(request):
    cl = MpesaClient()
    phone_number = '0700111222'
    amount = 1
    account_reference = 'reference'
    transaction_desc = 'Description'
    callback_url = 'https://api.darajambili.com/express-payment'
    response = cl.stk_push(phone_number, amount, account_reference,␣
↪transaction_desc, callback_url)
    return HttpResponse(response)
```

On your browser, you will receive the API response message and on the phone number specified you will receive an MPESA PIN prompt. Once you have entered the PIN, you will receive a notification on the callback URL you provided. If you used the exact callback URL in the example above (i.e. https://api.darajambili.com/express-payment), you can head over to https://darajambili.com to view the notification received.

2.2 Quick Start Guide

This is a quick start guide on seting up a simple project and implement some features of the django-daraja library.

2.2.1 1. Install

To install the package, run

```
$ pip install django_daraja
```

2.2.2 2. Create a django project

Run these commands to create a django project

```
$ django-admin startproject my_site
$ cd my_site
$ django-admin startapp my_app
```

2.2.3 3. Create a developer app

Head to https://developer.safaricom.co.ke and create a developer account, log in and create an app. You will use the **Consumer Key** and **Consumer Secret** of this app, as well as the **test credentials** assigned to you for the next step.

2.2.4 4. Environment Configuration

Hint: Test credentials (for sandbox testing) can be found at https://developer.safaricom.co.ke/test_credentials.

Add the details below at the bottom of your *settings.py* file

Warning: Adding sensitive configuration in the settings file is not recommended on production since you will most likely NOT want to have configuration settings - e.g. consumer keys/secrets - as part of your code, which will be added to version control. It is recommended to use a .env file and a library like *python-decouple* so that the configuration can be externalized when deploying to production

Listing 2: my_site/settings.py

```
# The Mpesa environment to use
# Possible values: sandbox, production

MPESA_ENVIRONMENT = 'sandbox'

# Credentials for the daraja app

MPESA_CONSUMER_KEY = 'mpesa_consumer_key'
MPESA_CONSUMER_SECRET = 'mpesa_consumer_secret'

#Shortcode to use for transactions. For sandbox  use the Shortcode 1␣
↪provided on test credentials page

MPESA_SHORTCODE = 'mpesa_shortcode'

# Shortcode to use for Lipa na MPESA Online (MPESA Express) transactions
# This is only used on sandbox, do not set this variable in production
# For sandbox use the Lipa na MPESA Online Shorcode provided on test␣
↪credentials page

MPESA_EXPRESS_SHORTCODE = 'mpesa_express_shortcode'

# Type of shortcode
# Possible values:
# - paybill (For Paybill)
# - till_number (For Buy Goods Till Number)

MPESA_SHORTCODE_TYPE = 'paybill'

# Lipa na MPESA Online passkey
# Sandbox passkey is available on test credentials page
# Production passkey is sent via email once you go live

MPESA_PASSKEY = 'mpesa_passkey'

# Username for initiator (to be used in B2C, B2B, AccountBalance and␣
↪TransactionStatusQuery Transactions)

MPESA_INITIATOR_USERNAME = 'initiator_username'

# Plaintext password for initiator (to be used in B2C, B2B, AccountBalance␣
↪and TransactionStatusQuery Transactions)

MPESA_INITIATOR_SECURITY_CREDENTIAL = 'initiator_security_credential'
```

2.2.5 5. Settings configuration

In `settings.py`, add `django_daraja` and `my_app` to the `INSTALLED_APPS` list

Listing 3: my_site/settings.py

```
INSTALLED_APPS = [
    ...,
    'django_daraja',
```

(continues on next page)

(continued from previous page)

```
    'my_app',
]
```

2.2.6 6. URL Configuration

In `urls.py`, Add the URL configuration

Python 2:

Listing 4: my_site/urls.py

```python
from django.urls import re_path as url, include
from django.contrib import admin

urlpatterns = [
    url(r'^admin/', admin.site.urls),
    url(r'^', include('my_app.urls')),
]
```

Python 3:

Listing 5: my_site/urls.py

```python
from django.urls import path, include
from django.contrib import admin

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', include('my_app.urls')),
]
```

In `my_app/urls.py`, add the code to create a home page, as well as the endpoint to receive notifications from MPESA

Python 2:

Listing 6: my_app/urls.py

```python
from django.urls import re_path as url

from . import views

urlpatterns = [
    url(r'^$', views.index, name='index')
]
```

Python 3:

Listing 7: my_app/urls.py

```python
from django.urls import path
from . import views

urlpatterns = [
    path('', views.index, name='index')
]
```

2.2.7 7. Create a view

In `my_app/views.py` Create a test index view

Listing 8: my_app/views.py

```
from django.shortcuts import render
from django.http import HttpResponse
from django_daraja.mpesa.core import MpesaClient

def index(request):
    cl = MpesaClient()
    # Use a Safaricom phone number that you have access to, for you to be␣
↪able to view the prompt.
    phone_number = '07xxxxxxxx'
    amount = 1
    account_reference = 'reference'
    transaction_desc = 'Description'
    callback_url = 'https://api.darajambili.com/express-payment'
    response = cl.stk_push(phone_number, amount, account_reference,␣
↪transaction_desc, callback_url)
    return HttpResponse(response)
```

Note:

- Use a Safaricom number that you have access to for the `phone_number` parameter, so as to be able to receive the M-PESA prompt on your phone.
- Once you have entered the PIN, you will receive a notification on the callback URL you provided. If you used the exact callback URL in the example above (i.e. https://api.darajambili.com/express-payment), you can head over to https://darajambili.com to view the notification received.

2.2.8 8. Run Migrations

On the command line, run migrations to add the models created by the library

```
$ python manage.py migrate
```

2.2.9 9. Run the server

Then run the server

```
$ python manage.py runserver
```

You can now visit your site at `localhost:8000` to view your project

If the STK push was successful, you should see an STK prompt on your phone (the phone number you provided), and you should see the response on the browser. It looks like this:

```
{
    "MerchantRequestID": "2134-9231241-1",
    "CheckoutRequestID": "ws_CO_DMZ_157917982_20112018173133556",
    "ResponseCode": "0",
```

(continues on next page)

(continued from previous page)

```
    "ResponseDescription": "Success. Request accepted for processing",
    "CustomerMessage": "Success. Request accepted for processing"
}
```

You will also receive a notification on the callback endpoint that you specified having the results of the STK push.

2.3 MPESA APIs

The django-daraja library supports the different MPESA APIs exposed by the Daraja platform. These are the supported APIs:

2.3.1 OAuth API

The OAUth API is used to generate an access token, which is used for authentication in all the other APIs.

Note: Functionality for the OAUth API has been automated in the django-daraja; the library will automatically generate an access token before making an API call and attach it to the request headers of the API call. Access tokens expire after an hour, so this library stores the access token for a maximum of 50 minutes to avoid repeated calls to the OAuth endpoint.

You can test the OAuth API by using the `MpesaClient.access_token` method

```
from django_daraja.mpesa.core import MpesaClient

cl = MpesaClient()
token = cl.access_token()
```

This will assign the `token` variable with an access token generated from the OAuth endpoint, or stored locally if available.

2.3.2 STK Push API

The STK Push API is used to push a prompt to a customer's phone, asking the customer to enter a PIN.

This simplifies the process of C2B payments, since the business can specify all the necessary parameters for the payment (e.g. amount, shortcode e.t.c), so that the customer will just enter their MPESA PIN to authorize the payment.

Example:

```
from django_daraja.mpesa.core import MpesaClient

phone_number = '07xxxxxxxx'
amount = 1
account_reference = 'reference'
transaction_desc = 'Description'
callback_url = 'https://api.darajambili.com/express-payment'
response = cl.stk_push(phone_number, amount, account_reference, transaction_
↪desc, callback_url)
```

This will assign the `response` variable with an `MpesaResponse` object containing the response returned from the STK Push API call.

Note:

- Use a Safaricom number that you have access to for the `phone_number` parameter, so as to be able to receive the M-PESA prompt on your phone.
- Once you have entered the PIN, you will receive a notification on the callback URL you provided. If you used the exact callback URL in the example above (i.e. https://api.darajambili.com/express-payment), you can head over to https://darajambili.com to view the notification received.

2.3.3 B2C Payment APIs

The B2C Payment APIs are used to make Business to Customer payments. Currently 3 transactions are supported in this model:

Business Payment This is a normal business to customer payment, supports only M-Pesa registered customers.

Salary Payment This supports sending money to both registere and unregistered M-Pesa customers.

Promotion Payment This is a promotional payment to customers. The M-Pesa notification message is a congratulatory message. Supports only M-Pesa registered customers.

Examples:

Business Payment

```
from django_daraja.mpesa.core import MpesaClient

phone_number = '07xxxxxxxx'
amount = 1
transaction_desc = 'Description'
occassion = 'Occassion'
callback_url = 'https://api.darajambili.com/b2c/result'
response = self.cl.business_payment(phone_number, amount, transaction_desc,␣
↪self.callback_url, occassion)
```

Salary Payment

```
from django_daraja.mpesa.core import MpesaClient

phone_number = '07xxxxxxxx'
amount = 1
transaction_desc = 'Description'
occassion = 'Occassion'
callback_url = 'https://api.darajambili.com/b2c/result'
response = self.cl.business_payment(phone_number, amount, transaction_desc,␣
↪self.callback_url, occassion)
```

Promotion Payment

```
from django_daraja.mpesa.core import MpesaClient

phone_number = '07xxxxxxxx'
amount = 1
transaction_desc = 'Description'
occassion = 'Occassion'
callback_url = 'https://api.darajambili.com/b2c/result'
response = self.cl.promotion_payment(phone_number, amount, transaction_desc,␣
↪self.callback_url, occassion)
```

This will assign the `response` variable with an `MpesaResponse` object containing the response returned from the Business Payment B2C API Call

Note:

- Test credentials to use for this scenario can be found at the developer portal (https://developer.safaricom.co.ke/test_credentials)
- Use *shortcode 1* as the shortcode, and the test MSISDN as the B2C phone number
- Once the transaction is complete, you will receive a notification on the callback URL you provided. If you used the exact callback URL in the example above (i.e. https://api.darajambili.com/b2c/result), you can head over to https://darajambili.com to view the notification received

CHAPTER 3

Indices and tables

- genindex
- modindex
- search