Projeto1SegApli

TP21-G04:

- Santiago Benites fc54392
- Inês Morais fc54462
- Mirguel Carvalho fc54399

Project done with python3.10

Instalation

Python

In order to install our program we will require the python programming language, taking this into account we recommend at least python version 3.10 which was the one we were using

First run `python -V` this will check if you already have python installed If you don't just follow one of the tutorials online, but take into account we will require python with version 3.10 or above.

Moreover we recommend the usage of a package manager like conda or mamba, if any are installed it can simplify the process.

But it is not required in our tutorial, if you do have a package manager installed you can just follow the tutorial we will give little warning wherever your path might deviate from the normal installation.

Take into account all our scripts were done with python you system may need python3, if so just change the scripts to this effect.

Required packages

The installation will require some packages, these can be installed with the pip package manager using install.sh in setup directory

In another note if you prefer it can be simpler to just run the pip install commands from you current python installation

The commands would be

```
pip install pycryptodome
pip install cryptography
```

If you are using an alternative package manager the best help I can give is to try to replace pip with your package manager, be it mamba/conda/other...

Directory Structure

The required directories for the program to run will be :

```
directories =
["src/Bank","src/Bank/auth","src/MBec","src/MBec/creditCard","src/MBec/user
sFiles","src/Store"]
```

If these do not exist the program wont work, we made the script setup/install.py to make sure that these exist, if they don't the script will create them. You must run this script and all other scripts from the root of the project, the directory that has src, setup and this README.

Running the program

You probably received a specific file structure, please don't disturb it, as it is required from normal execution. In order to run either MBec or the Store the bank.auth file is required to be present in the root directory of both of these (take into account that bank.auth is just the standard name of the file, it can change if you want to).

This means that bank.auth needs to be present in both src/MBec and src/Store so that the store and the MBec can run, if this is not true they will not start and both the program will instantly exit.

To run programs the required commands are, you must run all program from the root of the project.

```
python3 src/Bank/Bank.py [-p <bk-port>] [-s <auth-file>]

python3 src/MBec/MBec.py

python3 src/Store/Store.py [-p <st-port>] [-s <auth-file>]
```

When the required programs are run, MBec will now be able to take in input from the stdin. The input should be in the forms present in the project description

```
mbec [-s <auth-file>] [-i <ip-bank-address>] [-p <bk-port>] [-u <user-
file>] -a <account> -n <balance>
mbec [-s <auth-file>] [-i <ip-bank-address>] [-p <bk-port>] [-u <user-
file>] -a <account> -d <amount>
mbec [-s <auth-file>] [-i <ip-bank-address>] [-p <bk-port>] [-u <user-
file>] -a <account> -c <amount>
mbec [-s <auth-file>] [-i <ip-bank-address>] [-p <bk-port>] [-u <user-
file>] -a <account> -g
mbec [-i <ip-store-address>] [-p <st-port>] [-v <virtual-credit-card-file>]
-m <shopping-value>
```

In these declarations, the order of the input doesn't matter, nor does if the input has spaces between them.

Therefore the following commands do the same thing

```
python3 src/MBec/MBec.py (run the program with enter)
-a5000-n123.00

python3 src/MBec/MBec.py (run the program with enter)
-a 5000 -n 123.00
```

Take into account that all values should follow the project's directives, therefore a monetary value should be presented with its decimals, separated by a dot, if 123.00 were to be exchanged with 123, the program would not accept it.

Command will be taken from the stdin, and cannot exceed 4096 characters. Take into account no state is passed between executions of the MBec, the while loop is just done for ease of access. So that you don't have to recall the program from the command line, we don't recommend taking it out because it requires some small changes, but if you really want to do it we can provide some help.

Final considerations

In line 74 of bank you will find commented a script run that will delete all temporary files from the project, we will leave it commented in case you don´t want to use it.

```
#call(["python", "src/clearUserFiles.py"])
```

Moreover in line 142 to 146 of Cripto.py you will find commented

```
#with open(f"{os.getcwd()}/src/MBec/bank.auth","wb") as f:
#    f.write(cert.public_bytes(serialization.Encoding.PEM))

#with open(f"{os.getcwd()}/src/Store/bank.auth","wb") as f:
#    f.write(cert.public_bytes(serialization.Encoding.PEM))
```

Which are 4 lines which instantly copy the bank.auth file from the client to the server without the need to copy it manually. Just like the case above we left commented in case you don´t want to use it.

When testing we found ourselves having problem when using the classes VM to run our project specifically the sockets would timeout whenever we tried to make communications, to solve this problem we increased the number of cores in VM.

This is done in:

VM -> Settings -> System -> Processor

we were able to make it work with 4 cores

Moreover we tested this program in both windows and linux therefore we believe we got it covered on this front. We tested in Windows11 and Debian Based distribution, so these are the systems we recommend you operate on.