Oefentoets 1 DevOps - gitOps & Containerization

A. gitOps

1. In welke workflow mag je alleen vanuit de 'main' branch naar de productie omgeving opleveren?

    a. git flow
    b. GitHub flow
    c. Jenkins flow
    d. Develop flow

2. Je wilt een container starten met de image nginx, maar geeft geen expliciete tag op in het commando docker run nginx. Wat gebeurt er met de tag van de image die wordt gebruikt?

    a. De container start zonder tag, Docker gebruikt alleen de naam van de image
    b. Docker gebruikt automatisch de latest tag als deze beschikbaar is
    c. Docker pakt de laatste stabiele versie die is gecommitteerd
    d. Docker kiest een willekeurige versie van de image, ongeacht de tag

3. Wat is een geldige uitspraak over Git?

    a. First push before you commit!
    b. First push before you pull!
    c. Branching is cheap!
    d. Branch only if you're a maintainer!

4. Hoe maak je een nieuwe branch `develop`?

    a. `git branch create develop`
    b. `git create branch develop`
    c. `git checkout -b develop`
    d. `git checkout develop`

5. Welk commando doet het tegenovergestelde van `git clone ssh://git@gitlab.com/student1/weekopdr`

    a. `git unclone ssh://git@gitlab.com/student1/weekopdracht-1.git`
    b. `git unclone weekopdracht-1`
    c. `rm -rf .git`
    d. `rm -rf weekopdracht-1`
    e. Strikvraag! git ondersteunt alleen `https:` en `git:` protocol

6. De git historie tree is als volgt:

```
          A---B---C feature-1
         /
D---E---F---G main
```

   De huidige branch is `feature-1` en je geeft het commando `git rebase master`. Hoe ziet de 'history tree' er daarna uit?

   a.

```
                    A'--B'--C' main
                   /
D---E---F---G feature-1
```

   b.

```
                     A'--B'--C' feature-1
                   /
D---E---F---G main
```

   c.

```
             A'--B'--C' feature-1
            /
D---E---F---G main
```

   d.

```
             A'--B'--C' main
            /
D---E---F---G feature-1
```

B. Containerization/Docker

7. Wat doet het `.dockerignore` bestand?

   a. Bestanden of paden die hierin staan gaan niet mee in de *build context*
   b. Bestanden of paden die hierin staan komen niet in de *container registry*
   c. Bestanden of paden die hierin staan worden niet gecommit
   d. Bestanden of paden die hierin staan komen in de speciale *ignored layer*

8. Wat is in een Docker 'recept' de *preferred form*?

   a. `RUN command param1 param2`
   b. `RUN ["param1", "param2"]`
   c. `RUN ["executable", "param1", "param2"]`
   d. `["RUN", "param1", "param2"]`

9. Welk Docker 'commando' staat in *exec form*?

    a. `RUN command param1 param2`
    b. `CMD ["param1", "param2"]`
    c. `CMD ["executable", "param1", "param2"]`
    d. `["CMD", "param1", "param2"]`

Open vragen

10. Geef drie voorbeelden van 'package managers' die horen bij front-end of back-end applicaties (3 pnt).

11. Beschrijf in eigen woorden wat de term *reconcilliation* ('in overeenstemming brengen') inhoudt binnen GitOps. (3 pnt)

    Dit mag in het Engels of Nederlands. Verwerk in je beschrijving (wel) minstens 3 van de volgende 6 woorden op een passende manier:

    i. versiebeheer / version control
    ii. gewenste sitatie / desired state
    iii. imperatief / imperative
    iv. declaratief / declarative
    v. push / push
    vi. pull / pull