# INIVATA AWS CODING EXERCISE

**Choice 1) Show us a portfolio of your code**:

Existing code you wrote and kept aside (GitHub, loose files, you choose)

That includes terraform (and/or terragrunt calling terraform modules) and Ansible playbooks (one good inclusive playbook is enough). Bonus point (not mandatory): CI pipeline file, Dockerfile for your CI. Anything else you deem relevant. You’re going to need to illustrate your code and answer questions about it, so please don’t send us someone else’s code. It doesn’t need to be complex, it needs to be tidy, and it needs to be yours! We want to see your full potential; show us your best!

**Choice 2) Homework**:

Produce the following (please keep it super simple; we want to see if you can use the tool! No need for complex code/work. This should not take you longer than 2-3 hours. If you don’t know how to do something, please skip it, put a placeholder, write in English what you would do in code if you knew the tool; we’re more interested in seeing what you can do, rather than what you can’t do, so don’t be put off and submit it with blanks!)

***Hint:*** *we care about code linting*

* **Terraform module** that creates a VPC called ‘my-vpc’, a public subnet called ‘my-public-subnet’ in that VPC and the required routes to make it go out to the internet. Add a simple README file that documents the module and how to use it. For the purpose of this exercise, assume the state file is saved in a bucket that already exists and is called ‘my-tf-state’ and you have a dynamodb table for your lock ‘my-tf-lock’.
* **Important hint:** *Using Terraform registry IS ALLOWED (let someone else do the heavy-lifting for you)!*
* **An ansible playbook** that does the following:
  + Connects to a host with name tag my\_instance\_dev (this instance already exists and is an Amazon Linux 2 base AMI, nothing custom, in AWS) (lookup ansible aws dynamic inventory)
  + Creates a user called my\_app\_user uid 1503, group webapps gid 570
  + Deploys an app called my\_app that is in a git repo (details to follow)

**Details for the git repo:**

* + The git repo needs to be cloned in a tmp dir of your choice
  + The git repo is cloned using [git@bitbucket.org:my-org/my\_app.git](mailto:git@bitbucket.org:my-org/my_app.git)
  + The branch to be pulled down is taken from a var that you pass to the playbook
  + The private key for accessing the git repo is (nah, don’t even think about it, this key does not exist in reality anywhere.. ) – **Important hint: think carefully about committing private keys into code repos!**

-----BEGIN OPENSSH PRIVATE KEY-----

b3BlbnNzaC1rZXktdjEAAAAABG5vbmUAAAAEbm9uZQAAAAAAAAABAAAArAAAABNlY2RzYS

1zaGEyLW5pc3RwNTIxAAAACG5pc3RwNTIxAAAAhQQBGC1taQdsrNvbI0XKs1J9wCIlHkIn

S6hgBxDV66yy+FEVFcl9ZtIn0wb0VpJz3GenBZ7aFrZ859hMngutWAAuorgAvt1D6Sq9Vr

oOULyPqrq0LTXXQ+xvxYgZteBnC7LDbHgXOG83oSqJmcG/9NbUmtPo5FSyhXyhPVTl9LUW

xMv3ElsAAAEgDeYdlQ3mHZUAAAATZWNkc2Etc2hhMi1uaXN0cDUyMQAAAAhuaXN0cDUyMQ

AAAIUEARgtbWkHbKzb2yNFyrNSfcAiJR5CJ0uoYAcQ1eussvhRFRXJfWbSJ9MG9FaSc9xn

pwWe2ha2fOfYTJ4LrVgALqK4AL7dQ+kqvVa6DlC8j6q6tC0110Psb8WIGbXgZwuyw2x4Fz

hvN6EqiZnBv/TW1JrT6ORUsoV8oT1U5fS1FsTL9xJbAAAAQgCN33WUUecjJnlDAVlOOLLo

VhSJXGbb3+0r3P8NlVvVm3S0lYwHXFosUt9cIHryVg3Hnj/bgdk7ZTcGyKPyK9jOywAAAB

9hbm5hbGlzYS5iYXp6aUBJTlYtVUstTUFDMjUubGFuAQID

-----END OPENSSH PRIVATE KEY-----



**Details for my\_app:**

* + my\_app requires yum packages python3-devel and nginx (no nginx configurations needed, no python venv requirements, don’t worry, this is a simplified exercise)
  + the code for my\_app is copied from the tmp git clone dir to /opt/my\_app
  + my\_app is installed by running ‘make install’ as my\_app\_user
  + when the app is installed, nginx needs to be restarted