```python
# Source: https://stackoverflow.com/questions/31487478/how-to-convert-a-pptx-to-pdf-using-python
# Source: https://community.xibo.org.uk/t/using-python-to-automatically-update-xibo-datasets/19212

import configparser
import comtypes.client
import comtypes
import shutil
import os
import sys
import requests
from requests_toolbelt.multipart.encoder import MultipartEncoder
import json
import os
import mimetypes

# ------------------------
# VARIABLES
# ------------------------
config = configparser.ConfigParser()

# Config is read from same directory, where script is executed
# Configuration file must be named as settings.ini
config.read('settings.ini')
# Original PPTX file located on network share
orig_PPTX_file_path = config['DEFAULT']['orig_PPTX_file_path']
# Copied PPTX file, located on local HDD
PPTX_file_path = config['DEFAULT']['PPTX_file_path']
# Converted PDF file
PDF_file_path = config['DEFAULT']['PDF_file_path']
# Xibo address
Xibo_address = config['DEFAULT']['Xibo_address']
# Media name in Xibo, used for searching and replacing media (PDF)
Xibo_media_name = config['DEFAULT']['Xibo_media_name']
# Xibo client details
Xibo_client_id = config['DEFAULT']['Xibo_client_id']
Xibo_sercet = config['DEFAULT']['Xibo_secret']


# Looks for file timestamp, used for comparing
def file_timestamp(file_location):
    try:
        return (os.stat(file_location)).st_mtime
    except:
        sys.exit("Error: Can't get timestamp")


# Compares files and if they differ, copies file to local drive
def modification_compare(original_file, destination_file):
    print("Modification time: {}".format(file_timestamp(original_file)))
    print("Modification time: {}".format(file_timestamp(destination_file)))
    if file_timestamp(original_file) != file_timestamp(destination_file):
        try:
            shutil.copy2(original_file, destination_file)
            print(1)
            return 1
        except:
            print(0)
            return 0
    else:
        print("Files match, no files copied")


# Converts PPTX file to PDF
def ppt_to_pdf(input_file_name, output_file_name, format_name=32):
    powerpoint = comtypes.client.CreateObject("Powerpoint.Application")
    # powerpoint = win32com.client.DispatchEx("Powerpoint.Application")
    powerpoint.Visible = 1

    if output_file_name[-3:] != 'pdf':
```

```
70          output_file_name = output_file_name + ".pdf"
71      deck = powerpoint.Presentations.Open(input_file_name)
72      deck.SaveAs(output_file_name, format_name)
73      deck.Close()
74      powerpoint.Quit()
75
76
77  # API part
78  # Get access token
79  def get_access_token(xibo_location):
80      """
81      Request an access token from the Xibo API
82      Uses Client ID and Client Secret to Authenticate
83      Return: Access Token as a String
84      """
85      # URL to get access token from. Xibo Server URL + /api/authorize/access_token
86      auth_url = "{}/api/authorize/access_token".format(xibo_location)
87      print(auth_url)
88      # The Client Credentials from a Xibo Application you create at
        Administration>Applications>Add
89      client_id = config['DEFAULT']['Xibo_client_id']
90      client_secret = config['DEFAULT']['Xibo_secret']
91      # How we plan to authenticate
92      auth_data = {'grant_type': 'client_credentials'}
93      # Send a Post request to xibo
94      auth_response = requests.request("POST", auth_url, data=auth_data, verify=False,
        allow_redirects=False, auth=(client_id, client_secret))
95      # Load the json to a list
96      response_list = json.loads(auth_response.text)
97      return response_list['access_token']
98
99
100 # Script for loading files to Xibo
101 def send_media(new_media_filename, old_media_id, xibo_location):
102     """
103     Send data to Xibo using API
104     """
105     # http:// XiboURL /api/ + GET/POST listed in the API
106     url = "{}/api/library".format(xibo_location)
107     # Authenticate using access token
108     headers = {'Authorization': "Bearer " + get_access_token(xibo_location)}
109     # Specify the path to the file that is being uploaded
110     basename = os.path.basename(new_media_filename)
111     mimetype = mimetypes.guess_type(basename)
112
113     encoder = MultipartEncoder({
114         'files': (new_media_filename, open(new_media_filename, 'rb'), mimetype),
115         'name': config['DEFAULT']['Xibo_media_name'],
116         # replacemisel on need väärtused olulised
117         'oldMediaId': "{}".format(old_media_id),
118         'updateInLayouts': '1',
119         'deleteOldRevisions': '1'
120     })
121
122     headers.update({"Content-Type": encoder.content_type})
123
124     response = requests.post(url, headers=headers, data=encoder)
125     print(response.text)
126
127
128 # Script for searching for a file
129 def search_media(search_for, xibo_location):
130     """
131     Search media from Xibo using API
132     """
133     #http:// XiboURL /api/ + GET/POST listed in the API
134     url = "{}/api/library?media={}".format(xibo_location, search_for)
135     # Authenticate using access token
136     headers = {'Authorization': "Bearer " + get_access_token(xibo_location)}
137     # Specify the path to the file that is being uploaded
138
139     response = requests.get(url, headers=headers).json()
```

```
    try:
        media_id = (response[0])["mediaId"]
    except:
        media_id = 0
    return media_id


if modification_compare(orig_PPTX_file_path, PPTX_file_path) == 1:
    print("Starting to copy file")
    ppt_to_pdf(PPTX_file_path, PDF_file_path)
    print("PPTX converted to PDF")
    send_media(PDF_file_path, search_media(Xibo_media_name, Xibo_address),
    Xibo_address)
    print("File uploaded")
else:
    print("Files are identical, not copied")

#ppt_to_pdf(PPTX_file_path, PDF_file_path)

# Looks for ID (testing purposes)
fetched_data = search_media(Xibo_media_name, Xibo_address)
print(fetched_data)

# send_media('INFOTV.pdf', search_media('InfoTV PDF'))

```