S3 Backup and Restore Program (Windows)

Introduction

This program is written in **Python 3.8** and uses the library Boto3 to communicate with AWS S3 buckets to act as a simple file backup and restore program while maintaining the appropriate folder structure. The program is **built and tested on Windows 10.**

Setup

Credentials:

The program looks for an AWS configuration file at **`~/.aws/credentials.`** These credentials can be set by installing AWS CLI Version 1 and creating a user account on AWS IAM that has full read/write access to buckets.

1. Install AWS CLI V1 from https://s3.amazonaws.com/aws-cli/AWSCLI64PY3.msi.
2. Run `aws configure` in a command prompt.
3. Enter the required details from the created account as per the prompts: **Access key ID**, **Secret Access Key**, **region** (`us-west-2`), and **output format** (`json`).

This will set up the AWS credentials required by the backup and restore program.

Reference: https://docs.aws.amazon.com/cli/latest/userguide/cli-configure-files.html

Python and Dependencies:

Python 3.8 needs to be installed from https://www.python.org/downloads/. The program needs the python path set properly to work, which is an option when running the installer.

Running the **MakeFile.bat** will automatically install dependencies and build the exe files from Python source files.

If the included MakeFile.bat does not work and causes an error such as "pip is not recognized as an internal or external command", it is possible the path for Python was not set correctly.
https://stackoverflow.com/questions/23708898/pip-is-not-recognized-as-an-internal-or-external-command

The MakeFile will automatically install the following dependencies using pip:

- Boto3
- Pyinstaller (to create the exe)

It is preferred to follow the "Usage with code" section instead of the exe section, as the exe output contains a lot of unnecessary outputs**.**

Usage with Code (.py)

Backup:

After ensuring python is installed, the program can be run with these parameters:

- `python backup.py "c:\directory\to\back\up"`

The directory must be a valid Windows file path to a folder that needs to be backed up to an S3 bucket. **A valid directory must not include the \ at the end.** Please do not try any monumentally large directories, such as "`C:`" or any write protected directories that the program does not have access to write to**.**

Restore:

The program can be run with these parameters:

- `python restore.py "C:\directory\to\restore\to" "buckettorestore"`

The first parameter is a valid Windows file path to a folder that will contain the downloaded S3 files. The second parameter is the bucket to access and restore to the directory in the first parameter. The bucket must be a valid bucket which the AWS credentials are able to access.

Usage with executables (.exe)

Backup:

After compiling the executables using the MakeFile.bat, the program can be run with these parameters:

- `backup "c:\directory\to\back\up"`

The directory must be a valid Windows file path to a folder that needs to be backed up to an S3 bucket.

Restore:

The program can be run with these parameters:

- `restore "C:\directory\to\restore\to" "buckettorestore"`

The first parameter is a valid Windows file path to a folder that will contain the downloaded S3 files. The second parameter is the bucket to access and restore to the directory in the first parameter. The bucket must be a valid bucket which the AWS credentials are able to access.

Program Features/Structure

Backup:

The program will use os.walk to traverse a directory to upload all files and folders in the passed in directory while respecting the file and folder structure of the directory being backed up. The subfolders and files will all be uploaded onto a newly created bucket.

The bucket name is "css436pranavsbucketXXXX" where the last X's are a series of randomly generated numbers. This ensures that there is a very low chance of a duplicate bucket being created, but it may not always be the case.

The program will begin to upload files and folders into the bucket. When it reaches a file that already exists on the bucket, it will check if the local file's last modified timestamp is greater than the one existing on S3. If this is the case, the file on S3 will be overwritten with the newer local file. If there are no conflicts, the files will be uploaded with no issues.

Restore:

The program will traverse through the structure of the S3 and download the entire specified bucket on to a specified folder locally, while maintaining the file/folder structure of S3. The program starts to download all files and folders on S3. If it tries to override a file with the same name in the same directory, it will check if the local file is newer. If so, the program will offer a Y/N prompt to either overwrite or skip the file.

Note: The wording on the requirements document was slightly confusing to me, so I went with the understanding that the directory specified for restore was the local directory to copy the files of the bucket to.