# Proxmox: Cloning VMs that Auto-Start a Docker App (Example: Uptime Kuma)

This guide shows how to feed a working docker-compose.yml into your Proxmox clone workflow so the app starts on first boot. It uses your known-good clone script (clone\_from\_deb12\_docker\_base.sh) and adds two helpers:  
 • make\_user\_data\_from\_compose.sh – converts any compose file to a cloud-init user-data snippet for a specific VM  
 • compose\_clone.sh – convenience wrapper that combines: generate snippet → clone VM → bind snippet  
  
Your base template (VMID 9000) is Debian 12 with Docker and QEMU guest agent already present.

## Example App: Uptime Kuma

A simple, popular uptime monitor with a web UI at http://<VM-IP>:3001

docker-compose.yml:

version: "3.8"  
  
services:  
 uptime-kuma:  
 image: louislam/uptime-kuma:1  
 container\_name: uptime-kuma  
 restart: unless-stopped  
 ports:  
 - "3001:3001"  
 volumes:  
 - /opt/uptime-kuma/data:/app/data  
 environment:  
 # Adjust timezone as needed  
 - TZ=America/New\_York

## Install the helpers (once)

Copy these files to your Proxmox node (adjust paths as desired):  
 - docker-compose.yml (example app; use your real compose)  
 - make\_user\_data\_from\_compose.sh → /usr/local/sbin/make\_user\_data\_from\_compose.sh  
 - compose\_clone.sh → /usr/local/sbin/compose\_clone.sh  
  
Then:  
 chmod +x /usr/local/sbin/make\_user\_data\_from\_compose.sh /usr/local/sbin/compose\_clone.sh  
  
Assumes your working clone script lives at /root/clone\_from\_deb12\_docker\_base.sh and uses template 9000.

## Fast path (one command)

compose\_clone.sh 950 kuma-01 ~/.ssh/id\_ed25519.pub 'StrongPassw0rd!' /mnt/pve/pve-qnap/apps/kuma/docker-compose.yml

This will:  
 1) Create /var/lib/vz/snippets/kuma-01.yaml from your compose.  
 2) Clone 9000 → VMID 950, name kuma-01, inject SSH key + password.  
 3) Bind the per-VM user-data and update cloud-init so Docker brings the app up on first boot.  
  
Open: http://<VM-IP>:3001

## Manual steps (if you don’t use the wrapper)

1) Build per-VM user-data from a compose file:  
 make\_user\_data\_from\_compose.sh kuma-02 /path/to/docker-compose.yml  
  
2) Clone using your working script:  
 /root/clone\_from\_deb12\_docker\_base.sh 951 kuma-02 ~/.ssh/id\_ed25519.pub 'StrongPassw0rd!'  
  
3) Bind user-data and update cloud-init:  
 qm set 951 --cicustom "user=local:snippets/kuma-02.yaml"  
 qm cloudinit update 951  
 # If needed: qm start 951

## Why this keeps login working

We don’t put users: or ssh\_pwauth: into user-data. All login settings come from:  
 qm set <vmid> --ciuser dockeruser --cipassword '...' --sshkey ~/.ssh/id\_ed25519.pub  
This is the method you confirmed works reliably.

## Tips

• Disk resize: pass size to your clone script (before first boot) so the filesystem autogrows.  
• .env/secrets: add another write\_files block or mount via NFS and reference absolute paths.  
• Idempotent: re-running the same compose is fine; Docker will reconcile.  
• Logs: journalctl -u docker -e and docker ps -a are good first checks.