```python
import pynetbox
import csv
import time
from pprint import pprint
import re

SLEEP_TIME=3    # time to pause between each api call
IP_NETBOX_FILE = './ip-netbox.csv'

'''
# example format of the csv file, comments start in #

# column indices 0,1,2,3,4,5,6,7,8,9
#,Aggregate,Prefix,Tenant,TenantGroup,Status,Site,Description,Tags,Comments
# Example,10.103.0.0/16,10.103.10.0/23,"In the new devtest environment, IP allocated for â¦.","choose one of devtest, prod, staging, qa","Active, container, Reserved, Deprecated","aws
,,10.102.10.0/24,vnet-dev,devtest,active,azure,rg-dev,eastus,
,,10.102.20.0/24,vnet-prod,prod,active,azure,rg-prod,eastus,
,,10.102.0.0/24,vnet-staging,staging,active,azure,rg-staging,eastus,
,,10.0.0.0/16,vnet-test,qa,active,azure,rg-qa,eastus,

'''

netbox_url="http://198.168.56.76:80"
netbox_token="263136752a7d0a95d6xxxxxx4d828419b0b014"

def main(argv=None):
  update_tenant_groups()
  update_tenants()
  update_prefixes()

# function for creating slug names consistentnly
def get_slug(name):
  name = name.strip()

  char_to_remove = ["-", " "]
  for char in char_to_remove:
    name = name.replace(char, "_")

  name = name.lower()
  name = re.sub('_+', '_', name)
  return name

def update_tenant_groups():
  print("Inside tenant_groups")
  nbapi = pynetbox.api(url=netbox_url, token=netbox_token)

  #print("Getting all tenant groups")
  tgs = nbapi.tenancy.tenant_groups.all()

  # create an empty hash
  current_tgs = {}
  for tg in tgs:
    tg_slug = get_slug(str(tg))
    current_tgs[tg_slug] = ""
  #  print(str(tg))

  with open(IP_NETBOX_FILE) as csv_file:
    csv_reader = csv.reader(csv_file, delimiter=',')
    #print(row)
    for row in csv_reader:
      if ( '#' in row[0] ):
        #print ("Skipping " + str(row))
        continue
      else:
        #print ("Processing " + str (row))

        tg = get_slug(row[4].strip())
        #print ("tg " + tg)

        # Create tenant group only if non-empty
        if (tg != ''):
          if tg in current_tgs:
            print ("Tenant Group [" + tg + "] already exists ")
          else:
            print ("Creating tenant group [" + tg + "] ")
            #tg_slug = get_slug(tg)
            print("Adding: [" + tg + "]")
            create_tg = nbapi.tenancy.tenant_groups.create(name=tg, slug=tg)
            time.sleep(SLEEP_TIME)

            print("Added: [" + str(create_tg) + " " + tg + "]")
            current_tgs[str(tg)] = ""

  print ("Imported tenant group from CSV File")

def update_tenants():
  print("Inside tenants")
  time.sleep(SLEEP_TIME)
  nbapi = pynetbox.api(url=netbox_url, token=netbox_token)

  #print("Getting all tenants")
  tenants = nbapi.tenancy.tenants.all()

  # create an empty hash
  current_tenants = {}
  for tenant in tenants:
    tenant_slug=get_slug(str(tenant))
    current_tenants[tenant_slug] = ""
  #  print(str(tg))

  with open(IP_NETBOX_FILE) as csv_file:
    csv_reader = csv.reader(csv_file, delimiter=',')
    #print(row)
    for row in csv_reader:
      if ( '#' in row[0] ):
        #print ("Skipping " + str(row))
        continue
      else:
        #print ("Processing " + str (row))

        tenant_slug = get_slug(row[3].strip())
        #print ("tg " + tg)

        # Create tenant group only if non-empty
        if (tenant_slug != ''):
          if tenant_slug in current_tenants:
            print ("Tenant [" + tenant_slug + "] already exists ")
          else:
            group_slug=get_slug(row[4].strip())
            print ("Creating tenant [" + tenant_slug + "] ")
```

```
            time.sleep(SLEEP_TIME)
            tenant_group = nbapi.tenancy.tenant_groups.get(slug=group_slug)
            #print ("Slug here = " + tenant)
            #print (tenant_id.id)
            print ("retrieved tenant group id = " + str(tenant_group.id) )

            time.sleep(SLEEP_TIME)
            try:
              create_tenant = nbapi.tenancy.tenants.create(name=tenant_slug, slug=tenant_slug, group=tenant_group.id)
            except Exception as e:
              print("Exception occurred...")
              print(type(e))
              print(e)
              time.sleep(2*SLEEP_TIME)
              continue

            print("Added: [" + str(create_tenant) + "]")
            current_tenants[tenant_slug] = ""

  print ("Imported tenants from CSV File")

def update_prefixes():
  print("Inside update_prefixes ")
  nbapi = pynetbox.api(url=netbox_url, token=netbox_token)

  #print("Getting all prefixes")
  prefixes = nbapi.ipam.prefixes.all()

  # create an empty hash
  current_prefixes = {}
  for p in prefixes:
    tenant_slug = get_slug(str(p.tenant))
    prefix_slug = get_slug(str(p))
    current_prefixes[prefix_slug] = tenant_slug
    print('Adding hash: ' + prefix_slug + ' with value ' + tenant_slug )

  with open(IP_NETBOX_FILE) as csv_file:
    csv_reader = csv.reader(csv_file, delimiter=',')
    #print(row)
    for row in csv_reader:
      if ( '#' in row[0] ):
        #print ("Skipping " + str(row))
        continue
      else:
        #print ("Processing " + str (row))

        prefix = get_slug(row[2])
        tenant = get_slug(row[3])

        print ("Inside else: " + prefix + ' ' + tenant)

        #print ("tg " + tg)

        # Create tenant group only if non-empty
        if (prefix != ''):
          #print (prefix + ' value ' + current_prefixes[prefix] + ' ' + tenant)

          if ((prefix in current_prefixes) and (tenant == current_prefixes[prefix])):
            print ("Prefix [" + prefix + "] with value [" + str(current_prefixes[str(prefix)]) + " already exists in [tenant]... skipping...")
          else:
            print ("Creating prefixes [" + prefix + "], tenant = " + tenant)

            tenant_id = nbapi.tenancy.tenants.get(slug=tenant)
            print ("Tenant slug here = " + tenant)
            print ("retrieved tenant id = " + str(tenant_id.id) )

            time.sleep(SLEEP_TIME)
            create_prefix = nbapi.ipam.prefixes.create(prefix=prefix, tenant=tenant_id.id,
              status=row[5], #site=row[6],tags=row[8],
              description=row[7], comments=row[9])
            time.sleep(SLEEP_TIME)
            print("Added: [" + str(create_prefix) + "]")
            current_prefixes[prefix] = tenant

  print ("Imported tenant group from CSV File")

if __name__ == "__main__":
  main()
```