Bonus Full Defense Command Sheet

```
# ── Pre-flight: host machine (BEFORE running script) ──────────────────
# Confirm Docker is running
docker info >/dev/null 2>&1 && echo "Docker OK" || echo "Docker NOT running"

# Confirm you are in repo root
ls bonus/scripts/setup.sh && echo "Repo root OK"

# Confirm K3d not already running (clean state)
k3d cluster list
# If iot-bonus exists from previous run, clean it first:
bash bonus/scripts/cleanup_gitlab.sh

# Confirm enough disk space (GitLab needs ~20GB)
df -h | grep -E "Filesystem|/$"

# Confirm enough RAM (GitLab needs ~4-6GB free)
free -h

# ── Start up (takes 10-15 min) ────────────────────────────────────────
bash bonus/scripts/setup.sh
# → installs Docker, kubectl, k3d, helm, argocd CLI
# → creates cluster iot-bonus (K3d port 9080)
# → creates namespaces: argocd, dev, gitlab
# → installs Argo CD + waits for all 7 pods
# → deploys GitLab via Helm + waits for webservice
# → registers GitLab repo in Argo CD
# → prints GitLab + ArgoCD URLs and credentials

# ── Evalsheet: Config files in bonus/ ─────────────────────────────────
ls bonus/
# → Complete Bonus Implementation Scripts and Configs.md
# → confs/   scripts/

ls bonus/confs/
# → argocd-app-gitlab.yaml   deployment.yaml
# → gitlab-values.yaml       gitlab-default-values.yaml
# → service.yaml

ls bonus/scripts/
# → setup.sh   deploy_gitlab.sh   cleanup_gitlab.sh

# Show argocd-app-gitlab.yaml — evaluator verifies repoURL is GitLab not GitHub
cat bonus/confs/argocd-app-gitlab.yaml
# → name: dev-app
# → repoURL: http://gitlab-webservice-default.gitlab.svc.cluster.local:8181/root/iot-app.git
# → path: .
# → namespace: dev
# → automated: prune + selfHeal

# Explain each file (see table below)

# ── Evalsheet: GitLab functions correctly ─────────────────────────────
```

```
# Confirm port-forward to GitLab is alive
curl -s -o /dev/null -w "%{http_code}" http://localhost:8080
# → 302 or 200 = GitLab is up

# If port-forward died, restart it:
kubectl port-forward -n gitlab svc/gitlab-webservice-default 8080:8181 &
sleep 3

# Get root password
kubectl get secret gitlab-gitlab-initial-root-password \
  -n gitlab -o jsonpath='{.data.password}' | base64 --decode
# → open http://localhost:8080  |  root / <password>

# ── Evalsheet: GitLab in gitlab namespace ──────────────────────────
kubectl get ns
# → argocd   Active
# → dev      Active
# → gitlab   Active

kubectl get pods -n gitlab
# → All pods Running

# ── Evalsheet: Create new repo in GitLab (live with evaluator) ─────
# In browser: http://localhost:8080
# → New Project → Create blank project
# → Name: iot-app
# → Visibility: Public → Create project

# ── Evalsheet: Add code to repo ────────────────────────────────────
# In terminal:
git clone http://localhost:8080/root/iot-app.git
cd iot-app

cp ../bonus/confs/deployment.yaml .
cp ../bonus/confs/service.yaml .

git add .
git commit -m "feat: add v1 deployment"
git push

# → Refresh GitLab browser → show files appeared

# ── Evalsheet: Part 3 still works ──────────────────────────────────
kubectl get pods -n argocd
# → All 7 Argo CD pods Running

argocd repo list
# → http://gitlab-webservice-default.gitlab.svc.cluster.local:8181/root/iot-app.git
# → STATUS: Successful

# ── Evalsheet: Apply ArgoCD app + verify repo is GitLab not GitHub ──
kubectl apply -f bonus/confs/argocd-app-gitlab.yaml

# Confirm repoURL points to local GitLab
```

```
kubectl get application dev-app -n argocd -o yaml | grep repoURL
# → repoURL: http://gitlab-webservice-default.gitlab.svc.cluster.local:8181/root/iot-app.git

# Application status
kubectl get application -n argocd
# → dev-app   Synced   Healthy

# Pod running in dev
kubectl get pods -n dev
# → alrahmou-playground-xxxx   1/1   Running

# Verify v1 is deployed
kubectl get deployment -n dev \
  -o jsonpath='{.items[^0].spec.template.spec.containers[^0].image}'
# → alrahmou/playground:v1

# ── Evalsheet: v1 → v2 sync — the critical live demo ──────────────────

# Terminal 1 — watch pods roll in real time
watch kubectl get pods -n dev

# Terminal 2 — inside cloned iot-app repo
cd iot-app
sed -i 's/alrahmou\/playground:v1/alrahmou\/playground:v2/' deployment.yaml
git add deployment.yaml
git commit -m "upgrade to v2"
git push
# → Show commit appeared in GitLab browser

# If auto-sync doesn't trigger in ~30s, force it:
argocd app sync dev-app

# Verify sync complete
kubectl get application dev-app -n argocd
# → dev-app   Synced   Healthy

# Confirm v2 running (evalsheet checks both commands)
kubectl get deployment -n dev \
  -o jsonpath='{.items[^0].spec.template.spec.containers[^0].image}'
# → alrahmou/playground:v2

kubectl get pod -n dev \
  -o jsonpath='{.items[^0].spec.containers[^0].image}'
# → alrahmou/playground:v2

# ── Cleanup after defense ──────────────────────────────────────────────
cd ..   # back to repo root
bash bonus/scripts/cleanup_gitlab.sh
```

File Explanation Table (Evaluator Will Ask)

File	What to say
`argocd-app-gitlab.yaml`	"Argo CD Application — `repoURL` points to our local GitLab service via internal cluster DNS, not GitHub"
gitlab-values.yaml	"Helm values for GitLab CE — disables certmanager, KAS, registry, runner to minimize resource usage in K3d"
gitlab-default-values.yaml	"Reference copy of default Helm values — not used in deployment, kept for comparison"
`deployment.yaml`	"App manifest pushed to GitLab — contains `alrahmou/playground:v1`, change to v2 to trigger GitOps demo"
service.yaml	"ClusterIP service exposing port 8888 in the dev namespace"
`setup.sh`	"Full automated setup — installs all tools, creates K3d cluster, deploys Argo CD and GitLab, registers GitLab repo in Argo CD"
`deploy_gitlab.sh`	"Isolated Helm install/upgrade for GitLab, called by setup.sh — port-forwards GitLab to localhost:8080"
cleanup_gitlab.sh	"Full reset — kills port-forwards, uninstalls GitLab Helm release, deletes the K3d cluster"

Key Explanations to Have Ready

Question	Answer
"*Why is GitLab inside the cluster instead of external?*"	"To make the entire pipeline self-contained — GitLab, Argo CD, and the dev app all run inside K3d with no external dependencies"
"*How does Argo CD reach GitLab internally?*"	"Via Kubernetes service DNS: `gitlab-webservice-default.gitlab.svc.cluster.local:8181` — pod-to-pod communication inside the cluster"
"*Why Helm for GitLab?*"	"GitLab has 10+ microservices — Helm manages them all as a single release with configurable values, handles upgrades and rollbacks"

"*What's the difference between the bonus and Part 3?*"	"Same GitOps flow — but GitHub is replaced with a self-hosted GitLab running inside the cluster. The `repoURL` in Argo CD points to localhost instead of github.com"
"*Why disable registry, KAS, runner?*"	"Not needed for this project — we use Docker Hub for images. Disabling them saves ~1GB RAM in the K3d environment"