## **Getting Started**

##### **Fetch code**

git clone <https://github.com/tharunkanna14/azure-api-server.git>

cd azure-api-server

##### **Run The Sample**

Create and start virtual environment

**python -m venv venv**

**source venv/bin/activate**

Install django

**pip install django**

Install dependencies

**pip** **install -r requirements.txt**

sample check in this project:

If you are deploying this in any cloud service like Azure, AWS etc., make sure you configure the workflow according to this **settings.py**

"""

Django settings for Blog project.

Generated by 'django-admin startproject' using Django 2.1.5.

For more information on this file, see

https://docs.djangoproject.com/en/2.1/topics/settings/

For the full list of settings and their values, see

https://docs.djangoproject.com/en/2.1/ref/settings/

"""

import os

# Build paths inside the project like this: os.path.join(BASE\_DIR, ...)

BASE\_DIR = os.path.dirname(os.path.dirname(os.path.abspath(\_\_file\_\_)))

# Quick-start development settings - unsuitable for production

# See https://docs.djangoproject.com/en/2.1/howto/deployment/checklist/

# SECURITY WARNING: keep the secret key used in production secret!

SECRET\_KEY = 'place your django secret key here'

# SECURITY WARNING: don't run with debug turned on in production!

DEBUG = False

# Make sure you replace the allowed hosts with ‘\*’ when deploying it in cloud

ALLOWED\_HOSTS = ['Your IP']

# Application definition

INSTALLED\_APPS = [

'django.contrib.admin',

'django.contrib.auth',

'django.contrib.contenttypes',

'django.contrib.sessions',

'django.contrib.messages',

'django.contrib.staticfiles',

'django.contrib.humanize',

'blogapp.apps.BlogappConfig',

'rest\_framework'

]

MIDDLEWARE = [

'django.middleware.security.SecurityMiddleware',

'django.contrib.sessions.middleware.SessionMiddleware',

'django.middleware.common.CommonMiddleware',

'django.middleware.csrf.CsrfViewMiddleware',

'django.contrib.auth.middleware.AuthenticationMiddleware',

'django.contrib.messages.middleware.MessageMiddleware',

'django.middleware.clickjacking.XFrameOptionsMiddleware',

]

ROOT\_URLCONF = 'Blog.urls'

TEMPLATES = [

{

'BACKEND': 'django.template.backends.django.DjangoTemplates',

'DIRS': [str(BASE\_DIR.join('templates'))],

'APP\_DIRS': True,

'OPTIONS': {

'context\_processors': [

'django.template.context\_processors.debug',

'django.template.context\_processors.request',

'django.contrib.auth.context\_processors.auth',

'django.contrib.messages.context\_processors.messages',

],

},

},

]

WSGI\_APPLICATION = 'Blog.wsgi.application'

# Database

# https://docs.djangoproject.com/en/2.1/ref/settings/#databases

DATABASES = {

'default': {

'ENGINE': 'django.db.backends.sqlite3',

'NAME': os.path.join(BASE\_DIR, 'db.sqlite3'),

}

}

# Password validation

# https://docs.djangoproject.com/en/2.1/ref/settings/#auth-password-validators

AUTH\_PASSWORD\_VALIDATORS = [

{

'NAME': 'django.contrib.auth.password\_validation.UserAttributeSimilarityValidator',

},

{

'NAME': 'django.contrib.auth.password\_validation.MinimumLengthValidator',

},

{

'NAME': 'django.contrib.auth.password\_validation.CommonPasswordValidator',

},

{

'NAME': 'django.contrib.auth.password\_validation.NumericPasswordValidator',

},

]

# Internationalization

# https://docs.djangoproject.com/en/2.1/topics/i18n/

LANGUAGE\_CODE = 'en-us'

TIME\_ZONE = 'Asia/Calcutta'

USE\_I18N = True

USE\_L10N = True

USE\_TZ = True

# Static files (CSS, JavaScript, Images)

# https://docs.djangoproject.com/en/2.1/howto/static-files/

STATIC\_URL = os.environ.get("DJANGO\_STATIC\_URL", "/static/")

STATIC\_ROOT = os.environ.get("DJANGO\_STATIC\_ROOT", "./static/")

DEFAULT\_AUTO\_FIELD = 'django.db.models.AutoField'

Run the server

**python manage.py runserver**

##### 

###### Future Ready Talent Project deployment proofs

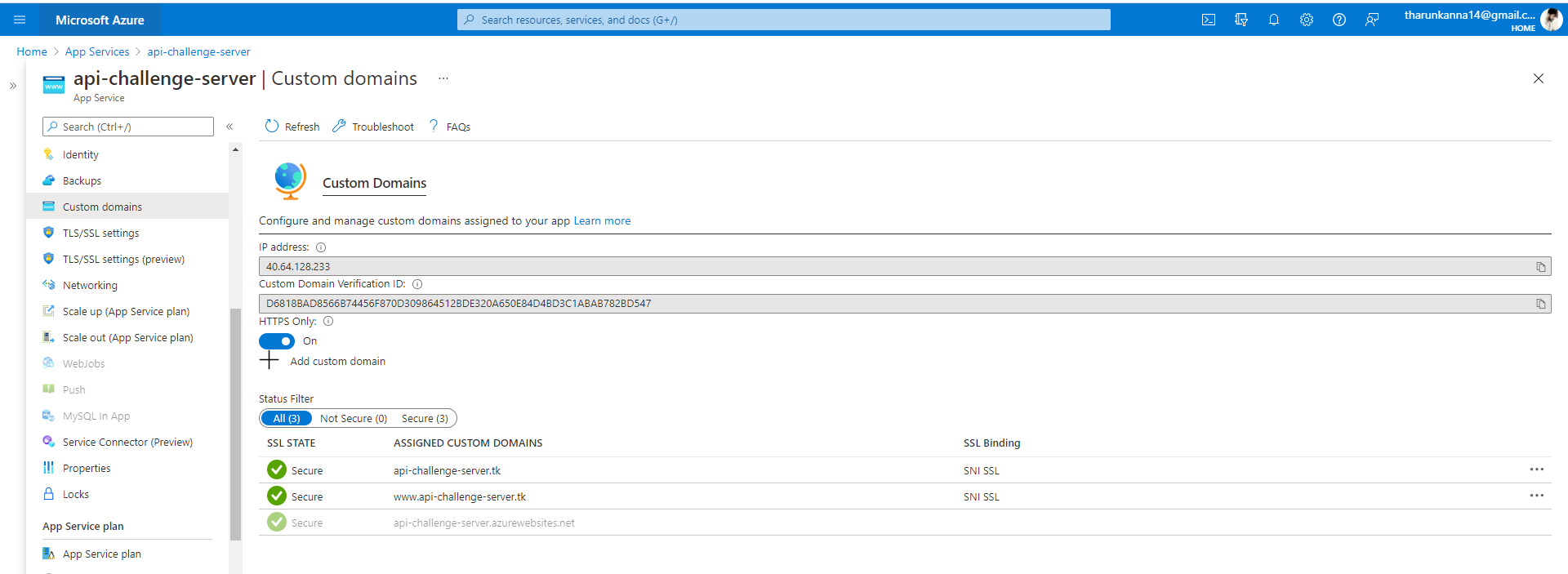
Using the API management service the CORS has been enabled for the website.

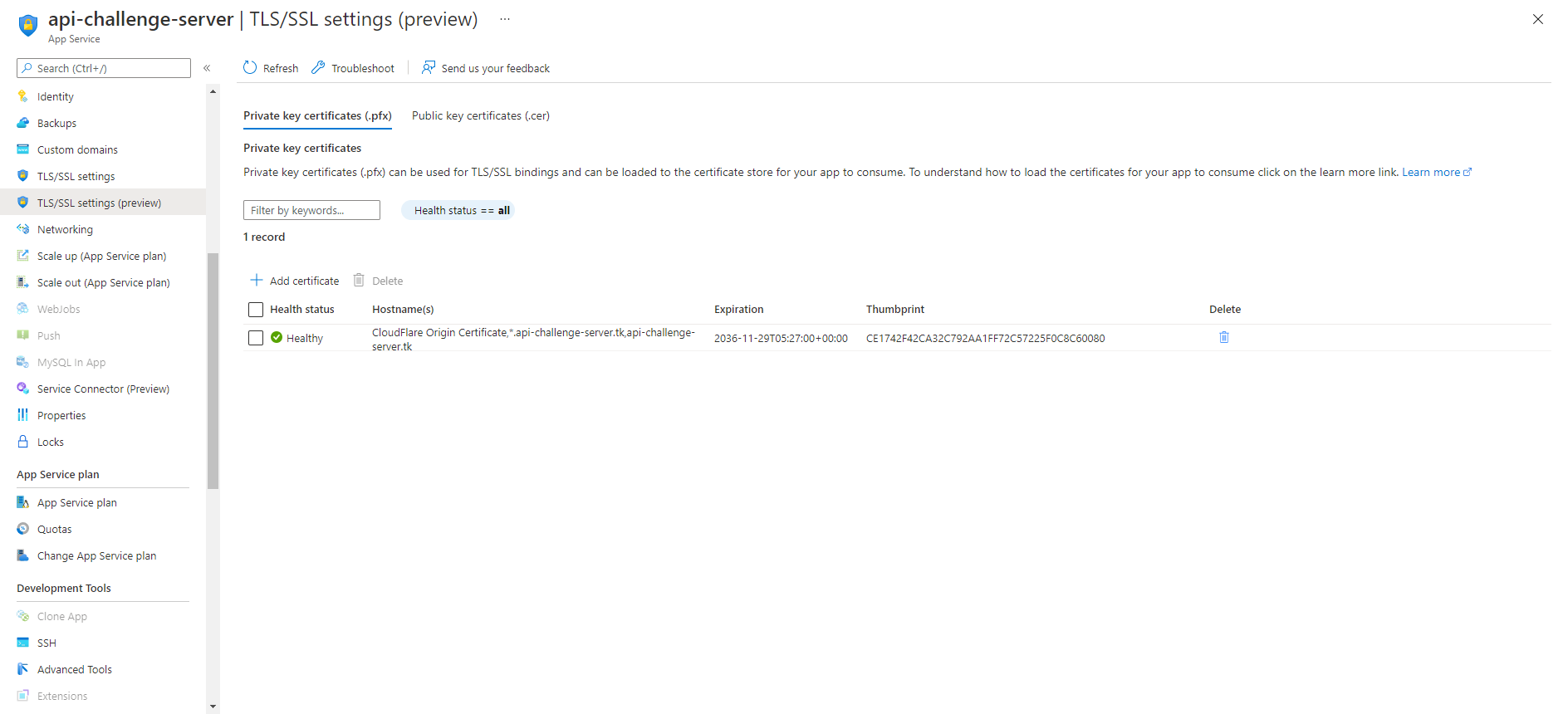
A secure HTTPS connection has also been established using Azure DNS and Azure Traffic Manager to withstand heavy workloads.

An application gateway load balancer has also been setup.

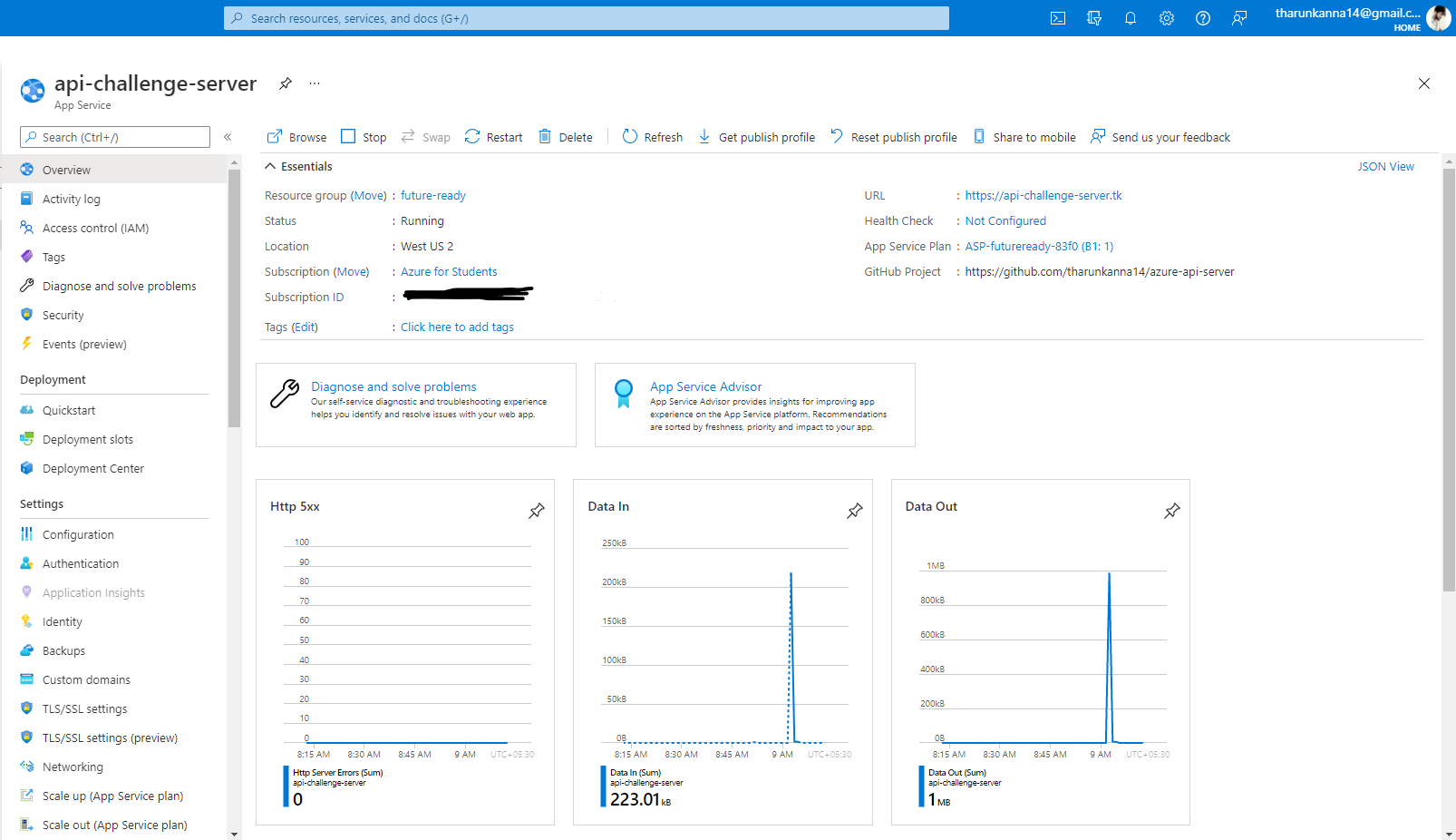
Custom domain name has also been set for users who can access this always on server to practice this challenge.

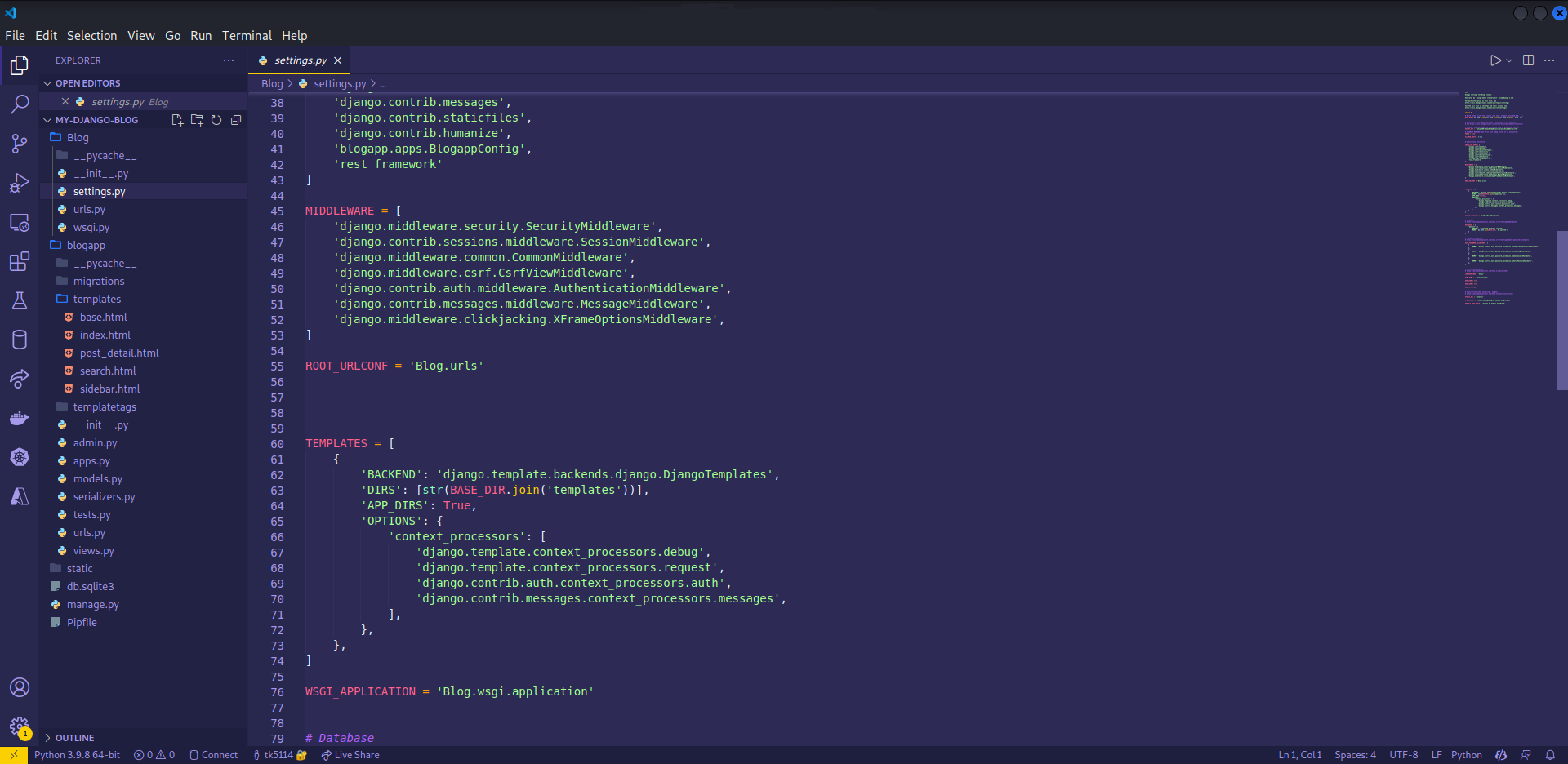
It is available on https://www.api-challenge-server.tk





##### **Azure deployment**

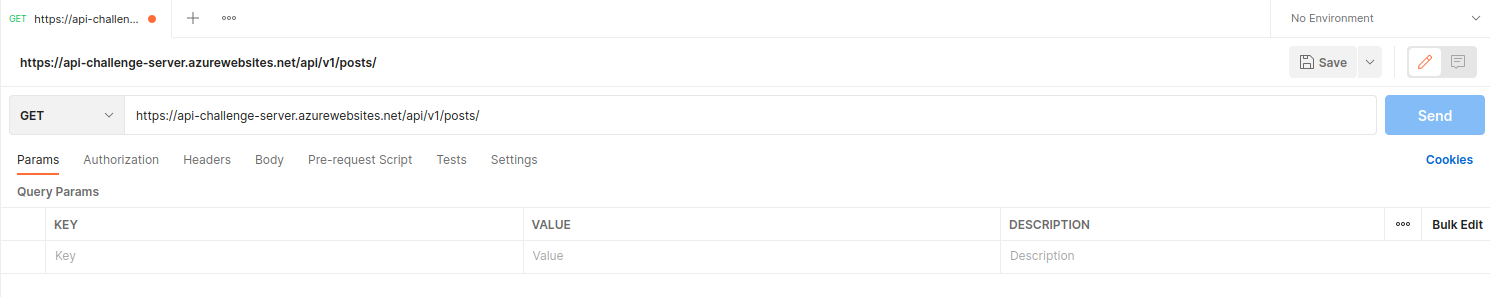
****



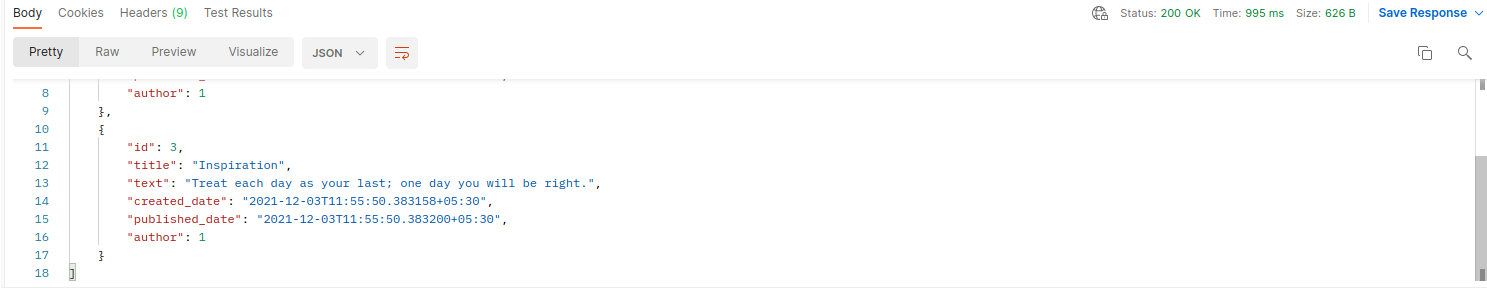
##### **Performing tests using Postman**

In your Postman workspace create a new collection and paste the link in the enter request URL.

Note: Make sure to append /api/v1/posts/ to the URL



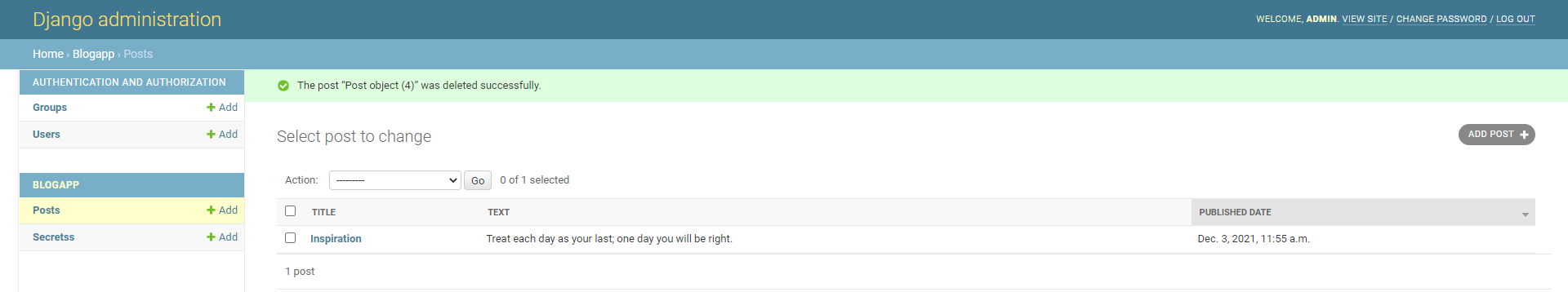
Choose a HTTP method of your choice and play around with the REST API. A sample GET method is shown below.



You can create your own flows using Postman which can give you an even better understanding.

##### **Database & Admin**

If you are new to django, there’s a db.sqlite file where all the data is being stored. You can explore the same by visiting the /admin page by appending it to the end of the URL.



##### **Results**

The CRUD operations are reflected in the main website i.e django central, where you get to see the HTTP methods performed as a flash card. Refresh the site once you have performed any operation using the /api/v1/posts/ page or Postman to see the changes.

