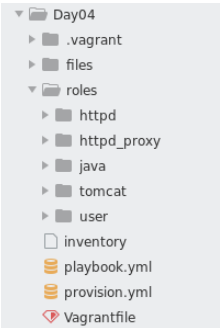
Lab Work Task. Day 4.
Making playbooks and output better.

Task

A) On Control Machine:

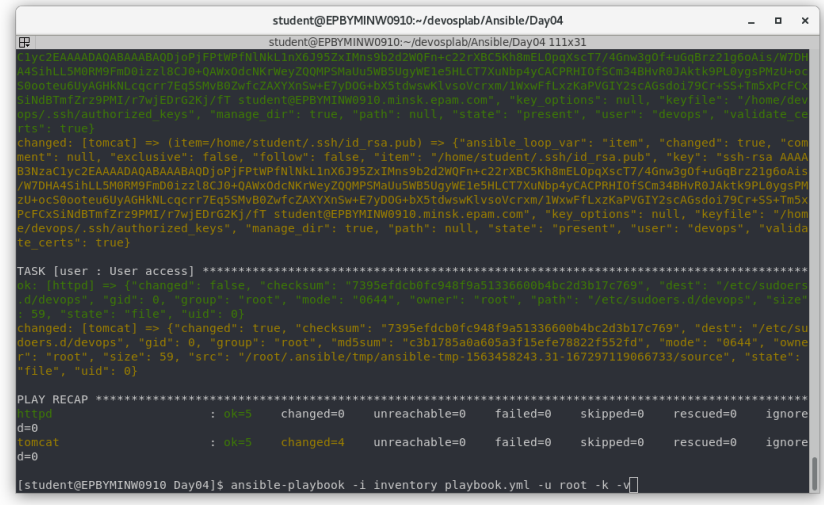
Create folder ~/ansible/day4. All working files should be placed there (Vagrantfile, playbooks, roles, callback plugins, etc)



Picture 1

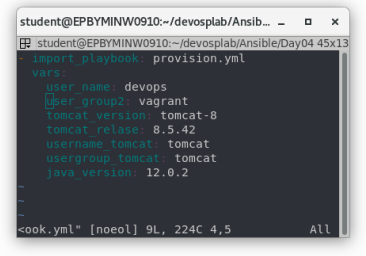
B) Refactor playbook from previous homeworks to use import_playbook statements: split playbook to plays, place each play into separate file, load each with import statement.

Picture 2.1 – Import playbook with variables



Picture 2.2 - Results

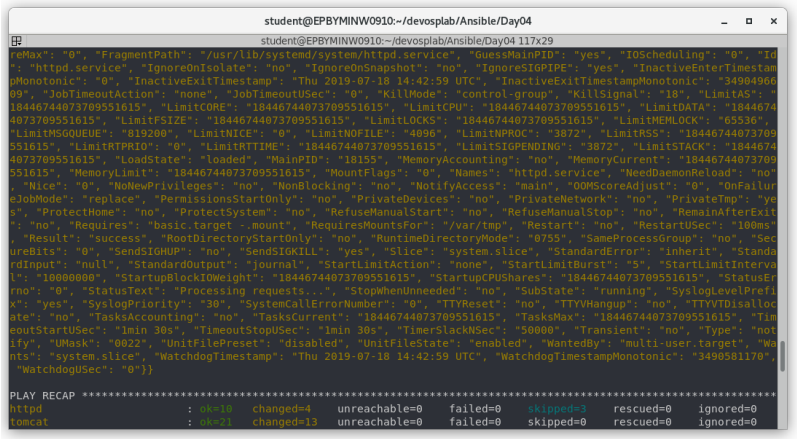
C) Refactor roles from previous homeworks to use import_tasks statements with necessary when clauses for making code cleaner and better understandable.



Picture 2.1 – playbook.yml

Picture 2.2 – provision.yml

Picture 2.2 – Project tree view



Picture 2.3 – Results

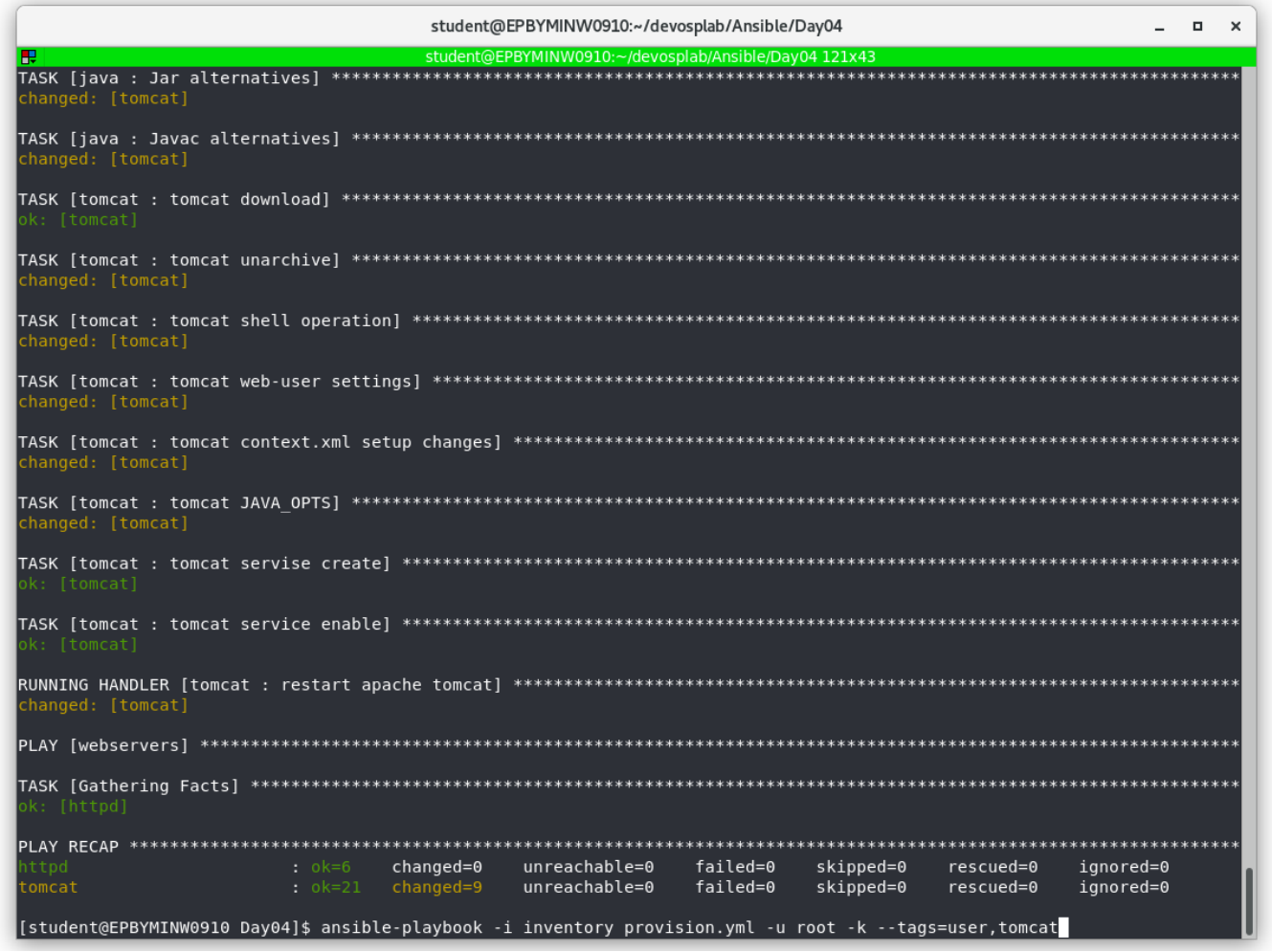
D) Equip tasks with tags (webserver, appserver, always, never (v2.5), sanity, etc). Run playbook with different tags.

```
playbook: playbook.yml

  play #1 (all): all    TAGS: [all,user,allways]
      TASK TAGS: [all, allways, user]

  play #2 (all): all    TAGS: [all,never,java]
      TASK TAGS: [all, java, never]

  play #3 (appservers): appservers      TAGS: [all,tomcat,allways]
      TASK TAGS: [all, allways, tomcat]

  play #4 (webservers): webservers      TAGS: [all]
      TASK TAGS: [all, allways, httpd, proxy]
[student@EPBYMINW0910 Day04]$
```

Picture 3.1 – ansible-playbook playbook.yml --list-tags

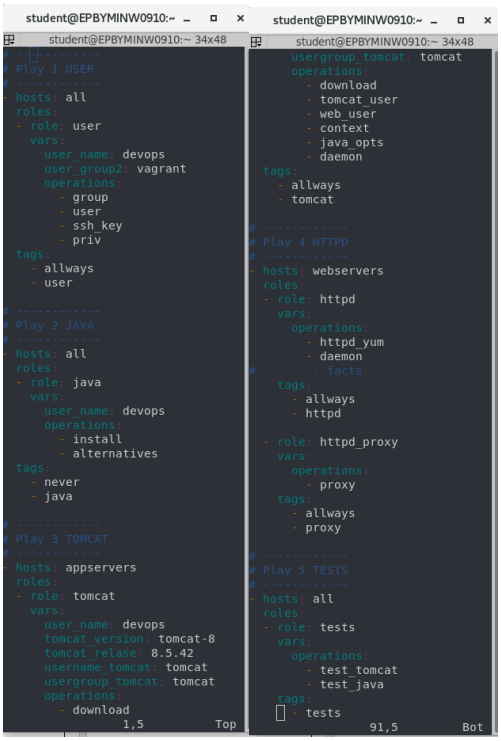
```
student@EPBYMINW0910:~/devosplab/Ansible/Day04
student@EPBYMINW0910:~/devosplab/Ansible/Day04 121x43
[student@EPBYMINW0910 Day04]$ ansible-playbook -i inventory provision.yml -u root -k --tags=user
SSH password:

PLAY [all] *********************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************
ok: [httpd]
ok: [tomcat]

TASK [user : User group create] ************************************************************************************
ok: [tomcat]
ok: [httpd]

TASK [user : User create] ******************************************************************************************
ok: [tomcat] => (item={u'home': True, u'name': u'devops', u'groups': u'devops, vagrant', u'uid': 1021})
ok: [httpd] => (item={u'home': True, u'name': u'devops', u'groups': u'devops, vagrant', u'uid': 1021})

TASK [user : User key] *********************************************************************************************
ok: [httpd] => (item=/home/student/.ssh/id_rsa.pub)
ok: [tomcat] => (item=/home/student/.ssh/id_rsa.pub)

TASK [user : User access] ******************************************************************************************
ok: [tomcat]
ok: [httpd]

PLAY [all] *********************************************************************************************************

PLAY [appservers] **************************************************************************************************

PLAY [webservers] **************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************
ok: [httpd]

PLAY RECAP *********************************************************************************************************
httpd                      : ok=6    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
tomcat                     : ok=5    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

[student@EPBYMINW0910 Day04]$
```

Picture 3.2 – Run with tag user

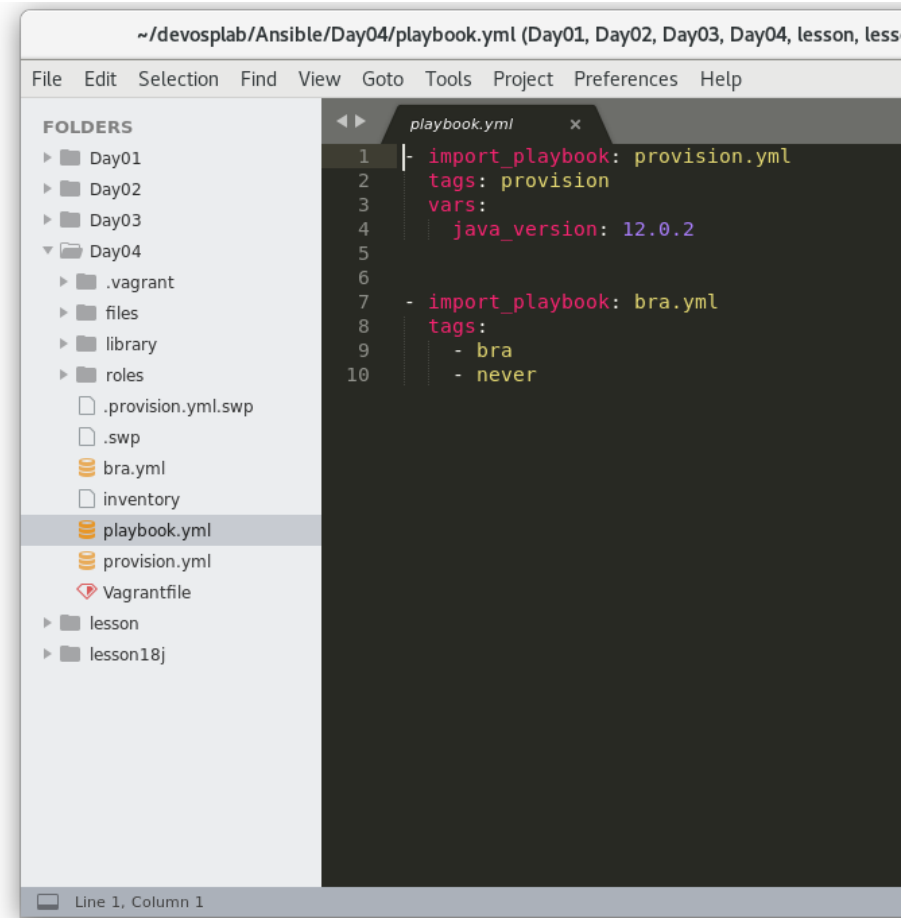


Picture 3.2 – Run with tags user and tomcat

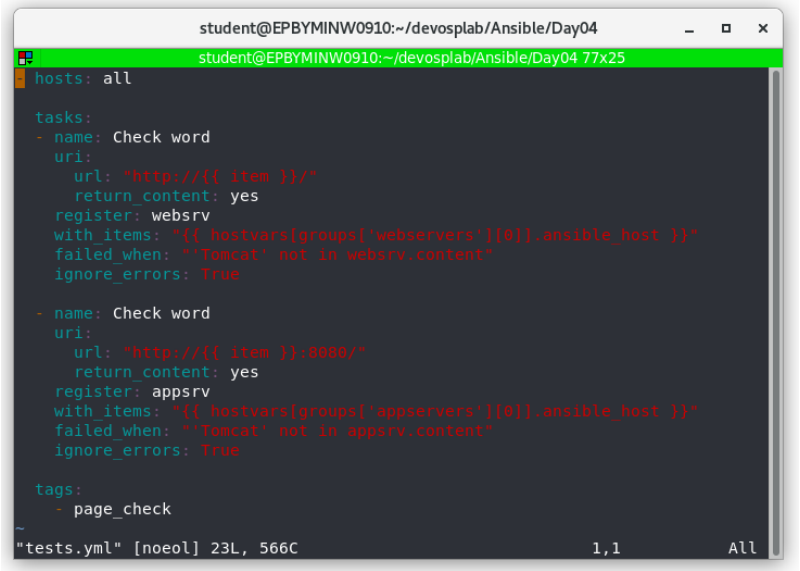
E) Refactor java and httpd roles from previous homeworks. Move out sanity checks to separate role. Simplify role usage (httpd tests, java tests, tomcat tests, etc). Pass necessary parameters. Call role with tag sanity. Test behavior.



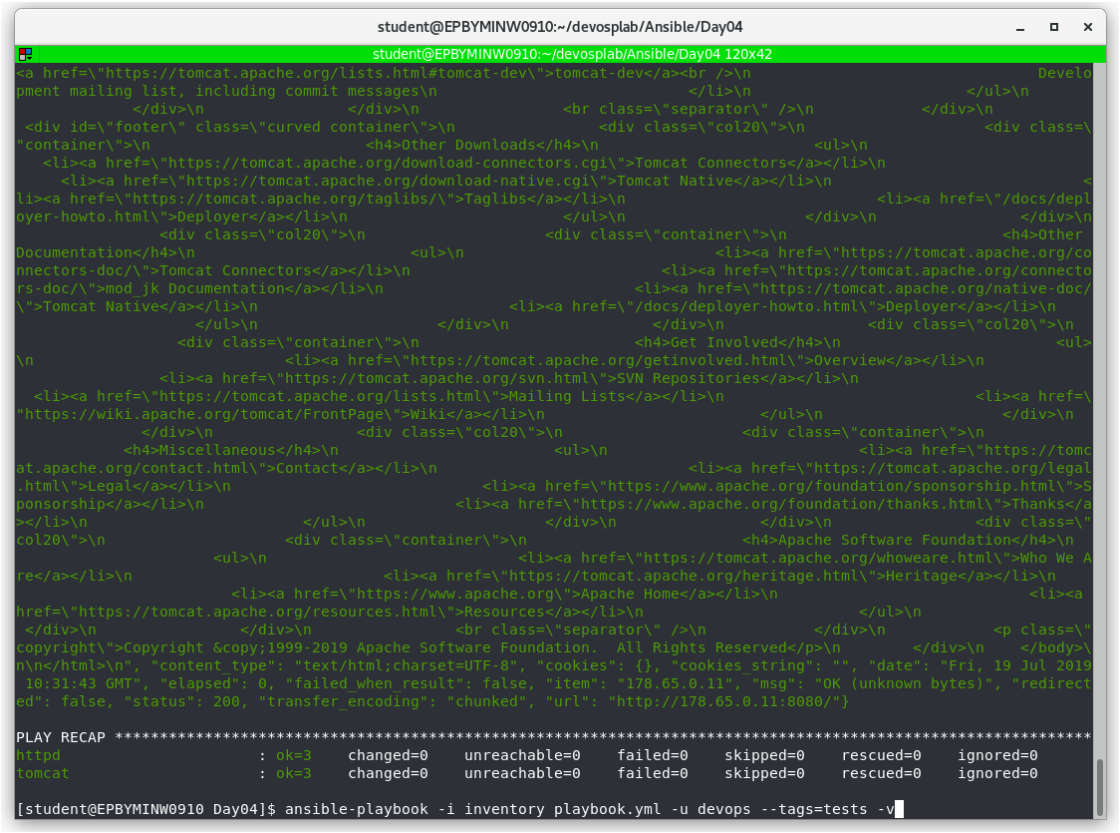
Picture 4.1 – provision.yml



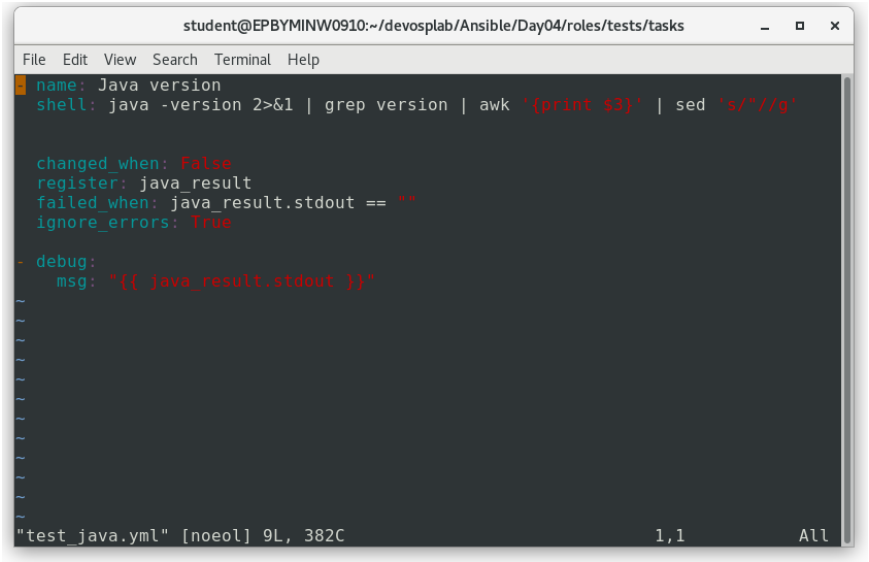
Picture 4.2 – playbook.yml



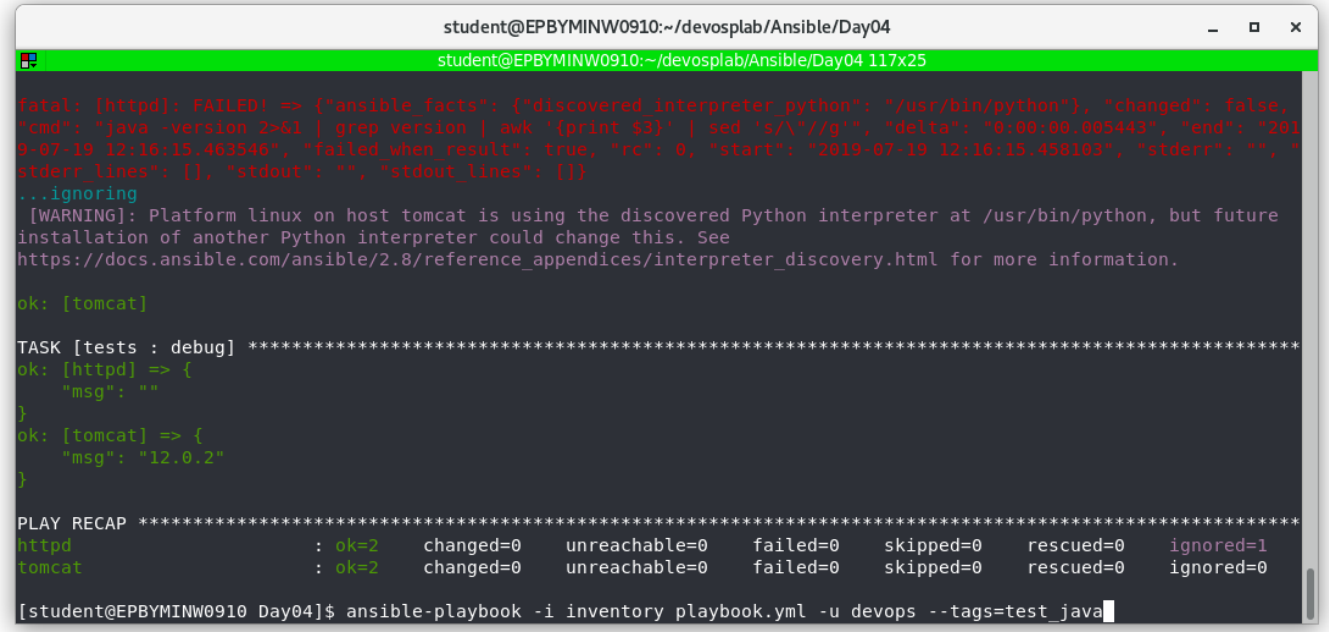
Picture 4.3 – Tomcat page test role



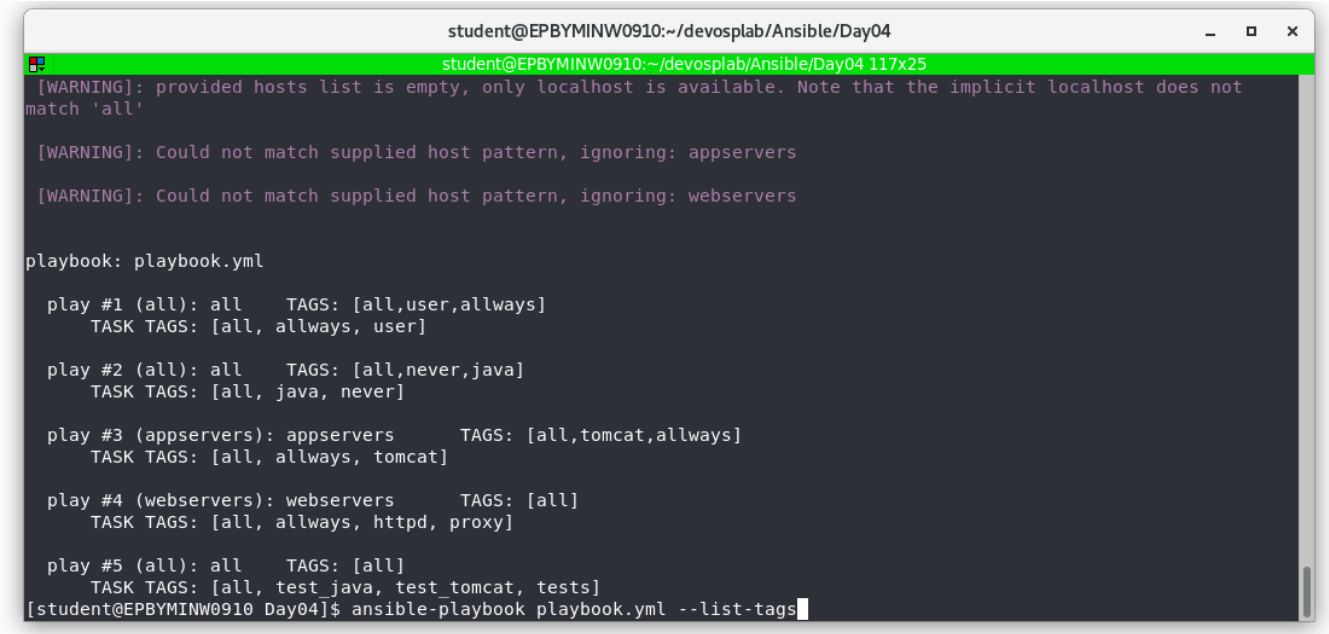
Picture 4.4 – Tomcat page test results



Picture 4.5 – Test java role

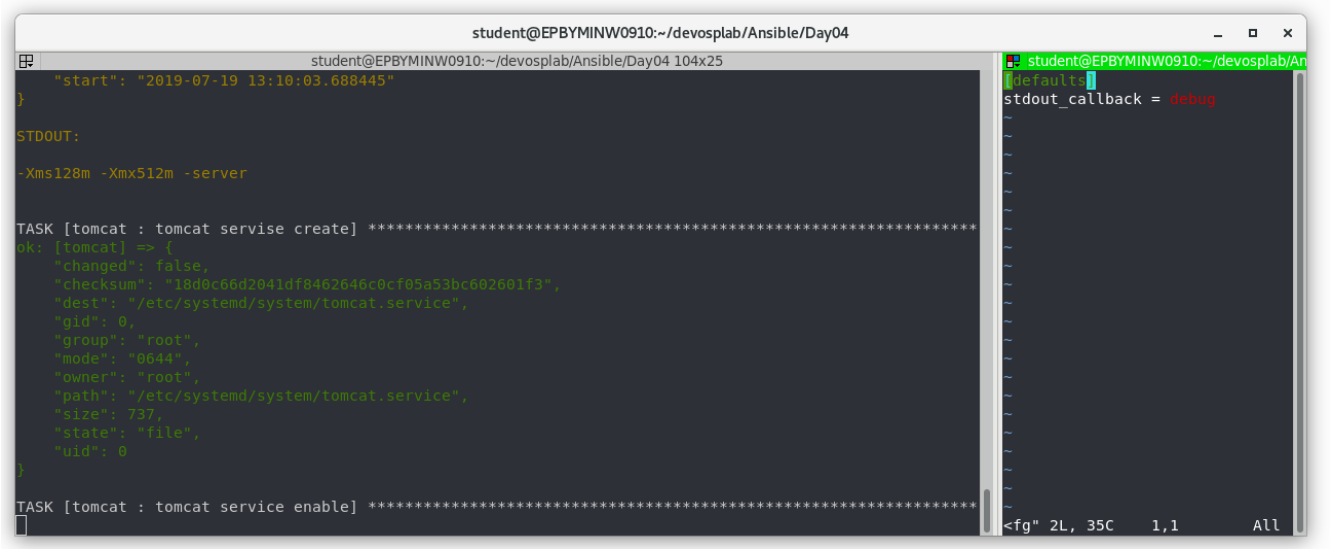


Picture 4.6 – Java test results

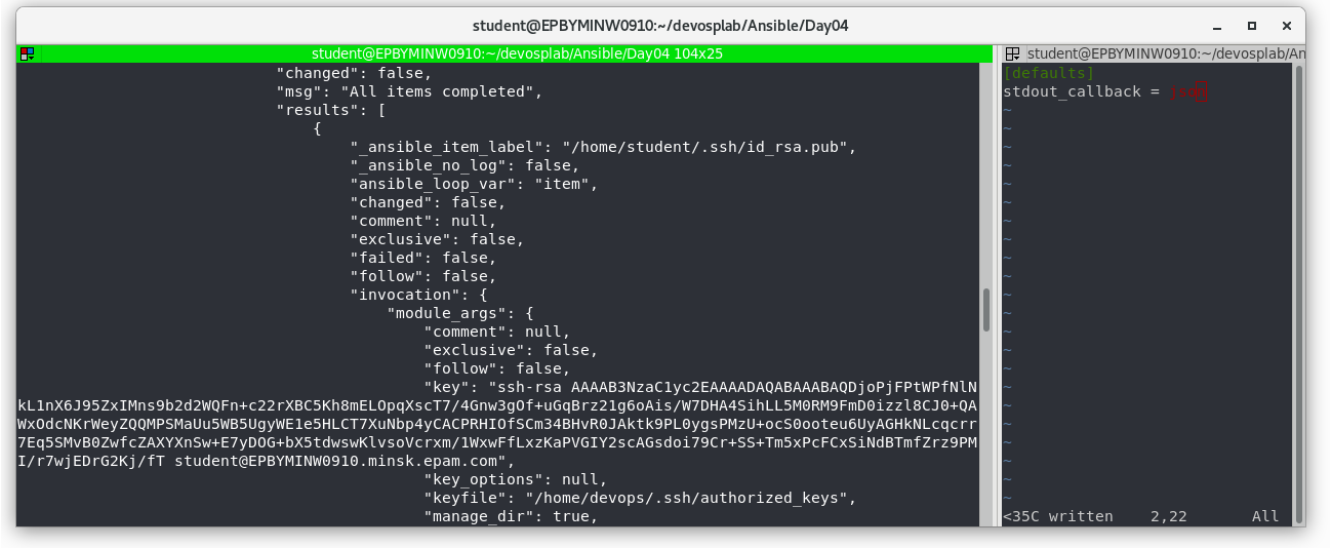


Picture 4.7 – Tags list

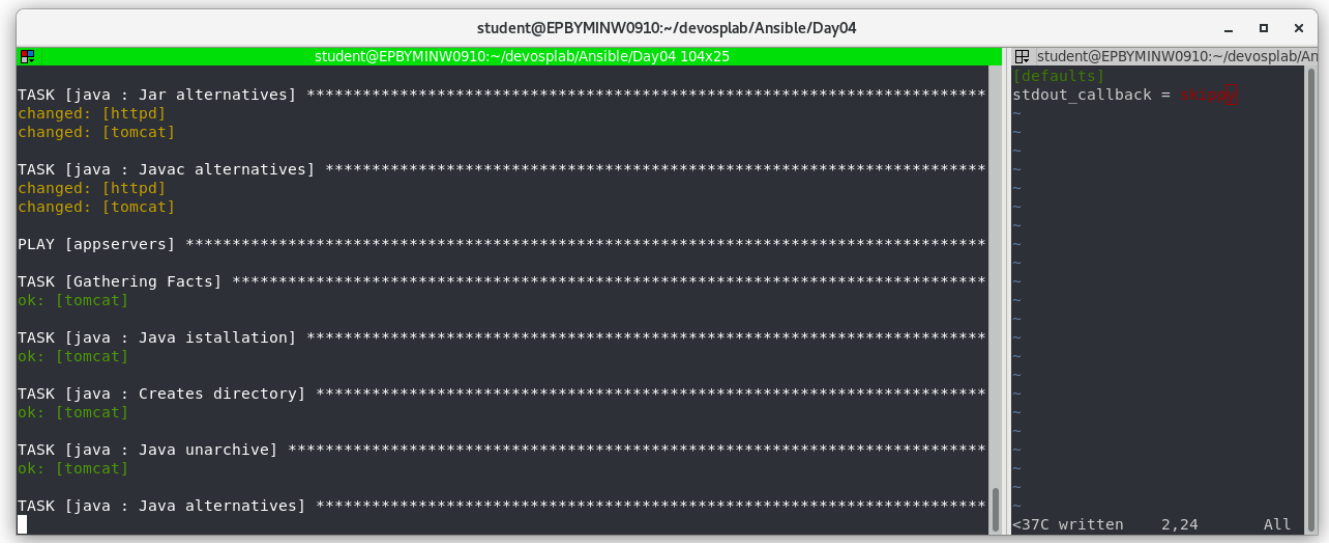
F) Use my custom call back plugin with your playbooks. Make necessary changes (if needed). Provide detailed information how you configured ansible to use this plugin.



Picture 5.1 – Call back plugin “Debug”



Picture 5.2 – Call back plugin “Json”



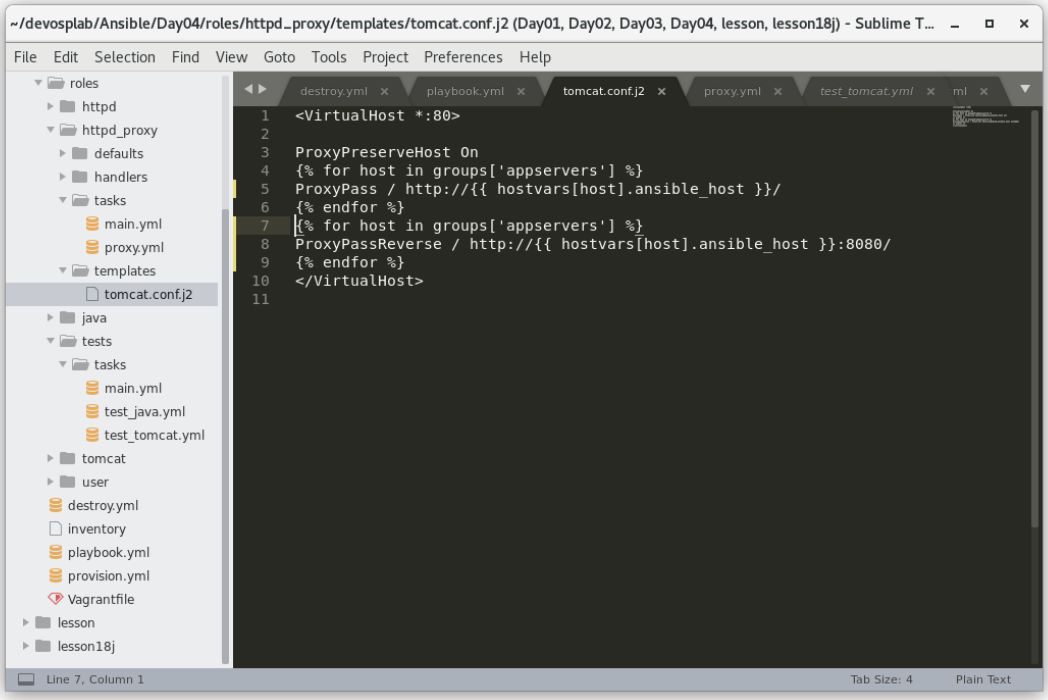
Picture 5.3 – Call back plugin “Skippy”

I am create ansible config file “/etc/ansible/ansible.cfg” and add into [deafault] stdout_callback with needed configuration.

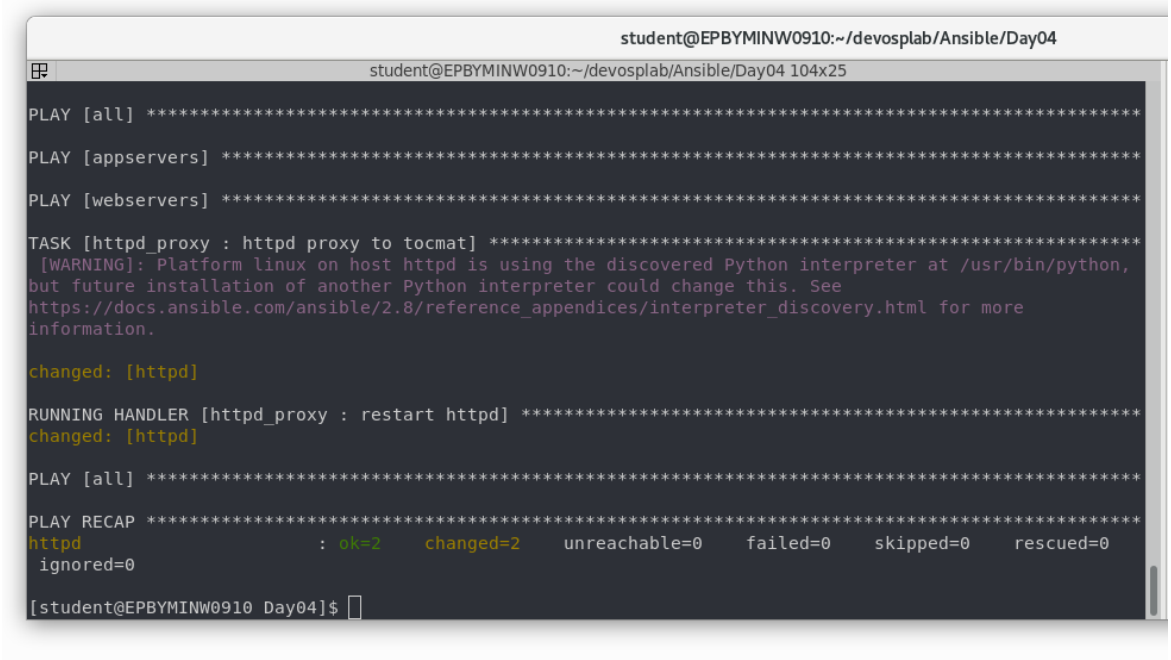
Also we can use export variable:

```
$ export ANSIBLE_STDOUT_CALLBACK=debug
```

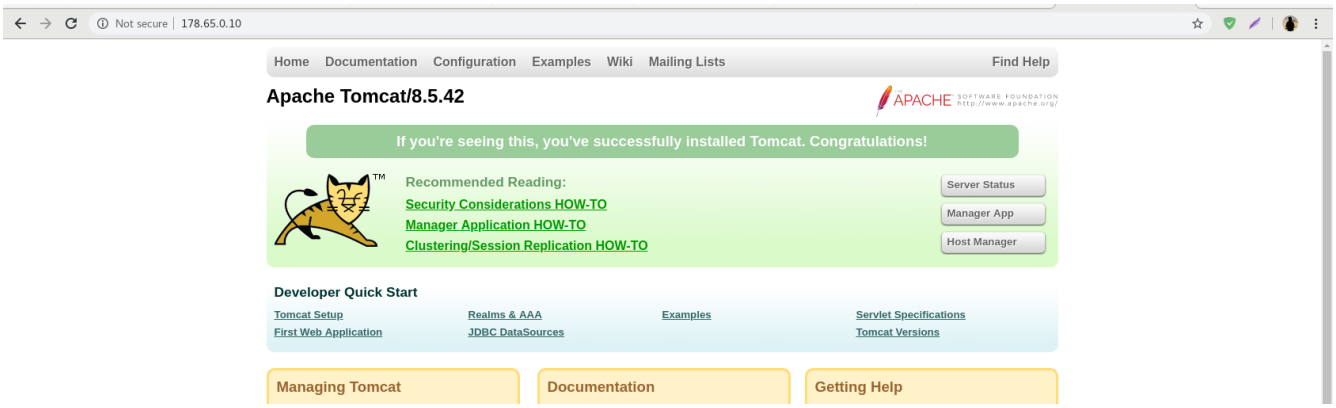
G) Create template file to produce httpd VirtualHosts configuration, develop simple playbook for testing needs, define a list of domains, specify different parameters for every domain.



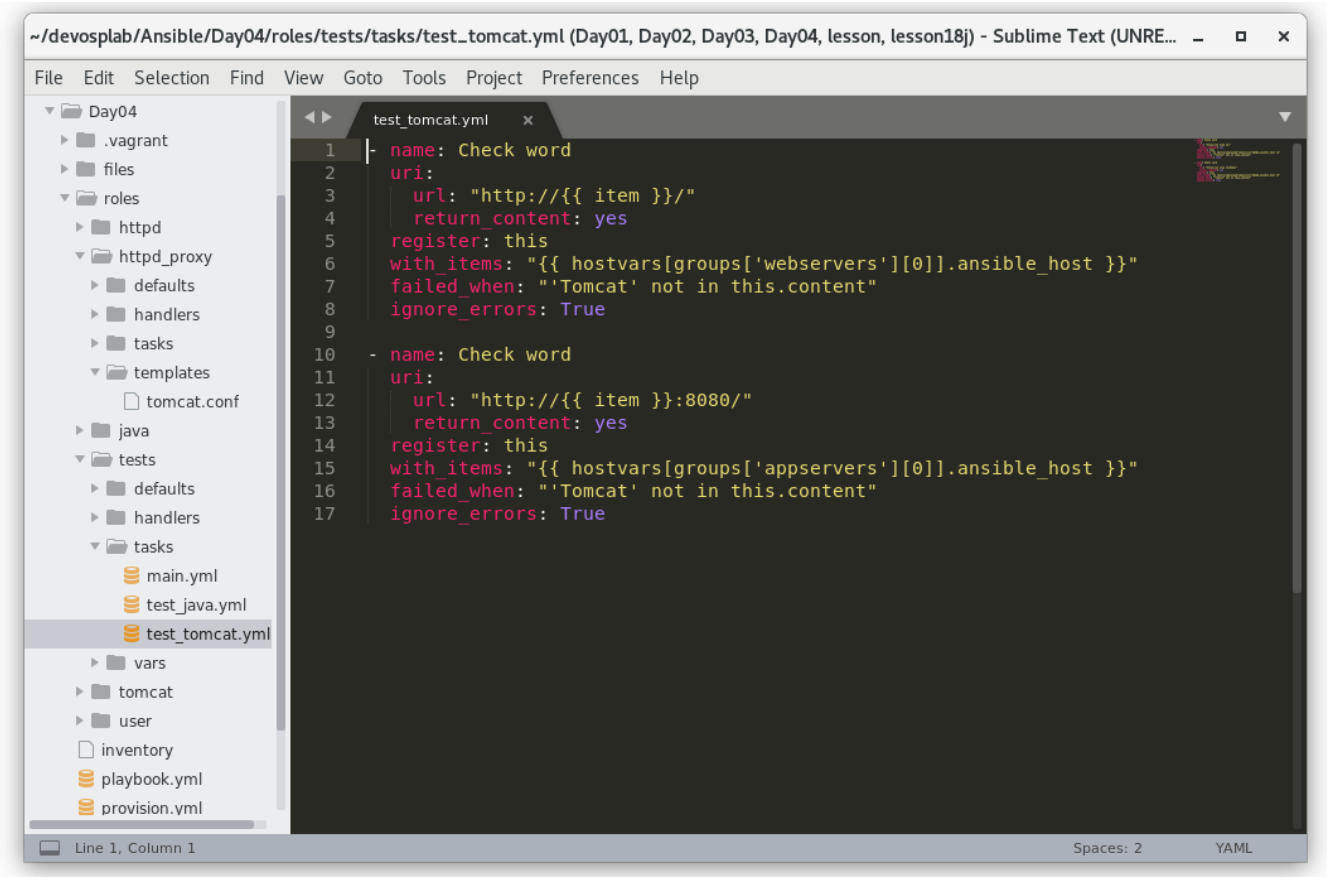
Picture 6.1 – Template httpd vhosts



Picture 6.2 – ansible-playbook -i inventory playbook.yml -u root -k –tags=proxy



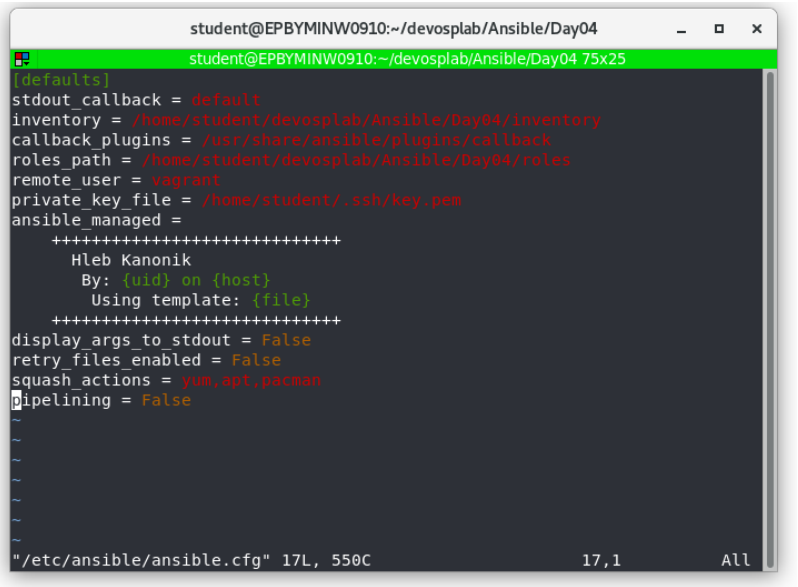
Picture 6.3 – Check results



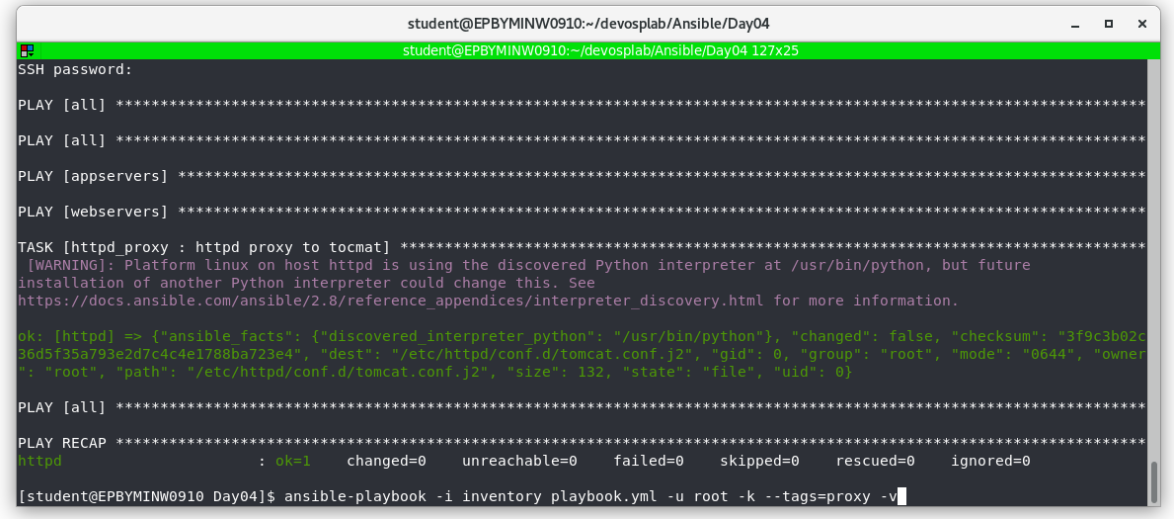
Picture 6.4 – Playbook for testing needs

H) Create Ansible Configuration file. Specify following settings (at least):

- Default inventory file to be used
- Callback plugins location and configuration
- Roles location
- Default remote user (vagrant)
- Default private ssh key
- Ansible managed comments string
- Enable displaying args to stdout
- Disable retry files creation
- Explore squash configuration
- Enable pipelining. Describe changes how you see them.



Picture 7.1 – Configuration file



Picture 7.2 – Provision after changing ansible.conf

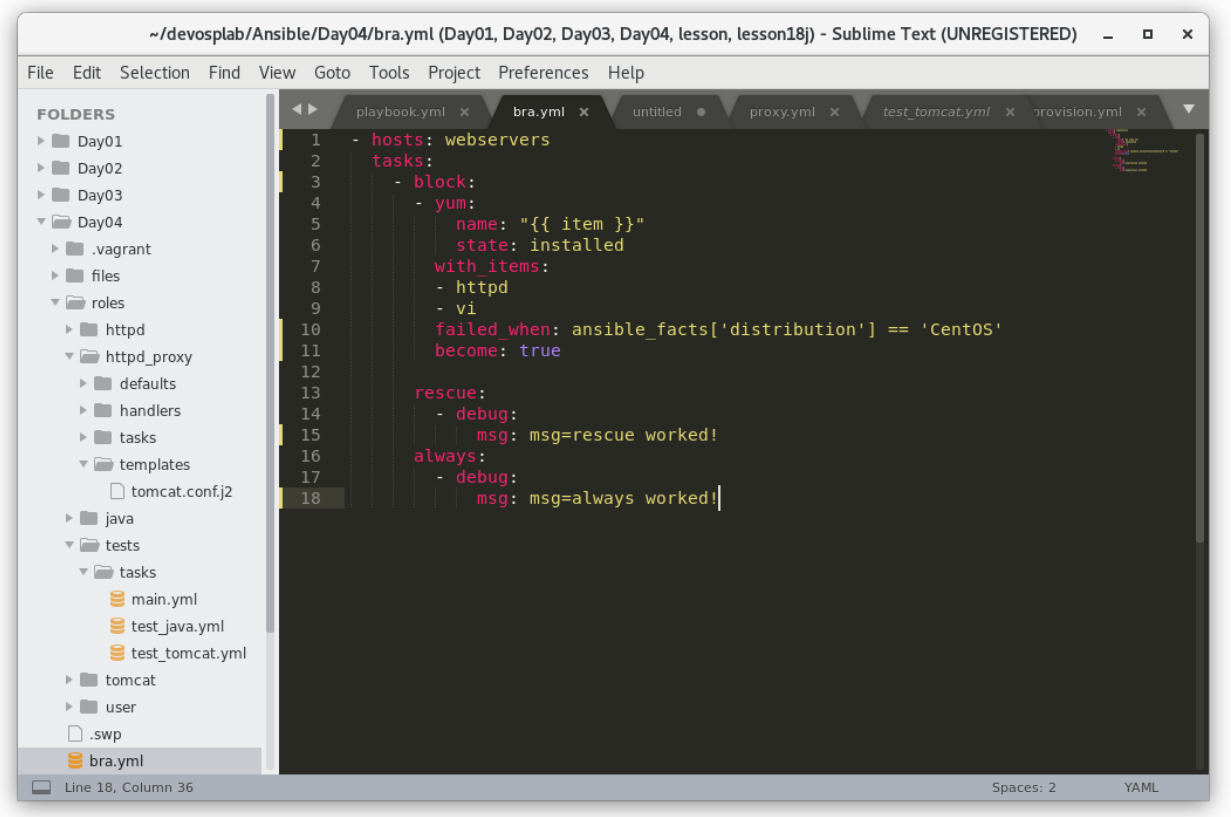
I) Develop simple playbook which uses Block/Rescue/Always flow. Describe behavior how you see this works.

The condition inside the “block” will be fulfilled when the condition in “when” is met. After an error appears in “block”, the execution goes to “rescue”. “Always” is always performed.

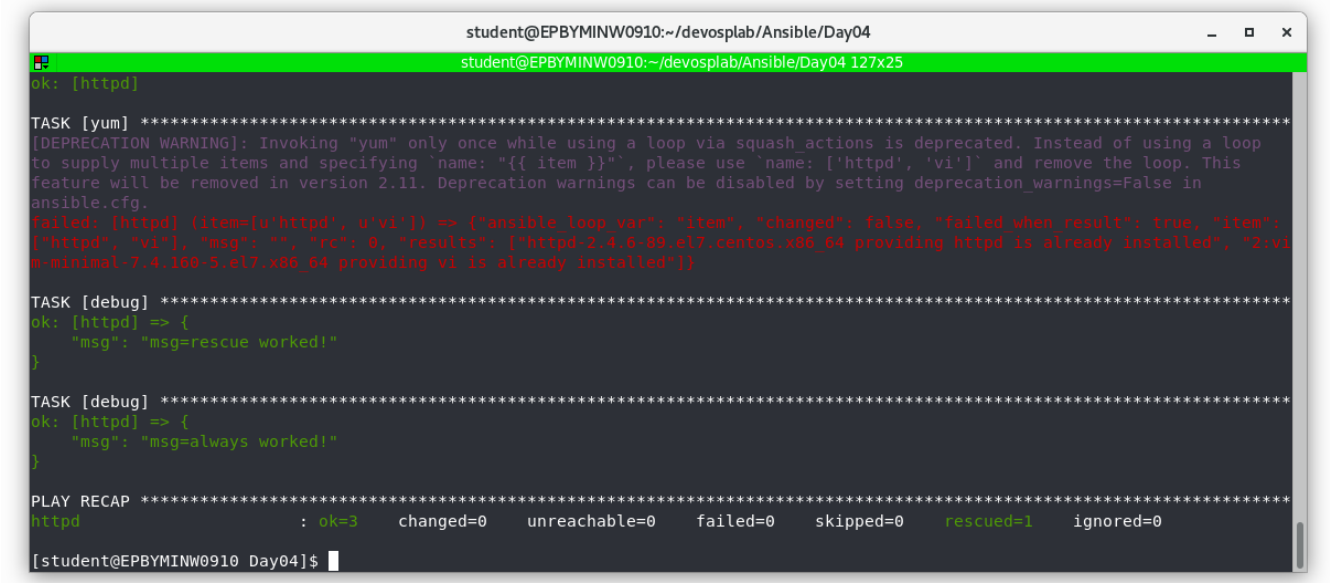
Using “block” you can intercept errors, for example:

Условие внутри “block” будет выполнено, когда будет соблюдено условие в “when”. После появления ошибки в “block” выполнение переходит в “rescue”. “Always” выполняется всегда.

С помощью “block” можно перехватывать ошибки, например:

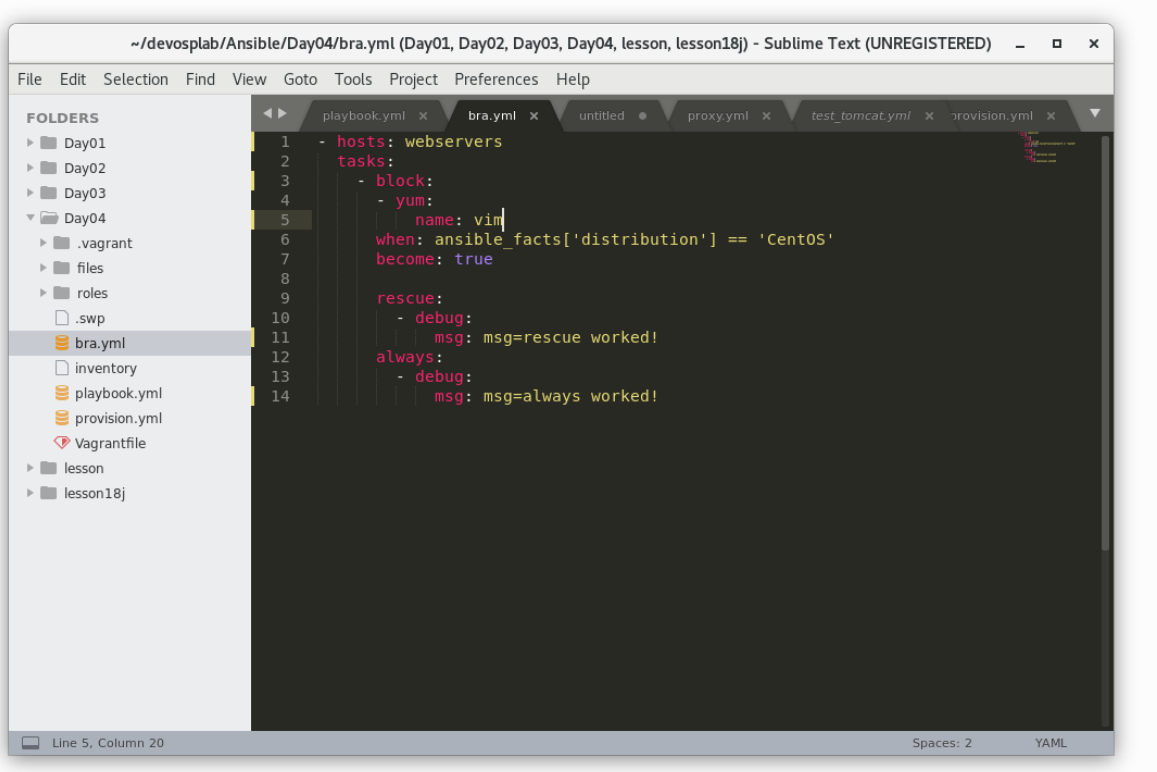


Picture 8.1

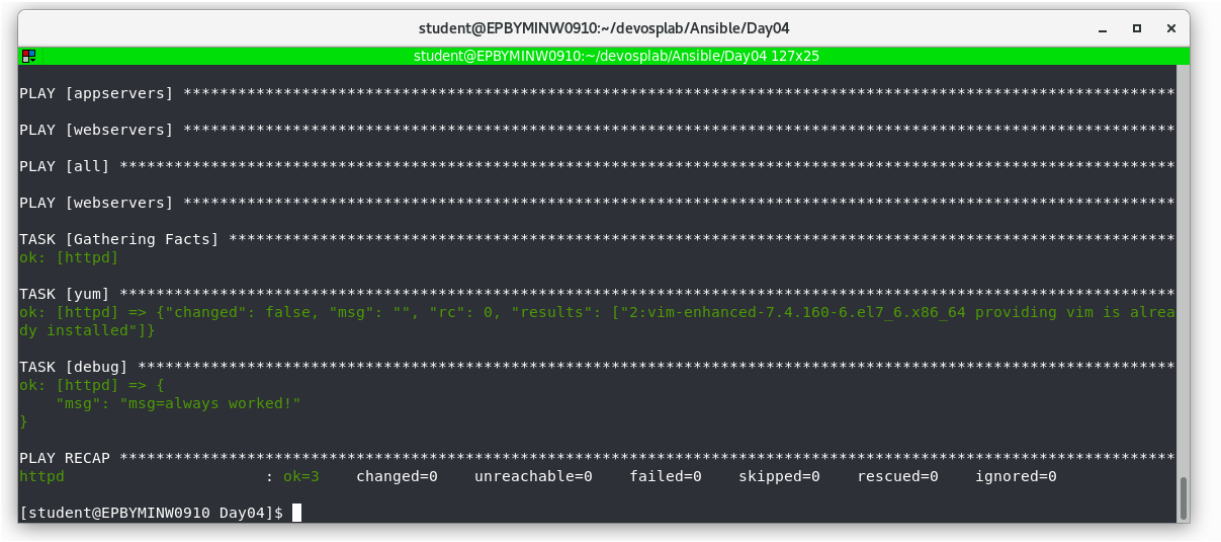


Picture 8.2 – rescue & always worked

Check results without error:

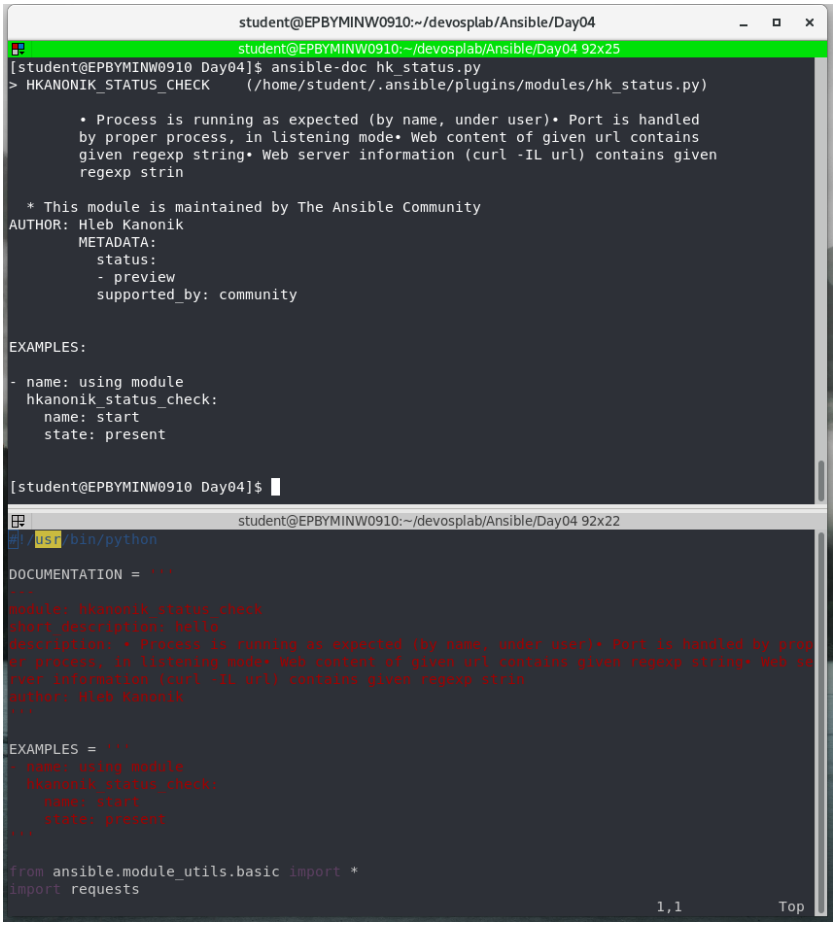


Picture 8.3



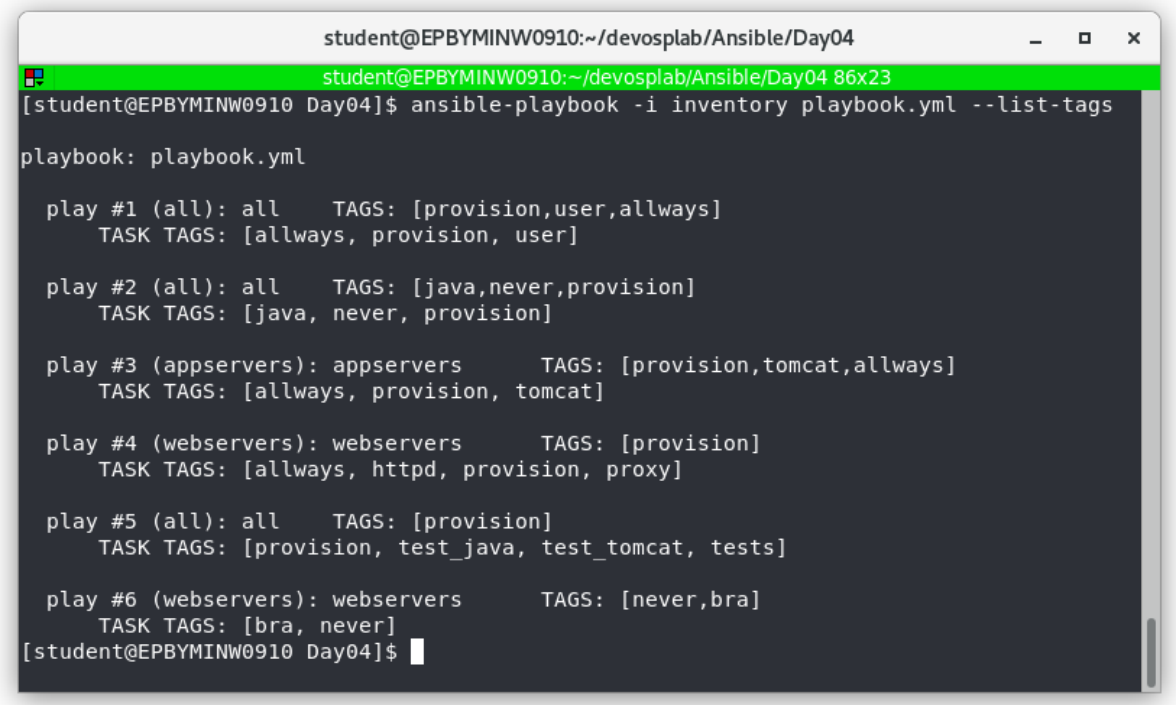
Picture 8.3 – Only always worked

J) Update Custom Module from previous homework to have Documentation built-in. Check documentation availability with ansible-doc command.



Picture 9

All roles in project:



Picture 10