Lab's Best Practices for Developer Efficiency

Operating System

Hardware is only as good as the VPN access / VDI requirements, but some things that are important to consider when selecting your team's computers are:

- Processing speed
- Ability for customization by individual devs

Some good questions to ask around your hardware are: What type of developer access will we have with this hardware? Can developers download apps they think are important?

> Strive to develop in an environment as close as you can to the one you'll be deploying to.

See Operating System tool comparison chart →

Network Access

VPNs can interfere with high-fidelity remote pairing tooling (for example, Tuple).

Open internet or VPN that only requires some traffic to be funneled through it have made it easier for our teams to collaborate in the past.

> Not all VPNs are equal. Even if your VPN allows some traffic through, check to make sure your tools aren't still being blocked.

Engineering Privileges

Local admin access, which allows engineers to install software locally and set up their workstations, leads to a more efficient onboarding process and enables teams to be more effective from the start.

Engineers greatly benefit from Source control and CI/CD privileges, which facilitate seamless collaboration and expedite the development process.
Deployment permissions enable rapid software development and user feedback cycles.

> Empower trust and accountability within the team by giving them the privileges for setting up their workstations and controlling the development process.

Install Dependency

Quick or automated approvals for adding new dependencies enable engineers to widen the engineering toolset and maintain past-paced development.
A longer approval ticket or self service process blocks the current work track and hinders story progress.
A 72-hour approval process could halt development for over half a week.

> Minimizing approvals and time-to-approval can directly improve developer experience, team velocity, and software delivery.

Backlog

Backlog management tools help organizations to track work across teams in an established way. Having one organized backlog facilitates better team alignment and stakeholder alignment and allows us to make clear prioritized decisions.

Our team believes Pivotal Tracker shines most when developing a new product, keeping the backlog laser-focused on the work at hand, taking a streamlined approach versus its competitors.

> Organize your backlog in a way that lets your team continue to pick up clear stories as they complete their work. Accomplish this with detailed stories, consistent pointing, and an established system of tagging.

See Backlog tool comparison chart →

Design

Design tools are necessary for many steps in the development process like wireframing, prototyping, and mocking up high fidelity designs. Since your designers will be in this tool everyday, it's best to use one that works best for them. Our designers have mentioned these features that are important to them in a tool:

- Collaboration / multi cursors in one file
- Ease of use/intuitive
- Speed/performance (loading large files)

Almost all of our designers agree that Figma is the current industry standard because of its ability to design, prototype, and share work all in one platform.

> Design files can get cluttered quickly. It can be helpful for both your designers and engineers (and anyone who needs to see a design file) if your design tool has organization features that make sharing and viewing easy.

See Design tool comparison chart →

IDE

When choosing an IDE, one of the most important factors is how fluent your developers are in that specific environment. The better they can use the IDE, the faster they can translate ideas into code.

However, smarter IDEs can understand your code and make it easier to make changes, refactor, and test.

The Tanzu Labs team are most used to IntelliJ and the suite of JetBrains IDEs which come by default with VMware provided laptops.

> Design files can get cluttered quickly. It can be helpful for both your designers and engineers (and anyone who needs to see a design file) if your design tool has organization features that make sharing and viewing easy.

See IDE tool comparison chart →

Pair Programming

Developers work in pairs. Pair programming ensures that development decisions are shared and discussed, maximizing the quality of code. It allows new team members to ramp up rapidly, helps reduce knowledge silos, and encourages shared team ownership of the code.

While co-location is ideal, remote pairing can be done using video chat and screen sharing tools that let pairs look at the exact same thing and have equal control. Ideally, pairs experience real time collaboration - any lag or delay in communication causes a slowdown.

Labs developers prefer Tuple, Pop, or Drovio because these tools allow annotation, sharing a cursor, and have high screen resolution.

> Sometimes a best option is a combination of tools, such as a video conference tool (like Zoom or Teams) plus a tool that is exclusively for pair programming (like Pop or Tuple), Experiment with different options to find the best for your team their pairing styles.

See Pair Programming tool comparison chart →

Whiteboarding

In-person whiteboard collaboration cannot be replaced, but tools exist that make group brainstorming easier. Digital whiteboarding tools can help facilitate both real time and async collaboration with different features that mimic in-person sessions such as:

- Sticky notes
- Commenting and voting
- Timers
- Flow charts, and more

Team members of all disciplines across Labs have become fluent in using Miro, which allows for seamless collaboration at all stages of development. A close alternative is Figjam, especially for our members who are already familiar with Figma's UI.

> A blank canvas might be a daunting starting point, but choosing a tool that has premade templates can help your team get started faster.

See Whiteboarding tool comparison chart →

Video Conferencing

At the heart of remote and hybrid working environments is video conferencing. Being able to host virtual meetings is critical for dispersed teams to help stay connected and collaborate together in real time.

Specifically with pairing, much of communication can sometimes be non-verbal. Seeing if your pair is happy, excited, sad, or frustrated may help improve overall team dynamic.

At Labs, all our team members have premium Zoom accounts for unlimited meeting time.

> Choose a video conferencing tool that allows you to integrate with your team's calendar. Creating meetings becomes easier when you can automatically have a conference room ready to go.

See Video Conferencing tool comparison chart →

A-sync Communication

The best async tools should facilitate quick and easy communication between group members.

Features such as channels allow us to manage projects in a convenient thread.

All Labs members (as well as the rest of VMware) are on Slack.

> Some async communication platforms allow guest accounts to join certain channels or groups if your client isn't yet on the same platform as you.

See A-sync Communication tool comparison chart →

Number of Timezones

Collaborating across time zones can be surprisingly difficult, and it takes a concerted effort to ensure effective communication for widely dispersed teams. Consider the start time, end time, and lunch break of team members in each time zone.

Another factor to consider is the spread of your timezones. You may only have your team across 2 time zones, but what is the difference between these two locations?

Labs has found that having all team members in the same timezone maximizes the amount of time we can spend pairing.

> Facilitate a team calendar workshop to align the team's collaboration expectations and optimize for key team meetings during everyone's working hours.

CI/CD Tools

Continuous Integration refers to an automated pipeline to test and compile the build artifact – letting the team know if the build breaks for any reason. Continuous Deployment is automation that rolls out every build that is created to a series of environments, including production.

Our developers have been most efficient when they have been able to customize the entire pipeline.

> Track how well your organization is delivering software using DORA metrics.

See CI/CD tool comparison chart →

Push to Prod

We want to be able to push to production constantly which means we need to be automating deployment. Manually deploying introduces risk because humans can make mistakes.

For your push to production platform, the main question you want to consider is where is your software going to run?

Knowing this platform helps us correctly staff the right developers experienced with your platform for your project.

> Run a Path to Production exercise with your team to ensure all your developers are on the same page.

See Push to Prod tool comparison chart →