educative
enterprise

# Developer Onboarding: *Building a Scalable Framework*

*Workbook, Templates, and Checklists included*

We've all seen the cycle and learned our lessons the hard way. Developers are hired, they're ramped up, then a couple months later, they move on to "greener pastures." With "The Great Resignation", the tech industry saw a 4.5% increase[1] in resignations during 2021 along. It's a frustrating and costly loop; each developer can cost upwards of $20,000 to $35,000[2] to become production-ready not to mention the incalculable amount of time consumed across your team.

What is driving developer resignation? How do you break this vicious cycle? What are the biggest keys to retaining your developers? How do you scale your team when you invest so much time and capital into the hiring loop?

One place to start is **developer onboarding.**

Why does great developer onboarding matter, you might ask? In short, developer onboarding holds an intimate relationship with retention, job satisfaction, and productivity. With thousands of startups created every year and large tech companies expanding, developer shortage is at an all-time high (and growing). Keeping developers on your team—while rapidly onboarding new developers—can make the difference between your product scaling or failing to keep up in today's competitive market.

The secret is out that **engineering enablement** is key to developer retention.

According to a 2021 Stack Overflow poll[4], opportunities to learn is one of the top three reasons developers consider leaving their current job. With the constant development of new technology and languages, learning is an inseparable part of an engineer's life. Without learning opportunities, your developers' careers get stuck in a loop of work with no career progression. Every developer is well aware that working alone doesn't get them a promotion. It's knowledge, new skills, and responsibilities that help progress their careers. The first step to creating a work culture around developer learning starts with developer onboarding.

Whether you are trying to grow your team or hire remote developers, having a scalable onboarding process can let you focus on what matters most: shipping great products, running a high-performing team, and growing your developers. Whereas developers may still leave no matter how much they love your company, and successful hiring is mainly dictated by the quality of the candidates in the pool, you hold all the keys to developing and optimizing a successful onboarding process.

[1] https://javascript.plainenglish.io/why-are-so-many-software-developers-quitting-their-jobs-e5a6c2a8f5ed

[2] https://devskiller.com/true-cost-of-recruiting-a-developer-infographic/

[3] https://www.forbes.com/sites/forbestechcouncil/2021/06/08/is-there-a-developer-shortage-yes-but-the-problem-is-more-complicated-than-it-looks/?sh=664d2be3b8e0

[4] https://stackoverflow.blog/2021/12/07/new-data-what-developers-look-for-in-future-job-opportunities/

There is no "one-size-fits-all" onboarding solution, but there are many tricks of the trade and best practices worth taking into account. We place all of these best practices into the bucket of **developer acceleration.**

What does developer acceleration mean, you might ask? Developer acceleration is the culmination of developer onboarding, upskilling, reskilling, mentorship, coffee chats, and more that brings your new developer up to speed with your team's experienced developers. Better yet, it's about creating the best combination of practices to get your developers to grow and stay on your team.

As for now, we'll focus on pushing the lever behind developer onboarding. Keep in mind, every company's tech stack looks different, so we'll cover some general technical onboarding practices. For the most part, however, we'll focus on the *non-technical* aspects of onboarding. Keep in mind that the non-technical side of onboarding is potentially even MORE important to a developer having a great experience at your company. The reason why? Because in the race to ramp up new developers to max productivity, these are the qualities that tend to get overlooked.

We hope that this e-book can offer a practical foundation for strengthening your own onboarding practice, no matter the size of your company or team. Use the **onboarding checklist** at the end, and feel free to tailor any resources here to fit your specific needs.

Your situation might look different than that of other engineering managers. But if you're like most, you're balancing strict deadlines, stakeholder requests, multiple teams, decision-making, and more. When a new developer is slow to ramp up, or moves to a different company, that means missed launch goals, upset stakeholders, and lost revenue.

That being said, doing all of this in a remote environment adds a twist. On one hand, hiring developers outside of your company's location provides more accessible talent. However, with more moving parts and less fluid communication, remote onboarding means there's less room for error. Tools that help promote remote visibility and scalable onboarding in the present set you up for success in the future.

## Personal Reflection

Leading with empathy is always a great place to start, so you might want to start by recalling your own onboarding experience. More likely than not, it's easier to remember bad onboarding experiences more than the good ones. Your new hire is no different.

Before we dive into planning out your developer's onboarding experience, spend some time reflecting on your own experience using the chart below. Keep in mind that positive and negative onboarding experiences shouldn't be limited to personal experience. Recall stories your coworkers or friends have shared, or stories you stumbled upon online.

| Positive experiences | Neutral experiences | Negative experiences |
|---|---|---|
|  |  |  |

Now that you've got your do and don'ts listed out, let's go over some best onboarding practices. It might also be worth noting here that the stakeholders in onboarding include your new team members and your internal teams. Keep these two parties in mind as you build your onboarding process. Here's what we'll cover:

## Table of Contents:

## Preboarding

### *What does success look like?*

You are familiar with onboarding, but you may not be aware that the best onboarding practices start long before your new hire's first day. Building out a structured onboarding plan begins by setting up the new hire for success on day one and setting up easy to follow steps. This is also the time for you to plan out the rest of the onboarding sequence and vision where your new hires fit into the bigger picture. Jump down to 30-60-90 days for a helpful template on laying out your new hire's onboarding experience.

A successful preboarding experience looks something like this: Your new hire has asynchronous contact with someone from your company. They receive easy-to-follow steps preparing them for their first day. Then they can confidently walk in (or log in) on the first day of work. From there, they should know what to expect on their first day, and have access to all the tools needed to go through the onboarding process.

Depending on the size of your company, HR will handle some of these housekeeping items. But it's a great plan to know each step that is required so nothing falls through the cracks. Finally, it's important to remember that your preboarding process should work at scale (that's where templates and checklists come in—more on those later).

*Here's what we'll cover in this section:*

- Background checks/Documents
- Payroll
- SaaS tools
- Equipment
- Office Environment
- Additional Questions
- Email Template

### Background Checks & Documents

This part may be the driest but most important part to getting your new hire ready for day one. Communication between your internal teams and the new hire is key to making sure all documents and background checks are accounted for. Ask yourself who is responsible for getting all the forms and background checks set up for your new hire's first day?

### Payroll

Keep in touch with your internal team responsible for sending the payroll information to your new hire. Make sure they will get their first paycheck promptly.

### Productivity and work tools

Make sure your new hire has all the productivity and work tool logins and access needed for day one. Having all this information sorted on a document will help your new hires ramp up faster and get started on their tasks without extra communication.

### Equipment

Whatever hardware your new hire needs, keep track of when your new hire will receive the equipment. This includes:

- Laptop
- Monitors
- Keyboards
- Mouse
- Cables (Adapters)

### Home Office Environment

For new remote hires, it's important that you acknowledge their remote working conditions before they start working. Giving them a stipend for a desk, a nice chair, a plant, or other office equipment supplies helps your remote worker feel valued and create a productive working environment at home.

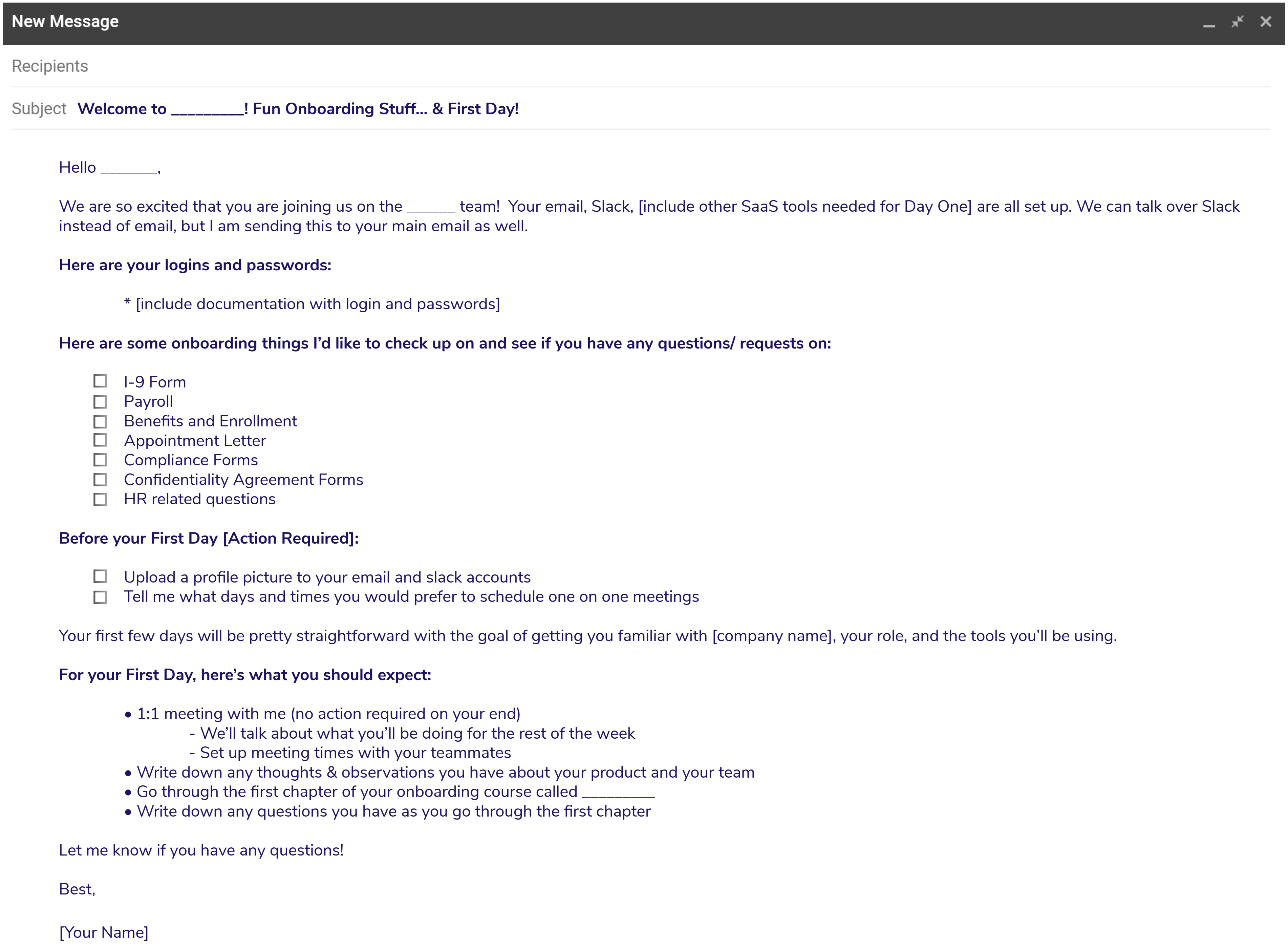
### Additional Questions & Next Steps

Always leave room for feedback and fill in any holes of communication your new hire may have. Paint a picture of what your new hire's first day will look like! Leave no stone unturned and highlight all expectations your new hire should have.

### Sample Email Template

Let's put all of the preboarding steps into a tangible template. Feel free to use or adapt this simple template to make sure your new hire's first day goes smoothly!

# Sample Email Template:



Recipients

Subject **Welcome to _________! Fun Onboarding Stuff... & First Day!**

Hello _______,

We are so excited that you are joining us on the ______ team! Your email, Slack, [include other SaaS tools needed for Day One] are all set up. We can talk over Slack instead of email, but I am sending this to your main email as well.

**Here are your logins and passwords:**

* [include documentation with login and passwords]

**Here are some onboarding things I'd like to check up on and see if you have any questions/ requests on:**

☐ I-9 Form
☐ Payroll
☐ Benefits and Enrollment
☐ Appointment Letter
☐ Compliance Forms
☐ Confidentiality Agreement Forms
☐ HR related questions

**Before your First Day [Action Required]:**

☐ Upload a profile picture to your email and slack accounts
☐ Tell me what days and times you would prefer to schedule one on one meetings

Your first few days will be pretty straightforward with the goal of getting you familiar with [company name], your role, and the tools you'll be using.

**For your First Day, here's what you should expect:**

- 1:1 meeting with me (no action required on your end)
  - We'll talk about what you'll be doing for the rest of the week
  - Set up meeting times with your teammates
- Write down any thoughts & observations you have about your product and your team
- Go through the first chapter of your onboarding course called _________
- Write down any questions you have as you go through the first chapter

Let me know if you have any questions!

Best,

[Your Name]

## The First Day

*What does success look like?*

Ideally, your new hire goes to sleep after their first day thinking “Ah, that was just as expected... I’m looking forward to working here!”. Your new hire should sleep well this first night not feeling overwhelmed and feel like they know what to expect in the next coming days. The last thing you want is for your new hire to jump off the computer after the first day and think “Wow, did I make the right decision working here?”

As for you, a successful first day is when your new hire’s day meets your communicated expectations. Alongside getting the first taste of your company culture and meeting new people, they’ll start to experience the cadence of work and communication. At this point, your new hire logs off for the day understanding what they’re doing for the first week.

*Here’s what we’ll cover in this section:*

- Important Links & Tools
- Mentor/Buddy
- Introductions
- Team bonding
- The First Day Schedule

### Important links & tools

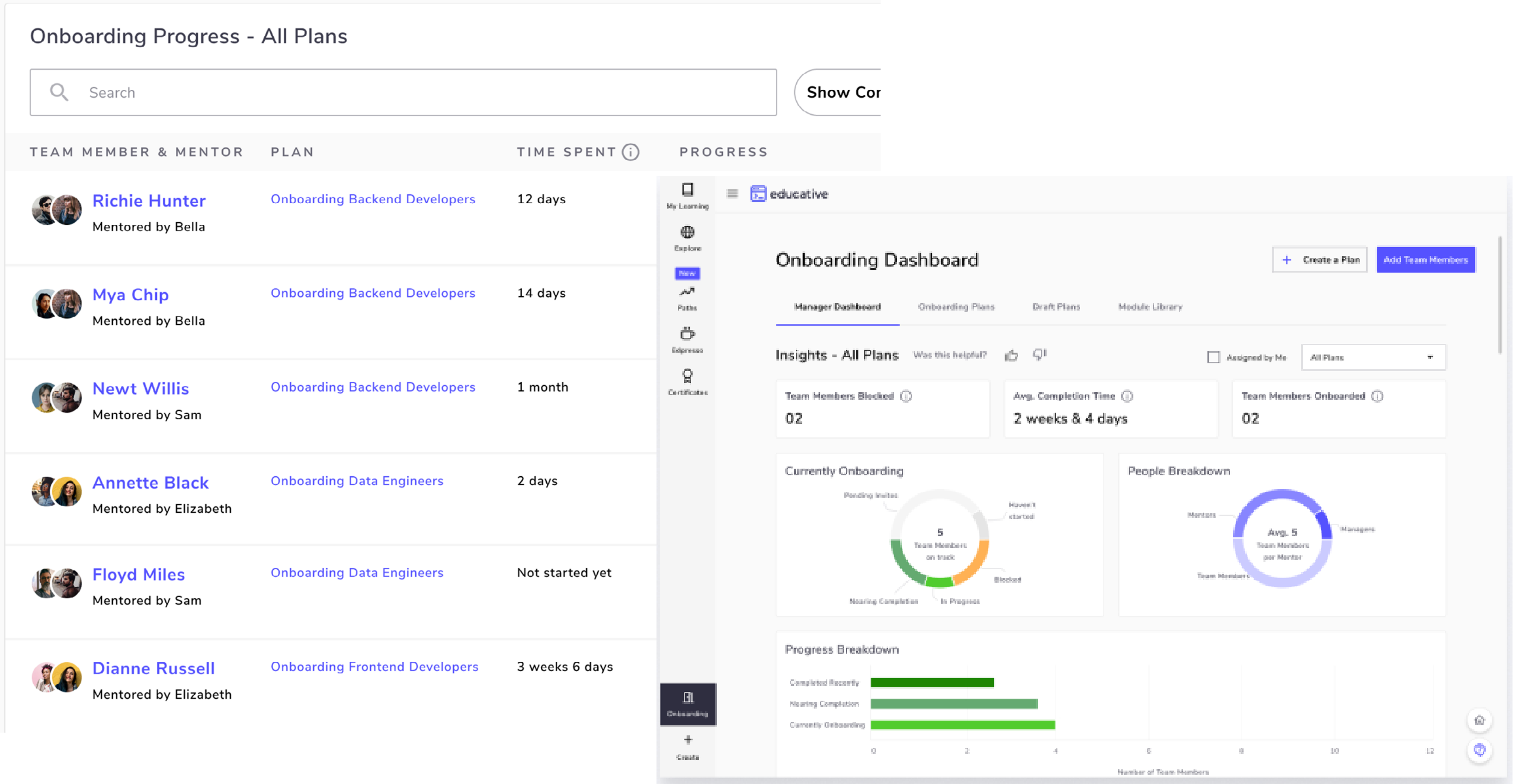
Your new hire should have access to a wiki with helpful links, tools, or documents that helps explain your team’s and company’s internal processes. For example, at Educative, we ask the new hire to set up their laptop password, then provide a page of important links and instructions for the new hire to reference in the future. Here’s a snippet of what our wiki page looks like:

### Mentor & Collaborator

Everybody wants a companion. Or perhaps a better way to put this is “someone you trust”. Assigning your new hire a mentor or collaborator will be invaluable in accelerating your new developer to understand the company culture, product, and internal processes. Most importantly, your new hire won’t feel alone in their onboarding process. Especially in a remote setting, the lack of a mentor or collaborator can make your developer feel like they’re on an island.

Your new hire’s mentor/collaborator will act as their go-to point of contact for technical questions, introductions, and a guiding hand for initial projects. Keep in mind that mentorship shouldn’t always fall on the most senior developer. Due to the risk of burnout for the senior developer, you can consider more junior developers who might benefit from developing leadership skills.

For example, to avoid our mentors from becoming burned out, we use our onboarding dashboard to help mentors asynchronously track a team member's progress and identify any blockers. By automating a progress tracker and also creating a button for team members to "ask for help," we hoped to create an environment where team members always feel supported. At the same time, mentors have a scalable tool to be readily available for multiple team members simultaneously without being distracted.



Onboarding Progress - All Plans

Search

Show Cor

| TEAM MEMBER & MENTOR | PLAN | TIME SPENT | PROGRESS |
|---|---|---|---|
| Richie Hunter<br>Mentored by Bella | Onboarding Backend Developers | 12 days | |
| Mya Chip<br>Mentored by Bella | Onboarding Backend Developers | 14 days | |
| Newt Willis<br>Mentored by Sam | Onboarding Backend Developers | 1 month | |
| Annette Black<br>Mentored by Elizabeth | Onboarding Data Engineers | 2 days | |
| Floyd Miles<br>Mentored by Sam | Onboarding Data Engineers | Not started yet | |
| Dianne Russell<br>Mentored by Elizabeth | Onboarding Frontend Developers | 3 weeks 6 days | |

Onboarding Dashboard

Insights - All Plans

| Team Members Blocked | Avg. Completion Time | Team Members Onboarded |
|---|---|---|
| 02 | 2 weeks & 4 days | 02 |

Currently Onboarding

People Breakdown

Progress Breakdown

*It might be helpful to remind your new hire that you and the people who were part of their interview are all in support of helping them succeed. Fostering a sense of a support network within the company can help your new hire avoid any sense of feeling alone.*

## Introductions

Time to introduce your new hire to the team! During your introductory call with your new hire, highlight various points of contact your new hire will work closely with. For example, here's a list of people you should have your new hire introduce themselves to:

- HR
- IT
- Team members
- Mentor & Collaborator

Have your new hire organize some one-on-ones with some team members and introduce the rest of the team during a daily standup.

*Always leave space for questions and actively encourage your new hire to ask any questions. A reminder that no question is a silly question helps to relieve some initial stress your new hire might be feeling in speaking up during a meeting. As a bonus, you might just pick up some valuable insight from a fresh perspective.*

## Daily standups/scrums

How are your daily standups/scrum sessions run? Introduce your new hire to the daily cadence of standups and answer any questions they might have about this experience. Your new hire gets to experience how your team communicates, and it might be a great way to introduce your new hire to the team and company culture.

Integrating daily standups into the onboarding experience provides a practical opportunity for new hires to hear about what other members of the team work on and any blockers or questions people bring up. After running the meeting, take some time to introduce the new hire, or have them introduce themselves.

## Team bonding

*"Sharing food with another human being is an intimate act that should not be indulged in lightly"* - M. F. K. Fisher

Never underestimate a time of gathering with food (even in a remote setting)! It's true that remote settings make these food gatherings feel less authentic compared to being in-person, but a meal with your team can help foster a more casual environment to build relationships. Send a stipend for UberEats, throw in some icebreaker conversation starters, and initiate introductions for your new hire.

It might seem like a small gesture, but research[1] shows that building stronger relationships can foster a more empathetic approach between individuals which in turn helps to create more efficient and effective communication skills. With remote work in mind, communication builds the backbone of a productive team. Other surveys for remote workers revealed that 10% of the participants noted that informal settings helped relieve stress and motivated them to perform well in their work.

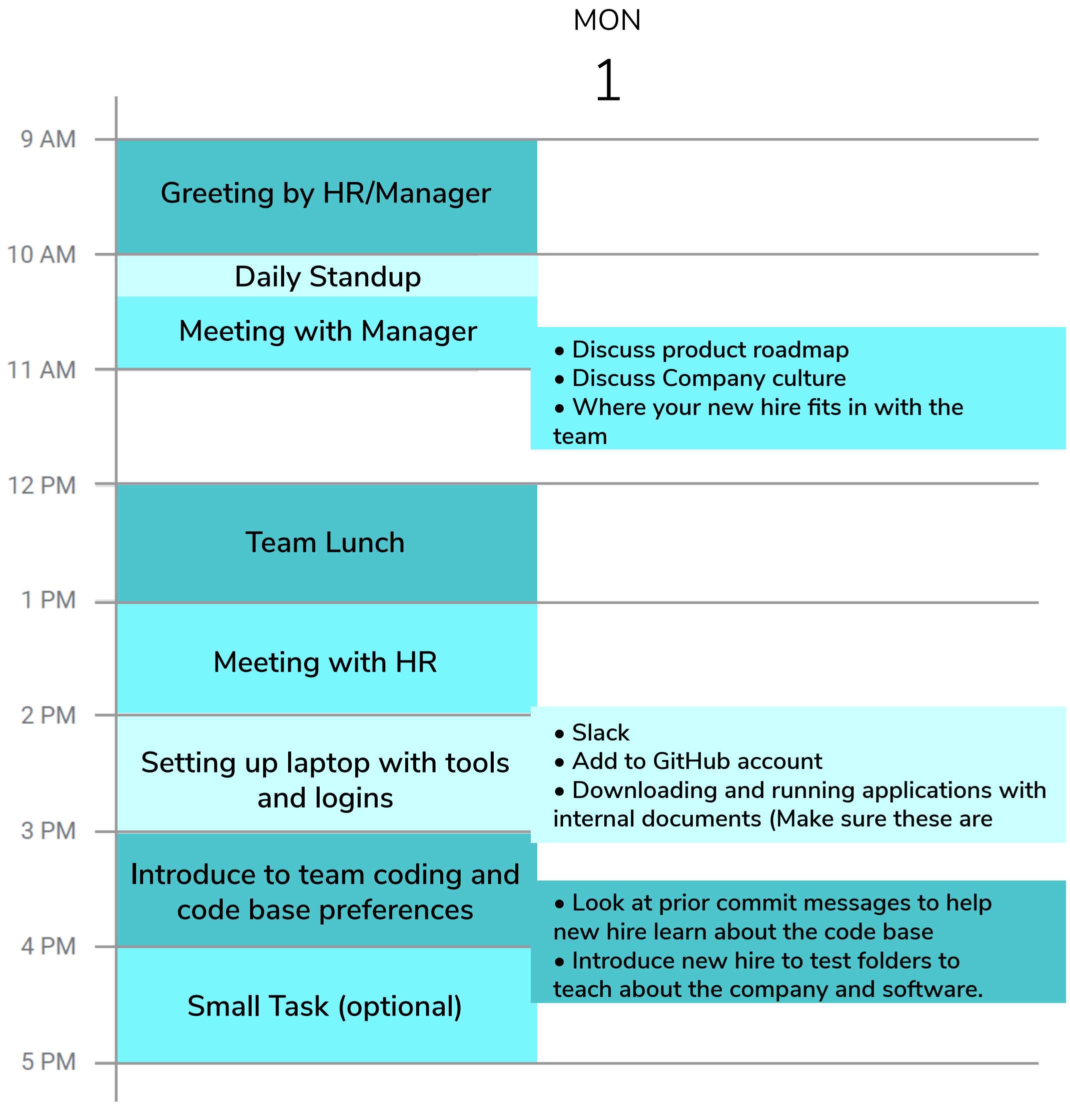
Here are some other creative ideas to get you started:

- Virtual team lunches
- Virtual happy hours
- Virtual cooking classes
- Outdoor gatherings
- Virtual scavenger hunts
- Virtual board games

[1] http://www.conscientiabeam.com/pdf-files/soc/91/JNMMC-2016-3(1)-1-24.pdf

## First Day Sample Calendar

Before we dive into planning out your developer's first week, here's a sample calendar aggregating The First Day.



MON

1

9 AM

Greeting by HR/Manager

10 AM

Daily Standup

Meeting with Manager

- Discuss product roadmap
- Discuss Company culture
- Where your new hire fits in with the team

11 AM

12 PM

Team Lunch

1 PM

Meeting with HR

2 PM

Setting up laptop with tools and logins

- Slack
- Add to GitHub account
- Downloading and running applications with internal documents (Make sure these are

3 PM

Introduce to team coding and code base preferences

- Look at prior commit messages to help new hire learn about the code base
- Introduce new hire to test folders to teach about the company and software.

4 PM

Small Task (optional)

5 PM

## The First Week

Phew, you got your new developer through the first day! It's time to start ramping up your new hire for the rest of the week.

This is all about setting the foundation for upcoming weeks, so let's take a closer look at healthy goals to set for your developer. (This can also provide a useful reference point for future onboarding). Similar to the first day, the first week should highlight clear steps, while providing asynchronous guidance from mentors and senior developers.

While every developer works differently, deliberate feedback and questions will help your new hire ramp up quickly without feeling overwhelmed.

*Here are some goals to set for your new hire:*

□ Learn programming best practices for your team/company
□ Start ramping up technology and language knowledge
□ Learn more about the product and projects
□ Meet with team members and managers
□ Set expectations and goals

### Learn programming best practices for your team/company

Update your wiki by highlighting good programming practices and have your new developer look through any documentation or resources that best reflect your company's preferred procedures. Namely, being explicit about your team's Git workflow can help your new developer ramp up more quickly and not leave your developer just to figure everything out themselves.

Here are some good programming practices we use for Educative's teams:

- Use proper naming conventions
- Follow the DRY Principle
- Avoid logging sensitive information
- Test coverage
- Write bug-free code
- SEO and accessibility score
- Performance
- Right code placement
- Get sign off on one and six pagers
- Do not make large pull requests
- Add proper pull request descriptions

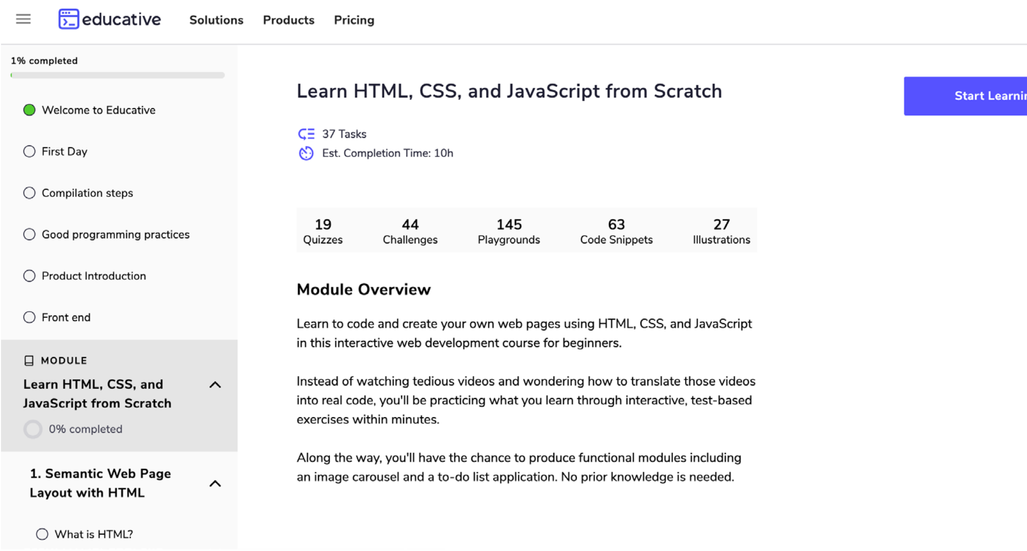
### Start ramping up knowledge in technology and languages

You saw potential in your new hire and it's time for them to learn or review the technical skills required to become production-ready. This is where you'll want to help provide your developer with resources, documents, videos, and tutorials that are readily available even beyond onboarding. The easier it is for your developer to access high-quality learning materials, the faster your developer will fill their skill gaps and confidently do their work.

Above all, find interactive and practical ways for your developer to practice coding as they learn. Like learning to ride a bike or learning to swim, everyone knows you won't learn until you try. Assign small tasks for your developer to work on throughout the week. There's no better way for your developer to learn your company's code review process and production deployment than to ship a small feature or fix a small issue.

At Educative, we onboard our developers by pairing meaningful courses with hands-on and personalized learning opportunities. By helping our new developers set up their dev environment quickly, we then provide instructional courses with hands-on coding environments to fill in any skill gaps. It's a multi-step process where we first Identify skills gaps and then encourage mentoring and coaching with interactive coursework to master their coding skills.

We split instruction into two major modules between front-end and back-end and typically have our new hires start our front-end course in the first week and give about three weeks to finish both front-end and back-end courses. Of course, all of this is paired with a mentor to help answer any questions or provide more context. It's all about tweaking the many knobs for developer acceleration.



1% completed

● Welcome to Educative
○ First Day
○ Compilation steps
○ Good programming practices
○ Product Introduction
○ Front end

MODULE
Learn HTML, CSS, and JavaScript from Scratch
0% completed

1. Semantic Web Page Layout with HTML
○ What is HTML?

Learn HTML, CSS, and JavaScript from Scratch

37 Tasks
Est. Completion Time: 10h

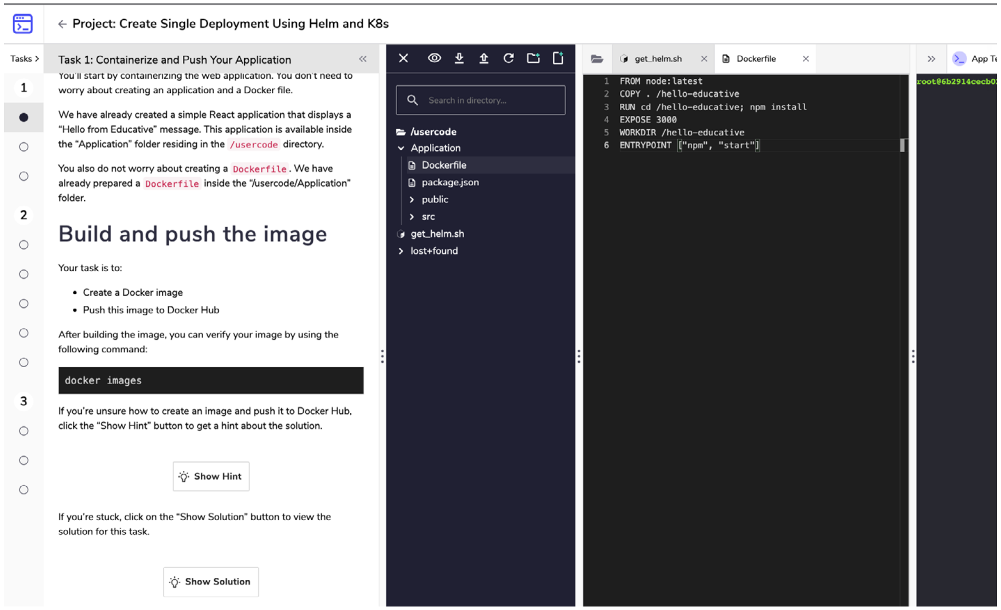
| | | | | |
|---|---|---|---|---|
| 19 | 44 | 145 | 63 | 27 |
| Quizzes | Challenges | Playgrounds | Code Snippets | Illustrations |

Module Overview

Learn to code and create your own web pages using HTML, CSS, and JavaScript in this interactive web development course for beginners.

Instead of watching tedious videos and wondering how to translate those videos into real code, you'll be practicing what you learn through interactive, test-based exercises within minutes.

Along the way, you'll have the chance to produce functional modules including an image carousel and a to-do list application. No prior knowledge is needed.



Task 1: Containerize and Push Your Application

You'll start by containerizing the web application. You don't need to worry about creating an application and a Docker file.

We have already created a simple React application that displays a "Hello from Educative" message. This application is available inside the "Application" folder residing in the `/usercode` directory.

You also do not worry about creating a `Dockerfile`. We have already prepared a `Dockerfile` inside the "/usercode/Application" folder.

Build and push the image

Your task is to:

- Create a Docker image
- Push this image to Docker Hub

After building the image, you can verify your image by using the following command:

```
docker images
```

If you're unsure how to create an image and push it to Docker Hub, click the "Show Hint" button to get a hint about the solution.

Show Hint

If you're stuck, click on the "Show Solution" button to view the solution for this task.

Show Solution

```
FROM node:latest
COPY . /hello-educative
RUN cd /hello-educative; npm install
EXPOSE 3000
WORKDIR /hello-educative
ENTRYPOINT ["npm", "start"]
```

## Learn more about the product and projects

Help your developer get the lay of the land. We recommend you take a multi-faceted approach to help your new hire get comfortable with the codebase and problems you're solving on the team. Look to provide your developer with:

- Small tasks for small wins
- Documentation on product roadmaps and vision
- Full access to the company wiki
- Helpful links and resources
- Time to read through past pull requests

## Meet with team members and managers

This might be a good place to remind you that your developer's first week can feel exhausting. They're juggling learning new technologies, processes, and procedures, while constantly meeting new people. Don't feel like your dev has to meet everyone on the team in one week. Help your dev manage their meetings, so that they can stay focused on learning and executing their tasks.

Keep in mind that it can take on average 23 minutes to refocus after becoming distracted by another task. Keeping big windows of time for your developer to stay focused on their task helps improve overall productivity. We recommend front-loading meetings in the day, so that your developer can use the rest of the day to focus on their tasks.

## Set expectations and goals

Some people thrive off of surprises. For most, predictability helps avoid unnecessary stress. That being said, the first week is a good time to set expectations for your developer. Getting on the same page early mitigates misunderstandings in the future and helps your new developer keep track of their own progress.

*Here's a list of software engineering work practices to get you started:*

- **Do not assume:** *If you're unclear on implementation, write out a one-pager, and submit for approval*
- **30-minute rule:** *Reach out quickly after spending 30 minutes tackling the problem*
- **Be proactive in code reviews:** *improves code quality and code base knowledge*
- **Done is better than perfect:** *Write high-quality code, but don't waste time perfecting*
- **Always work on a specific JIRA task:** *Log time against it*

In addition, this might be a good time to help communicate your goals for your new developer and set up the 30-60-90 days ahead.

***Ask yourself the following questions to help set goals for your developer:***

What tasks do you expect your developer to complete in 30 days? 60 days? 90 days? Beyond?

What small tasks can help your developer experience small wins in the next coming days?

What coding and code ownership goals do you have for your developer?

What leadership and teamwork goals do you have?

What are some expectations you have for your team?

What are the qualities of your high-performing developers?

## 30-60-90 Days

Your developer's toes are officially wet. Now it's time to get them comfortable swimming by themselves. After your new hire's first week, they should have their dev environment set up and be ready to take on the next set of tasks. If there were some tasks that didn't fit in the first week, move them into the following weeks.

### First 30 Days

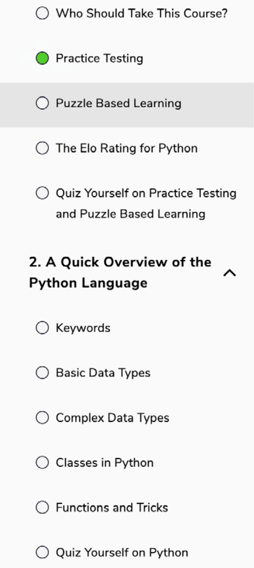
Your developer is still getting to know your company, customer, and product. The first 30 days should be spent by:

1. *Learning*
2. *Ramping up small tasks to projects*
3. *Establish routine*
4. *Getting to know the team*
5. *Setting career goals*

### Learning

With an emphasis on learning, the first couple of weeks should be dedicated to helping your developer refresh on or learn new technical skills. At Educative, we dedicate the first 3-4 weeks to helping our devs learn about the technology and languages we use at our company. We believe that by investing early in our engineers, we grow their career and our product simultaneously.

For more experienced developers, our first couple of weeks provide an opportunity to dive into expert material on cutting-edge technology and trends. This continuous focus on personalized learning keeps our engineers motivated and engaged in their work and career.



○ Who Should Take This Course?

● Practice Testing

○ Puzzle Based Learning

○ The Elo Rating for Python

○ Quiz Yourself on Practice Testing and Puzzle Based Learning

**2. A Quick Overview of the Python Language**

○ Keywords

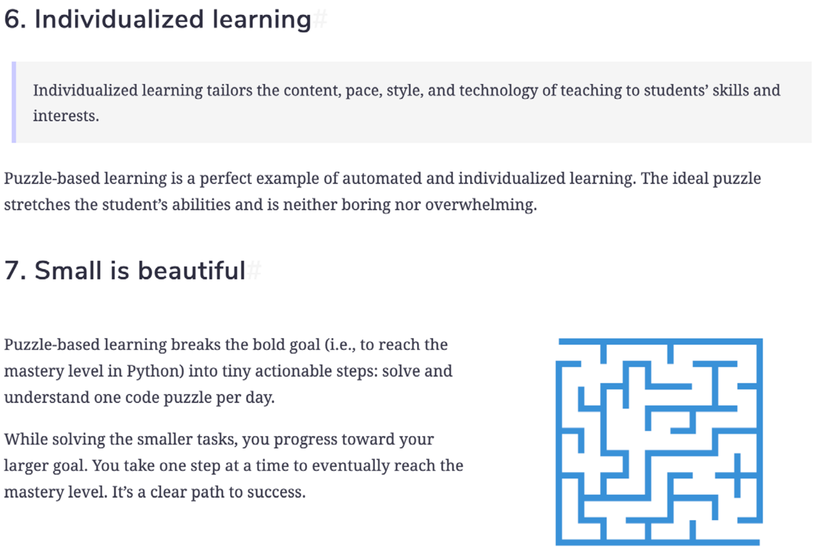
○ Basic Data Types

○ Complex Data Types

○ Classes in Python

○ Functions and Tricks

○ Quiz Yourself on Python



### 6. Individualized learning

> Individualized learning tailors the content, pace, style, and technology of teaching to students' skills and interests.

Puzzle-based learning is a perfect example of automated and individualized learning. The ideal puzzle stretches the student's abilities and is neither boring nor overwhelming.

### 7. Small is beautiful

Puzzle-based learning breaks the bold goal (i.e., to reach the mastery level in Python) into tiny actionable steps: solve and understand one code puzzle per day.

While solving the smaller tasks, you progress toward your larger goal. You take one step at a time to eventually reach the mastery level. It's a clear path to success.

### Ramping up small tasks to projects

To support your new hire's learning, consider assigning them small tasks and projects to help them experience your team's typical workflow before implementation. Having your developer sit in on code reviews can also help them better understand your codebase and what high-quality code looks like.

### Establish routine

This time period also helps your developer get into their working groove. Particularly for remote workers, this might mean which hours they feel most productive, learning when and how to communicate with the rest of the team, and setting up automations for their workflows. You'll establish a cadence of one-on-ones with your new hire to deliver feedback, answer questions, and develop rapport.

### Getting to know the team

Intro meetings between cross-disciplinary teams should continue to happen throughout the first weeks. This will come in handy later as your developer starts to work with cross-disciplinary teams, such as Product and Design.

### Setting career goals

Get serious about your developer's personal and professional goals. Goals help keep your developer on track to becoming production-ready, but also help your new hire grow in their career. At Educative we look at three major areas that are then broken down into trackable attributes.

| Technical Skills | Delivery | Strategic Impact |
|---|---|---|
| • Debugging and observability<br>• Quality & testing<br>• Software design and architecture<br>• Product support<br>• Security | • Incremental value delivery<br>• Self-organization | • Business acumen & strategy |

## 60-90 Days

Here’s where things get a little fuzzy. Every company has different goals; every developer has different levels of experience; every manager has different expectations. There is no one-size-fits-all approach at this phase.

With any goal, it’s important to consider the following characteristics of your developer:

### Starting point

Every developer starts their first day of work at a different place. Some might be coming off a nine-month rigorous bootcamp and others have ten years of experience under their belt. Your goals should match the career direction and step where your developer enters your company.

### Pace

Regular goals and deadlines provide your developer with a measurable working pace. The pace at which tasks should be completed at this phase should never overwhelm your developer.

## Goal Categories

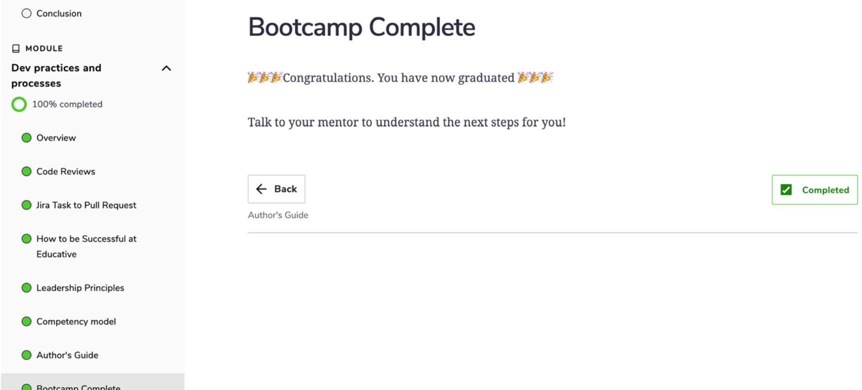
Next up, consider the three following categories for how you plan to set your goals. Each category comes with its own set of questions to help you to reflect and create goals.

*1. Learning*
*What are your developer’s skill gaps?*
*How will they fill their skill gaps?*
*What system of accountability do you have set up?*

*2. Performance*
*What performance goals best suit your developer’s experience?*
*How will they meet these goals? In what amount of time?*
*What expectations do you set for new developers?*

*3. Personal*
*What career goals does your developer have?*
*How will they meet these goals?*
*What are their interests?*

Now, let’s put your learnings to practice by using the 30-60-90 Goal Setting Template on the next page.



Bootcamp Complete

🎉🎉🎉Congratulations. You have now graduated 🎉🎉🎉

Talk to your mentor to understand the next steps for you!

# Developer Onboarding:

*30-60-90 Goal setting template*

## Learning Goals

30 days

•
•
•

60 days

•
•
•

90 days

•
•
•

## Performance Goals

30 days

•
•
•

60 days

•
•
•

90 days

•
•
•

## Personal Goals

30 days

•
•
•

60 days

•
•
•

90 days

•
•
•

## Learning Tasks

30 days

•
•
•

60 days

•
•
•

90 days

•
•
•

## Performance Tasks

30 days

•
•
•

60 days

•
•
•

90 days

•
•
•

## Personal Tasks

30 days

•
•
•

60 days

•
•
•

90 days

•
•
•

### Preboarding:

- □ I-9 Form
- □ Payroll
- □ Benefits and Enrollment
- □ Appointment Letter
- □ Compliance Forms
- □ Confidentiality Agreement Forms
- □ HR related questions
- □ Handle Equipment
    - □ Laptop
    - □ Office Supplies
    - □ Stipend for desk/chair/lamp/etc
- □ Set up accounts
    - □ Email
    - □ Slack
    - □ GitHub
    - □ Internal services login
- □ Prepping for first day
    - □ Send zoom meeting invites
- □ Assign mentor

### Onboarding: First Day

- □ Walkthrough Wiki doc including important links and tools
- □ Introduce developer to team and what they're working on
- □ Introduce developer to company resources (HR, IT)
- □ Introduce developer to daily standup
- □ Arrange a team lunch (remote or in-person)
    - □ Icebreaker questions or games (Guide to staying connected)
    - □ Stipend for food

### Onboarding: First Week

- □ Learn good programming practices
    - □ Update internal wiki
    - □ Share document on best coding practices
    - □ Communicate Git workflow
- □ Start ramping up technology and language knowledge
    - □ Assign resources, courses, or links for languages
    - □ Aggregate list of technologies for your developer to learn
- □ Learn more about the product and projects
    - □ Schedule meeting to answer questions, discuss product and product roadmap
    - □ Aggregate list of resources for your developer to access
- □ Meet with team members and managers
    - □ Schedule one-on-one cadence with you and your new hire
    - □ Assign task to new hire to schedule meetings with team members
- □ Set expectations and goals
    - □ Update list of work expectations
- □ Aggregate list of small tasks and projects for new hire

## Final Thoughts

Rapidly scaling our engineering team at Educative has been a tremendous learning experience. Going from a team of one to ten to now more than 60 developers in just a few short years came with its own unique challenges. We're still working out the kinks, but we wanted to share some of our best tools and practices for developer acceleration that helped grow and scale our engineering team to what it is now.

We're still learning and improving how to onboard developers at scale. Along the way, we've learned that having the right tools and resources in place was critical to overcoming hurdles more quickly and efficiently. Likewise, we hope you'll be able to face your own challenges using some of our learnings.

*Happy learning!*