VK Guide

How to build remote teams properly

🔥Blog | 📰 Newsletter | 🚀 Founders Guide | 🦸 CTO Guide

Vadim Kravcenko - *12th April 2022*

This article is part of the [Technical Manager Guide]() that I'm writing for technical leads to scale their development and be good leaders. Being a good manager is all about empowering & enabling your teammates.

While there were many skeptics of remote workspaces a few years back, most of the companies in the last year had to try out remote work in one way or the other. To their surprise, the effectiveness of the people did not drop when they worked from home. Who could've thought? In my opinion, a decentralized, remote workforce is the future of the digital economy. Winning at the remote game means building your company in a way that empowers the employees, establishes self-sufficient teams that function autonomously with little supervision. Of course, the whole concept revolves around trust, but hey, if you don't trust your employees, that's another issue.

I'D LIKE TO WORK FROM HOME SO I CAN BE MORE PRODUCTIVE.
I CAN'T MANAGE YOU AS EASILY WHEN YOU'RE OUT OF THE OFFICE.
THAT'S WHY I'D BE MORE PRODUC-TIVE.
BUT YOU'D BE MISSING OUT ON ALL OF THIS.
DILBERT.COM @SCOTTADAMSSAYS
8-9-19 2019 Scott Adams, Inc./Dist. by Andrews McMeel

Dilbert comics always on spot

There will always be people who abuse the system, abuse your trust and try to game the mechanics for their benefit. That's fine, as soon as you find these people, let them go. However, they probably need a stricter environment to function correctly, more robust control over their work - meaning they thrive when someone is looking over their shoulder. A remote gig is not a good fit for these kinds of people. Try to focus on the employees who thrive in such environments rather than those few bad apples. I will elaborate later on some tools that you can use to mitigate the most common pitfalls regarding trust, but we'll come to that later. Over the years, I've had the privilege to be part of fantastic remote teams and building some myself. So here are a few tips that I put down together that would be useful to an early-stage founder that wants to build a remote workforce.

Disclaimer: It's all subjective, my thoughts, my methods - what has worked for me may not work for you, but I'm pretty confident it will help you.

I'm going to split up the topics into four categories, which I think are the most important when you have remote teams:

1. **Onboarding**: it's always hard to start at a new company, much so for remote workers. I've had situations where our team members left because they were not appropriately onboarded and felt lost.

2. **Day-to-day business:** We'll discuss methods that make life easier when you don't see your employees every day. Small things that may seem unimportant but have a huge impact.

3. **Culture:** There's no way around it. To make sure people want to stay with you - you need to keep them happy and engaged.

4. **Results:** I'm assuming you're not a charity and need something done, rather than have people work for you just for the sake of working. This chapter is where we'll be making sure that the team delivers and pushes the company forward.

Onboarding

Imagine your first day at the office - you come in, everyone smiles, you get introduced to people, brand-new faces, new coffee machine - by the end of the day, you feel pleasantly exhausted from all this information. On the other hand, imagine a remote developer who starts at a company with poor onboarding - same place - he's still in his living room with his computer - no introduction, just a 10-minute meeting to talk about the project. You feel exhausted and lost - what should you do, where should you start. What's happening? Let's try to avoid such situations, they are unpleasant, and no sane employee will want to stay with you if you treat them like that. We can avoid them simply by preparing for future hires:

1. **Start with implementing proper onboarding processes**. This means having a standard procedure for introducing a person to the whole company and to the team where he will be working.

2. **Assign a mentor/guide** who will be the go-to person for the first few months until the person gets settled in.

3. **Have an employee handbook -** all the stuff that the person might need to know - your principles, work ethic, etc. EVERYTHING needs to be in there.

4. **Have some documentation of the project on which they will work.** Set up a meeting with the product owner to understand the project's vision, then with the team members to dive deep into how they are implementing this vision. What challenges are they facing?

Keep it personal, or at least make sure that the system is in place to make the person feel welcomed. Engage with him, guide him through his first few days. For example, at mindnow, our digital agency (BTW, check out the cool cases that we built), we have "Your first week" and "Your first month" articles in our knowledge base - basically, a step-by-step guide on adapting to the new environment and what you need to know, etc. Learn from your new hires - the onboarding process doesn't stay constant; it improves with every new employee, who brings in something of his own and enhances the knowledge base through his own experience. For example, maybe she thinks there are not enough meetings with the team members to understand the project or this project is too big to get to know in a week. All feedback is essential. I remember I had a software engineer who left the company after only two months - I got devasted. So I dug deep into what went wrong, what did we miss, how can we improve. In the end, this resulted in us making drastic changes to our onboarding, and we never had such incidents ever again.

Every employee that leaves due to flawed processes is a failure on your part as a business owner.

Day-to-day business

Now that we have the person properly onboarded, it's time to think about communications in a remote environment. In an office, you're aware of your surroundings and can quickly call a 5-minute huddle to discuss some important topic. It's practically impossible with remote employees - you don't know if they're deep in their work or if they're already on-call with someone or just went to take a shower. The first thing to do is to set up a system for asynchronous and synchronous communication. For example, for synchronous communication, there should be rules for how you organize meetings:

1. The inviter should communicate a clear agenda and goals as early as possible. People should be able to prepare for the meeting.
2. Only the relevant people should be involved. What's the point of having a person in the meeting who does not bring any value.
3. Someone should take notes and document action items. Everything that is not explicitly defined - can be regarded as forgotten.
4. Send out a summary email after the meeting to keep everyone informed about the decisions.

5. Everyone should always be on time and with a functioning camera. It would be best if you were an example here. If you start showing up a few minutes later, everyone else will begin showing up later also.

6. Stable connection to avoid situations when you're feverish talking about the vision only to hear "I lost you there for a second"

For more asynchronous communication, I would recommend using Slack or Microsoft Teams. It's more of a preference; either is fine. With a remote team - this is where you will be having most of your communications. Async communication is where your employees can get burned out quickly. People work in different schedules, and there might be notifications during the evening or on weekends. For example, a notification for a developer might mean a message from a product owner that says something broke. It's a source of anxiety for some people — set up strict rules when notifications should be disabled and alternative communication methods if something is on fire. Once you have the communications figured out - it's time to switch to a more exciting topic - automation. We need to remove as many nuisances as possible from the daily lives of the developers.

The less time they spend thinking about tests, meetings, code style, etc. - the more time they have to focus on the crucial tasks - architecture, complexity, coffee.

Automate everything, integrate everything, join your different services (HubSpot, Jira, HR Software, Monitoring Software ...) into one big hub. For example, Daily standup via Slack, Jira summary message in Slack, Linters, Continuous deployment with Slack notifications, Automatic Tests on branch push. Automation also means documenting processes, so the developer doesn't have to spend 3 hours to figure out how to do something. Imagine if you don't know how to do something in an office environment - it's much worse when you're alone.

So instead, every complex process needs to be written down as a step-by-step guide (preferably with pictures) and shared with everyone. One solution to the problem when people focus too much on their projects is a Fix-it-Friday. A day when a developer can do anything good for a project/company (if the project is not behind deadline, of course). That means he can forget about the tasks and focus on refactoring, implementing a new library, discussing architecture, rework the deployment, improve the documentation, add some automation that would benefit the company. Possibilities are endless if you empower people. In the end, the more resources you invest into proper processes, proper automation, proper documentation, the easier it will be for a remote dev to work on your projects.

Culture

Culture is crucial. How you empower employees and trust them will be one of the factors that will impact your team's output. What kind of environment you set up is how your team will perform. If you have an environment where you give your colleagues autonomy and decision-making tools - then your team will thrive without you micromanaging every step of their way. Trust your employees to make good decisions and hire intelligent people. Keep everything transparent - the feedback, the decisions, the mistakes, the praise. It's easy to hide when you're remote, be silent when you need to speak up or forget to praise your peer when you don't see him face to face. Be an example of transparency - write in channels instead of private messages, take ownership of your mistakes, share management decisions with all employees, give praise publicly.

One aspect where the culture suffers is if you have a hybrid team - part of the team is going to the office, part of the team is remote.

A hybrid company tends to put the remote workers at a disadvantage - they get forgotten, decisions made without them, it feels underwhelming for the people who are not part of "it." Invite everyone to company retreats, go somewhere together, if you're present in several countries - let your employees visit different offices. Meeting each other face to face periodically has a considerable effect on morale. Keep track of the mental health of your employees. The line between work and life gets blurred when you're working from home. Make sure your employees know that it's OK to turn off notifications. It's OK not to respond immediately in the evening. Let them recharge during the off-time.

Discipline

Your team is motivated, and they feel your trust - how are we going to make sure that your goodwill doesn't get abused? We need a way to make sure the things that need to finish on time - really get done on time.

First of all - keep an eye on the results - you must distinguish your perceived performance of the developers and his actual results. For example - a developer might have good presentation skills and sweet talk while underdelivering on the promises. His perceived performance might be adequate, but it might not represent their actual performance.

HOW ARE YOU DOING ON YOUR UNSPOKEN OBJECTIVES?
MY WHAT?
I'M REFERRING TO THE GOALS I HAVE IN MY MIND THAT I'VE NEVER MENTIONED.
HOW ARE THOSE GOING?
I'M TOTALLY NAILING THEM.
Dilbert.com DilbertCartoonist@gmail.com
5-29-12 © 2012 Scott Adams, Inc./Dist. by Universal Uclick

The solution is NOT Time tracking - which is only relevant during the first few months of a developer when you're getting to know each other. But better make your KPIs transparent and explicit.

For example, if it's a Jira Task, define the acceptance criteria instead of a vague description. On the other hand, if it's a product owner - KPIs like the percentage of the closed sprint, team velocity should be in focus.

There's always something granular that you can measure, but don't overdo it because people tend to optimize themselves towards the KPIs. For example, developers would write lengthy code if you would have measured your developers by lines of code (which is weird, don't do that). One of the ways I found to control my productivity from home is a method called timeboxing. I recommend it to everyone. Create timed entries in your calendar - blocks of 2-, 4-, or 6- hours where you only do the task. No emails, no meetings, no nothing. Just the task at hand - it keeps you focused, and you don't waste time.

Why not go remote

Remote work is not a silver bullet. Of course, it has its benefits - you don't have to commute, you spend less time talking about the weather with your coworkers. But then again - those aspects are also important. During the commute, you can listen to a podcast. During your work, you can recharge by talking about sports with your coworkers. It's all part of the experience, and remote work takes that away.

1. There is no Pair-Programming, no easy exchange of expertise between peers. Usually, you can have a quick chat about a new framework in the office - it's much more challenging when you're remote. However, a good alternative would be to have a channel where everyone can write everything - it's like a neverending stream of talking, office noise.

2. Employees miss that feeling of belonging. It's hard to replicate, but we can come to a similar atmosphere if we spend enough resources organizing face-to-face events.

In the end, I must say that times are changing, and giving your developers the possibility to work from home is not a perk now; it's a necessity. We're moving from paying people for the time they spend working for you to paying them for the value they bring you. Trust your team, set up a system that empowers them, and reap the fruits of their motivation.