Inception-of-Things — UPDATED

Evaluation Checklist (v4.0)

Preliminaries

Before starting:

- ☐ Defense can only happen if the evaluated group is present
- ☐ No empty work / wrong files / wrong directory / wrong filenames (grade = 0 if failed)
- ☐ Clone Git repository on the group's machine
- ☐ Ensure folders `p1/`, `p2/`, `p3/` exist at repo root (optional `bonus/`)

Global Configuration and Explanation

Evaluated students must explain in simple terms:

- ☐ Basic operation of K3s
- ☐ Basic operation of Vagrant
- ☐ Basic operation of K3d
- ☐ What is continuous integration and Argo CD

Part 1: K3s and Vagrant

Configuration Checks

- ☐ `p1/Vagrantfile` exists and is understandable (similar to subject example)
- ☐ Exactly 2 virtual machines defined in Vagrantfile
- ☐ **NEW**: Uses latest stable version of distribution of their choice (NOT mandatory CentOS anymore)
- ☐ **NEW**: Private network interface with IPs 192.168.56.110 (server) and 192.168.56.111 (worker) — interface name may vary (enp0s8, enp0s9, eth1, etc.)
- ☐ VM names include a team member login + S (Server) and SW (ServerWorker)
- ☐ Scripts present: `p1/scripts/k3s_server.sh` and `p1/scripts/k3s_worker.sh` (or equivalent)

If something doesn't work → evaluation stops here

Usage Checks

- ☐ Use `vagrant ssh` to connect to both VMs
- ☐ **NEW**: Verify IP/interface using `ip a` (or `ip a show <interface_name>`) instead of `ifconfig eth1` — modern distros use predictable interface names (enp0s8, enp0s9, eth1)
- ☐ Hostnames are correct: `<login>S` and `<login>SW`
- ☐ Both VMs use K3s (server mode + agent mode)
- ☐ Verify K3s services are active and enabled:
  - Server: `sudo systemctl is-active k3s && sudo systemctl is-enabled k3s`
  - Worker: `sudo systemctl is-active k3s-agent && sudo systemctl is-enabled k3s-agent`
- ☐ Verify cluster with `kubectl get nodes -o wide` on Server machine — output should show both nodes
- ☐ Both nodes show STATUS: Ready
- ☐ Both nodes show correct INTERNAL-IP: 192.168.56.110 and 192.168.56.111
- ☐ Evaluated group must explain the output

If something doesn't work → evaluation stops here

Part 2: K3s and Three Simple Applications

Configuration Checks

- ☐ Shut down other VMs to avoid space/performance issues (optional but recommended)
- ☐ `p2/Vagrantfile` exists and is similar to Part 1 style
- ☐ Only 1 virtual machine defined
- ☐ **NEW**: Uses latest stable version of distribution of their choice
- ☐ **NEW**: Private network interface with IP 192.168.56.110 — interface name may vary
- ☐ VM name is `<login>S`
- ☐ Configuration files present: `p2/confs/apps-ingress.yaml` (or separate files)
- ☐ Extra files in `p2/` folder? Ask for explanations

If something doesn't work → evaluation stops here

Usage Checks

- ☐ Use `vagrant ssh` to connect to the VM
- ☐ **NEW**: Verify IP/interface using `ip a` or `ip a show <interface_name>`
- ☐ Hostname is correct: `<login>S`
- ☐ VM uses K3s in server mode
- ☐ Run `kubectl get nodes -o wide` — should show controller name + internal IP (192.168.56.110)
- ☐ Run `kubectl get all -n webapps` — should display:
  - 3 deployments: app1-deployment (1 replica), app2-deployment (3 replicas), app3-deployment (1 replica)
  - 3 services: app1-service, app2-service, app3-service
  - 5 pods total (1 app1 + 3 app2 + 1 app3), all Running
- ☐ Traefik ingress controller running: `kubectl -n kube-system get deploy,svc traefik`
- ☐ Ingress configured: `kubectl -n webapps get ingress` — should show webapps-ingress with hosts app1.com, app2.com
- ☐ Evaluated group must explain each output
- ☐ Demonstrate Ingress works (command deliberately not given — they must show you)
- ☐ Access 3 applications by changing Host header (use curl or browser):
  - `curl -H 'Host: app1.com' http://192.168.56.110` → shows app1
  - `curl -H 'Host: app2.com' http://192.168.56.110` → shows app2
  - `curl http://192.168.56.110` (no Host header or default) → shows app3

If something doesn't work → evaluation stops here

Part 3: K3d and Argo CD

Configuration Checks

- ☐ With evaluated group's help, start up the infrastructure
- ☐ Configuration files present in `p3/` folder — check content and ask for explanations:
  - `p3/confs/argocd-app.yaml` (Argo CD Application manifest)
  - `p3/dev-app/deployment.yaml` (or `p3/k8s/dev/deployment.yaml`)
  - `p3/dev-app/service.yaml` (or `p3/k8s/dev/service.yaml`)
- ☐ Setup script present: `p3/scripts/install_k3d_argocd.sh` (or equivalent)
- ☐ At least 2 namespaces in K3d: `argocd` and `dev` — verify with `kubectl get ns`
- ☐ At least 1 pod in `dev` namespace — verify with `kubectl get pods -n dev`
- ☐ Group members understand difference between namespace and pod
- ☐ All required Argo CD services running (7 pods expected):

    - `kubectl get pods -n argocd` — should show all pods Running
- ☐ Argo CD installed and configured — accessible in web browser with login/password (group provides credentials)
- ☐ GitHub repo name includes a member login (examples: wil_config, wil-ception, usrali2026/Inception_of_Things)
- ☐ Docker image used in GitHub repo — can be Wil's (`wil42/playground`) or custom
- ☐ If custom Docker image: Docker Hub repo name includes a member login
- ☐ Verify two required tags exist in Docker Hub: v1 and v2
- ☐ Argo CD Application configured with:
    - Correct repoURL pointing to GitHub repository
    - Correct path (e.g., `p3/dev-app` or `p3/k8s/dev`)
    - Auto-sync enabled (if applicable)
- ☐ Extra files in `p3/`? Ask for explanations

If something doesn't work → evaluation stops here

Usage Checks (The GitOps Flow)

- ☐ Navigate through Argo CD application with evaluated group's help — understand how it works
- ☐ If explanations are confused or they can't explain something → evaluation stops now (this is critical)
- ☐ Verify Argo CD Application status:
    - `kubectl get application -n argocd` — should show dev-app, Synced, Healthy
- ☐ Verify v1 application is accessible — check pod is running with v1 image:
    - `kubectl get deployment -n dev -o jsonpath='{.spec.template.spec.containers[0].image}'`
    - Should show: `wil42/playground:v1` (or custom image with v1 tag)
- ☐ Verify Docker Hub is used (important — if any doubt, evaluation stops)
- ☐ Update the application with evaluated group's help:
    - Edit configuration file in GitHub that Argo CD watches (change v1 → v2)
    - Commit and push modification
    - Understand this triggers automatic update
    - Must be able to explain the whole process
- ☐ After pushing v2 to GitHub:
    - If auto-sync didn't happen → manually sync in Argo CD (or trigger refresh)
    - If auto-sync happened → skip manual sync
- ☐ Verify application was successfully synchronized:
    - `kubectl get application dev-app -n argocd` — should show Synced
    - `kubectl get pods -n dev` — should show new pod with v2 image
    - Old pod with v1 should be terminated (rolling update)
- ☐ Confirm v2 is running:
    - `kubectl get deployment -n dev -o jsonpath='{.spec.template.spec.containers[0].image}'` → should show v2
    - `kubectl get pod -n dev -o jsonpath='{.items[0].spec.containers[0].image}'` → should show v2
- ☐ Verify rollback capability (optional but recommended):
    - Change back to v1, commit, push
    - Verify Argo CD syncs back to v1

If something doesn't work → evaluation stops now

Bonus: GitLab Integration

Only evaluate bonus if mandatory part is flawless

- ☐ Configuration files exist in `bonus/` folder — ask for explanations
- ☐ GitLab functions correctly and is properly implemented

- ☐ GitLab deployed in Kubernetes cluster (namespace: gitlab)
- ☐ Create a new repository in GitLab with evaluated group's help
- ☐ Add some code to it — verify operation successful in GitLab
- ☐ Part 3 operations still function correctly
- ☐ Repository used in Argo CD is local GitLab repository (not GitHub)
- ☐ GitLab repo contains the two versions (v1/v2) of chosen application
- ☐ Synchronization and version change (v1 → v2) complete with no errors

If synchronization works → validate bonus

Final Ratings

Check appropriate flag:

- ☐ **Ok** — Mandatory complete
- ☐ **Outstanding project** — Mandatory flawless + bonus works
- ☐ **Empty work** — No files / wrong structure
- ☐ **Incomplete work** — Parts missing or broken
- ☐ **Cheat** — Suspicious behavior detected
- ☐ **Crash** — Serious errors
- ☐ **Incomplete group** — Missing team members
- ☐ **Concerning situation** — Issues detected
- ☐ **Forbidden function** — Unauthorized tools used

Leave a comment on this evaluation

Key Differences from Old Scale

Aspect	Old Scale (CentOS-focused)	New Subject v4.0
OS requirement	"latest stable CentOS" mandatory	"distribution of your choice"
Network interface check	`ifconfig eth1`	`ip a` or `ip a show <interface_name>` (predictable names)
Interface names	eth0/eth1 expected	enp0s8, enp0s9, eth1, or other predictable names
Modern practices note	Not emphasized	Explicit "modern practices" note added
Part 2 namespace	Not specified	`webapps` namespace expected
Part 3 structure	Generic	Specific folder structure: `p3/dev-app/` or `p3/k8s/dev/`

This updated checklist now matches the subject v4.0 exactly, accounting for distribution flexibility, modern interface naming, and current project structure.

Notes for Evaluators

Part 1 Specific Checks:

- Verify both VMs are running: `vagrant status`

- Check hostnames match pattern: `<login>S` and `<login>SW`
- Verify K3s services are both active AND enabled (not just active)
- Network interface may be eth1, enp0s8, enp0s9, or other predictable names

Part 2 Specific Checks:

- Applications should be in `webapps` namespace (not default)
- Verify Traefik is running (K3s default ingress controller)
- Ingress should route based on Host header
- Default backend (app3) should handle requests without Host header

Part 3 Specific Checks:

- Argo CD Application should point to correct GitHub repository
- Path in Argo CD Application should match actual folder structure
- Verify GitOps workflow: Git push → Argo CD sync → Pod update
- Check that both v1 and v2 tags exist and work

Version: 4.0
Last Updated: February 2026
Based on: Subject v4.0 and current project implementation