RIT | Golisano College of Computing and Information Sciences
School of Information

NSSA220 Task Automation

Lab 5 – Automate Adding User Accounts

The Basics:

Much of the time, a system administrator will write scripts to automate repetitive tasks. For example, adding users to a system often requires a script to make the job easier and more efficient. Although considerable time is spent writing the script, it will inevitably translate to less work later on when adding additional users to the system in the future. This assignment aims to write a script that will automate adding local Linux user accounts to the CentOS 8 virtual machine in RLES.

Because the file can come from any department in the organization, the formatting is unreliable and needs to be considered when writing the code. However, the data does not need to be fixed. That would require some guesswork. Moreover, you can only work with the information given. However, make sure that it does not prevent the script from running or prevent other users in the file from being added. **Do not alter or modify the file in any way**. *The file provided* is the file the grader will use to run the script.

Script Requirements:

The script must be written in Python 3 and titled "*add_user.py*." It will be run and tested on the CentOS 8 virtual machine in RLES. The instructor or teaching assistant will run it and award points based on the requirements and overall functionality. Points awarded will be based on the criteria in the *"Script Grading Rubric."*

Additional Information:

1. The CSV file contains a header line defining the fields; your script must detect the header line and **not** process it as if it were a user addition.
2. The users should have a default group as indicated in the File. Again, **do not edit, or alter the file**, you must use the data as is.
3. You must be able to create unique user names, which should be constructed as first initial followed by the last name (e.g., John Smith would be assigned the user name '*jsmith*').
4. What if the file has three employees named John Smith? Or "Jason Smith?" To handle duplicate names, append a number, to the account name. For example, jsmith, jsmith1, jsmith2, and so forth.
5. The script must handle embedded special characters, for example what if the persons last name is "O'Donnell?"
6. The script must handle missing or incorrect information in the fields (what happens if you are given a file with a field that is not populated or has incorrect data? The script must handle this possibility. You **DO NOT** need to correct it; you just need to deal with it so that it doesn't cause the script to crash and report it to the user. For example, *"User account was not added due to missing information."*
7. The user's home directory should be located in /home/*department*, where department is the user's department, e.g., for Natasha Richardson in the CEO department, the correct home directory would be /home/**ceo**/nrichardson in lowercase.
8. Any member assigned to the "office" group must use "csh" (C shell) as their default shell, everyone else is assigned the Borne Again Shell, or Bash as the default.
9. The script must detect incorrect data (e.g., what if the user's full name in the CSV file is '555-1212'? i.e., what if the person doing data entry made mistakes?). Again, it is not up to you to correct it, but the script should indicate that the record is invalid.

10. Set the default password to "***password***" for each new user.
11. Expire the password so the first time the user logs in they must change it. For testing purposes use "***1$4pizz@***" (quote are not part of the password) as the password to change to.
12. Each record in the CSV file indicates the default group for the corresponding user. Your script should be able to handle the need for a new group, in other words, create the group if it does not exist.

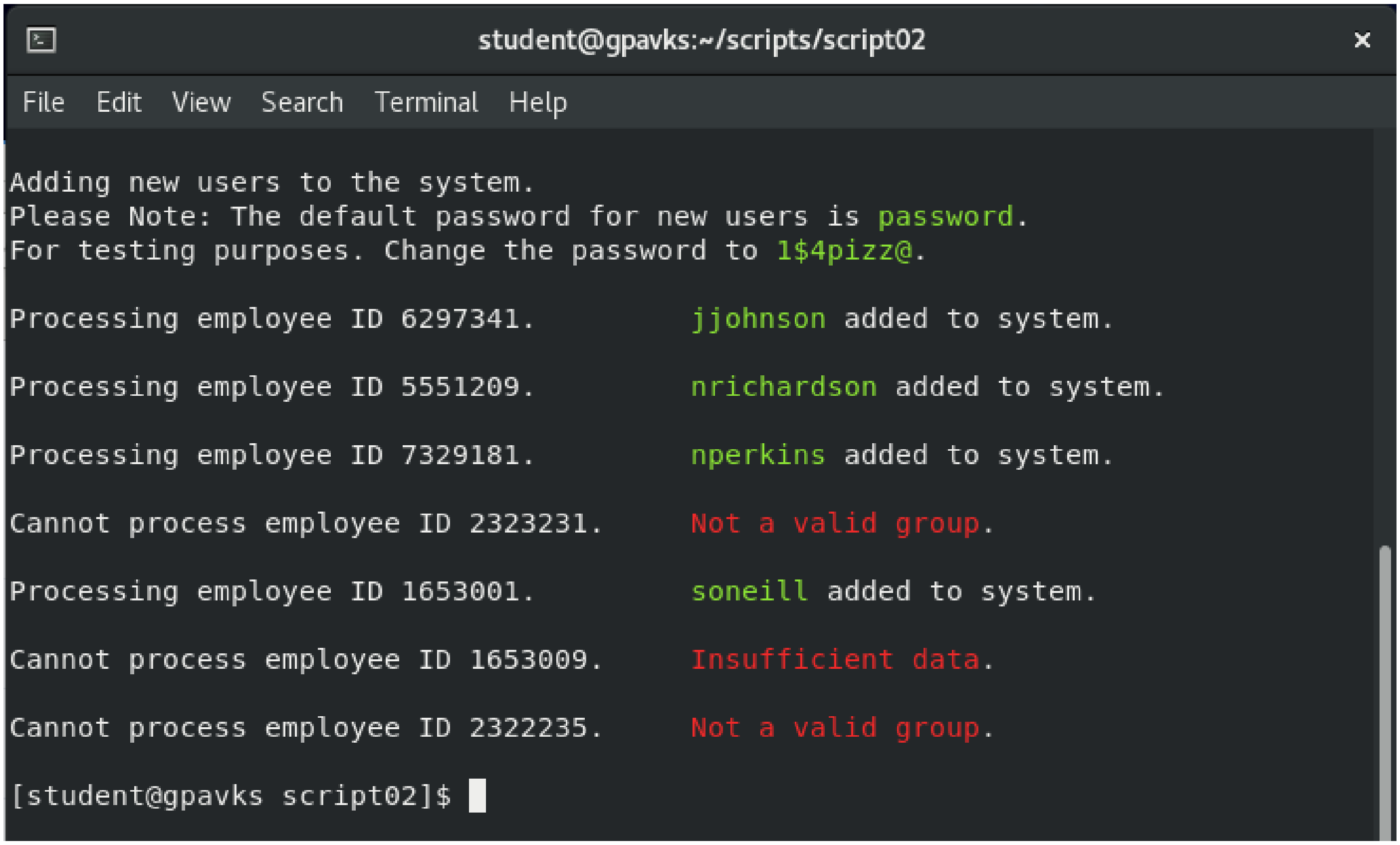
In addition to the *linux_users.csv* file you will also find a Python script titled "*remove_users.py*." You may use this script to remove the users and to start with default users on the CentOS 8 virtual machine.

Script Demo:

To see a demo of the script, download the zip file (script_demo_files) from myCourses to the CentOS 8 virtual machine. Extract the files to the /tmp directory. Run the following command in /tmp to see the script in action.

```
scriptreplay --timing=time.log add_user.scr
```

Figure 1 – Sample Script Output



```
Adding new users to the system.
Please Note: The default password for new users is password.
For testing purposes. Change the password to 1$4pizz@.

Processing employee ID 6297341.        jjohnson added to system.

Processing employee ID 5551209.        nrichardson added to system.

Processing employee ID 7329181.        nperkins added to system.

Cannot process employee ID 2323231.    Not a valid group.

Processing employee ID 1653001.        soneill added to system.

Cannot process employee ID 1653009.    Insufficient data.

Cannot process employee ID 2322235.    Not a valid group.

[student@gpavks script02]$ 
```

RIT | Golisano College of Computing and Information Sciences
School of Information

Script Grading Rubric

Requirements	**Points**	**Points Earned**
Script contains the shebang!	2	
Script has executable permissions set.	2	
Script is commented with student's name, date.	2	
Script is titled "*add_user.py*"	2	
Script clears the terminal when it runs.	2	
The script checks to see if the group in the file exists and if not creates it.	5	
The script can read in and process the csv file as is.	5	
The script ignores the header line in the csv file.	5	
The script properly formats the user ID.	5	
The script can handle duplicate user IDs.	5	
The script appends a number to a duplicate user ID and increments the number for multiple occurrences beginning with number 1.	5	
Each user is assigned the correct group membership.	5	
Each user is assigned their home directory based on organization they belong to.	5	
Each user is assigned a default shell.	5	
Each user is assigned a default password.	5	
Password is expired and the user must change it the first time they log in.	5	
The script communicates when bad records exist.	5	
Script is sufficiently commented.	5	
Script is written in Pythonic style.	5	
Script runs with no errors.	10	
Script is fully functional and runs as expected.	10	
	Final Grade	