# Low-Value Task Delegation Checklist

For Physician-Developers | DoctorsWhoCode.blog

*Burnout isn't from working too hard. It's from working on the wrong things.*

doctorswhocode.blog

Use this checklist to audit your current workload and identify tasks that are draining time without requiring your clinical brain. For each item, check whether the task is currently on your plate — then decide: **automate, delegate, or eliminate.**

## 1 Identify the Low-Value Tax

Tasks you shouldn't be doing yourself

- [ ] **Repetitive build or deployment failures**
  Broken tool pages, failed deploys on OpenMFM or DoctorsWhoCode.blog that require manual intervention each time.
- [ ] **Recurring documentation formatting errors**
  APSO notes, consultation templates, or PDF outputs that break after dependency updates.
- [ ] **Manual form validation checks**
  Interactive clinical tools (preeclampsia calculators, cervical length trackers) that require manual testing after every update.
- [ ] **Boilerplate setup for each new project**
  Scaffolding a new microsite, setting up a dev environment, or pulling standard configurations from scratch.
- [ ] **Repetitive inbox triage**
  Emails or messages that follow a predictable pattern and consume decision-making bandwidth without delivering clinical or intellectual value.

## 2 Evaluate Delegation Options

Who or what handles this instead of you

- [ ] **Can traditional code solve this deterministically?**
  Cloning repos, checking file configurations, running test suites — if the outcome is predictable, use a script, not an LLM.
- [ ] **Does this require reasoning over ambiguous inputs?**
  Error diagnosis, draft generation, pattern interpretation — this is where an AI agent adds real value.

- [ ] **Is there a home-network or cloud node that could run this automatically?**
  Consider lightweight compute (mini PCs, VPS nodes) that trigger on failure events rather than requiring your presence.

- [ ] **Could a well-prompted agent generate a pull request or draft output for your review?**
  Human-in-the-loop review is still high value. Removing the generation step from your plate is the win.

## 3 Build Observability Before You Deploy

Delegation without visibility is abdication

- [ ] **Every automated task logs its output**
  You should be able to audit what happened on any individual run without manual intervention.

- [ ] **API calls are proxied and token costs tracked**
  Use a proxy layer (e.g., Tailscale Aperture or equivalent) so API keys never live on agent nodes and all calls are logged centrally.

- [ ] **Failure alerts route to you — not to silence**
  An automation that fails quietly is worse than no automation. Define what a failed run looks like and where you hear about it.

- [ ] **You can reconstruct what the agent did on any given run**
  Granular logging per job, not just aggregate status. Especially important for tools touching clinical content.

## 4 Treat Your Automation Like a Product

Version 1 is a prototype — plan to improve it

- [ ] **Does v1 solve the core interruption?**
  Even an imperfect automation that handles 70% of cases buys back significant time. Ship it, then refine.

- [ ] **Have you defined what 'good output' looks like for review?**
  You are still the reviewer. Define your acceptance criteria so review doesn't take as long as the original task.

- [ ] **Is there a feedback loop from failures back to the automation?**
  Edge cases you discover should improve the agent's prompts, triggers, or environment setup.

- [ ] **Is the automation documented well enough that future-you can maintain it?**
  A workflow only you can maintain and only you can debug is not delegation. Write the README.

**The Core Principle**

*The question is not whether you are working hard enough.*
*It is whether the things you are working on deserve your brain.*