IriusRisk DevOps Technical Challenges

1. Scripting

Using 2 pokemon names from the list of existing pokemons:
https://bulbapedia.bulbagarden.net/wiki/List_of_Pok%C3%A9mon_by_name
Create a script in bash, python or ruby that accepts 2 pokemon names as parameters and compares the attack stats of them. The one with the highest attack wins!
You must use the poke api from https://pokeapi.co/

Note: If you do this script in bash, you can use jq (available on your linux system packages), here is the manual: https://stedolan.github.io/jq/manual/

As an example with bash, this is the input and the expected output:

```
$ ./pokefight.sh charmander squirtle
charmander (52) vs squirtle (48)
charmander wins!
```

2. Docker

You must create a docker-compose.yml file that brings up our example application stack from scratch. The application stack is composed of 2 softwares: An apache that serves our web application and a postgresql database to host information about our application.

The requirements are:

- Use Ubuntu or Centos official docker images as a base (do not use any other public base containers or stripped ubuntu or centos).
- Use the same base image for both containers (do not use ubuntu for one and centos for the other).
- Both of these must be defined and must be controlled from docker-compose as a whole single stack.
- You must use dockerfiles to build the containers inside the docker-compose.yml, it must work by executing "$ docker-compose up -d" only on a clean machine.
- The containers must apply all security and version patches on the system when building.
- Our application is a file outside the apache container for maintainability, it should only be mapped inside the apache container and not included in the container image.
- Our application is a php file which I include here. It only does a connection test against the database. It should show "connected".

Note: you may need to change the pg_connect string pointing it to the right hostname.

```
<?php
$connection = pg_connect ("host=localhost dbname=iriusprod
user=iriusprod password=iriusprod");

if($connection) {
      echo 'connected';
} else {
      echo 'there has been an error connecting';
}
?>
```

- The apache container should wait until the database container is ready to start.
- Only the minimum ports should be exposed outside the containers
- You should use docker networks and assign fixed ip's to the containers
- Database should not be network reachable by anyone than the apache container
- You should try to make the container images as small as possible.
- You should use fixed names on the containers to easily identify them
- Do whatever you need to do in order to achieve all of this in the most efficient way.
- There is no need to use HTTPS for this lab, you can skip it.

3. Ansible

Given the next laboratory:
https://iriusrisk-devops-challenges.s3.eu-west-1.amazonaws.com/ansible.tar.xz

Create an Ansible playbook to upgrade all the prod and staging systems.

Requirements are:

- All systems shall be updated.
- All the staging boxes should have a directory named ~/bin if it does not exist it shall be created.
- In staging, files .bashrc and x.sh should be copied into ~/ and ~/bin respectively.
- Ensure that the script x.sh is run daily at 3:15PM by root user in staging boxes.

Note: If you haven't worked with Ansible before, you can propose a solution in a similar language or pseudocode.