The Ultimate Guide to Hiring Software Developers

Everything you need to know about growing your software development teams—both onsite and remote.

STX NEXT
python powerhouse

Table of Contents

Introduction

Who *doesn't* need software engineers these days?

Startups are hunting for fresh talent to get their ideas off the ground, while larger companies are always hungry for more programmers to outrun the competition in the tech race.

There's just one problem: **hiring skilled developers who can bring real value to your team is a notoriously daunting task.** After all, there's just *so much* to consider, *so many* causes for concern.

You can't simply jump into recruiting without proper preparation; learning why hiring developers is challenging, how to find developers, and how to get developers to find you is the key to a solid foundation for recruitment.

Then, the recruitment process itself has multiple stages, each of them distinct and complex, presenting plenty of pitfalls and potential missteps.

Also, in order to even have a recruitment process in the first place, you need recruiters, and you need to make sure they're the right people for the job.

Last but not least, there's the issue of knowing what makes a good developer and whether a candidate is a good fit for you and your company.

Are we close? Does any of this sound familiar? Is it keeping you up at night?

If so—good. Hate to break it to you, but **you're right to be worried.**

Here are some stats, taken from the fourth edition of the Candidate Experience 2017 eRecruiter survey:

50% of candidates

will share their recruitment experience with their friends from the business and advise them not to apply for a job at your company.

39% of candidates

claim the biggest room for improvement is in how well you explain the reasons behind turning them down.

33% of candidates

with a negative recruitment experience will never again apply for a job at your company.

Scary, we know. But fear not! We're here to help.

We wrote this ebook because we understand that **a sound recruitment process is absolutely essential for your company's well-being.** And it's in your best interest to understand this, as well.

Read on, and you will learn how to set up a successful hiring process and why it's so important to make sure it runs like a well-oiled machine every step of the way.

Welcome to your ultimate guide to hiring software developers. Let's begin.

What Should You Consider Before Kicking Off The Recruitment Process?

There is one fundamental question you have to ask yourself before you even start thinking about recruiting:

“Do I need to hire another developer in the first place?”

This may seem a bit self-evident, but it really isn’t. You’d be surprised how often developers get hired **without a well-defined purpose,** just to “expand the team.”

Don’t make that mistake. You have to know *exactly* why another developer is necessary, otherwise you end up wasting their valuable time. In order to establish whether you do need to add a new member to your team, ask yourself something else:

“Who do I need? What kind of skill set do they need to possess?”

This depends on what projects you have in the works. Are they missing something? Are you falling behind schedule? Are there any areas in need of improvement? If so, well, there you have it—that’s where you should allocate more human resources.

The easiest way to learn what is missing or could be improved is to **talk to your developers and get feedback directly from them.** Your current team is also your best window into diagnosing the possible symptoms of work falling behind schedule, which may include unfinished tickets at the end of a development sprint, low team velocity, or failure to meet deadlines.

Having identified your needs, you can **set clear and specific expectations** for your developers right from the start:

What projects will they be working on?

What will their responsibilities be?

What role will they serve on the team?

That last question in particular is crucial. Skills are one thing, but sometimes you may require someone to fill a specific role that's missing from your team. Beyond needing just another software developer—which is a very valid reason in itself—it usually goes one of three ways:

1) You need a specialist.

You may be missing a specialist who fits a certain niche with their unique expertise and know-how. A developer experienced with a particular framework, such as Flask or Django. Someone well-versed in React Native, so you can build a cutting-edge mobile app. A machine learning expert, if you're getting into new technologies. Those are but a few examples. As your projects change, so do your personnel needs. Adapt accordingly.

2) You need a leader.

You may have all of your technical requirements met, but still lack a team member with the right personality to lead the others and make sure the work runs smoothly. A Scrum Master is someone you always want to have on your team, but an arguably greater necessity is an individual who is a developer themselves and understands all the ins and outs of both coding and the technical requirements of a particular project.

3) You need a breath of fresh air.

You may have assembled a full team of highly skilled professionals, whose only additional need is someone to bring something new to the table. This can benefit the team in all sorts of ways: increased motivation, improved atmosphere, reshuffling of responsibilities, teaching and learning from one another, approaching potential problems from a different angle—you name it.

Once you've addressed all of these preliminary concerns, the time is ripe to consider another burning issue with hiring software engineers…

Why Are Developers So Hard To Find?

The greatest problem with hiring software engineers is **competition.**

The bigger the city you're in, the fiercer the fight to recruit the best. You might think that a larger market provides a healthier stream of developers, but in reality it's almost impossible to hire good developers in places like London or Silicon Valley.

Top talent gets sucked up by the major players in the blink of an eye, leaving you straining to find anyone to work on your project.

Hiring software engineers is especially tricky for companies without a well-established, prestigious brand. If you fall under this category, you may find that developers either don't want to join you or will leave you the minute the project gets boring for them.

Moreover, the market is skewed toward the developers themselves, who can pick and choose the companies who will provide them with the best benefits and compensation.

All is not lost, though.

While the competition may be fierce and high-impact developers ever harder to find, the task is not impossible. We'll show you how to get started on recruitment and face the many challenges it presents.

How To Start Recruiting Software Developers?

When you start looking for developers, you have to decide between two courses of action:

1) Hire a headhunting agency.

A potentially effective solution, though a costly one.

Say you do decide to hire a headhunting agency from either the UK or the US to meet your recruitment needs. Do you know how much it's going to cost you?

Investing such amounts in new hires could possibly pay off in the future, but the initial cost will delay the moment you start seeing a return on your investment.

Thankfully, there's another option.

2) Recruit by yourself.

The alternative to headhunters is to take the recruitment effort upon yourself. It is a significant challenge, but depending on the size of your company, this solution may prove more cost-effective.

One of the issues with "insourcing" your software engineer recruitment is that **you may not have the right skillset onboard to find good hires.** For example, your HR department may find it difficult to identify and attract quality candidates.

Another potential obstacle could be your project managers trying to tackle the task themselves. They may know their way around software development, but are they competent enough as recruiters to get the job done?

At STX Next, we believe wholeheartedly in taking recruitment into your own hands.

In fact, we believe it so strongly that the challenges we've mentioned above inspired us to write this ebook and share the best practices that, in our collective experience, lead to valuable software development hires.

Having said that, let's take a closer look at what you can do to give yourself a fighting chance in the competitive business of hiring developers. There are two ways to go about it: **outbound and inbound.**

Outbound Methods Of Recruitment

Is it possible to create an effective process for recruiting software engineers that brings valuable candidates at a reliable pace?

Yes, it is. Here's how to make it happen.

In this section of the ebook, we're going to discuss **outbound methods of recruitment.** Using the **outbound** approach, your strategy is to **find viable candidates for your software engineering positions and entice them to apply for a job at your company.** The idea is to reach out and pluck talent from the job market.

This stands in opposition to the **inbound** approach, where **the developers find you and come to you in order to work at your company**. You can read all about it in the next section.

Outbound recruitment certainly is a tall order, but there are proven tactics you can use to increase the odds of swaying developers to your side, ranging from classic LinkedIn messages to more creative methods like workshops and hackathons.

1) Direct search via LinkedIn and other portals

The typical way of reaching potential software development hires is to send them a message through LinkedIn or a similar medium. **This method is common for a reason—it's fundamental to your recruitment success.** For STX Next, this is still the top channel for finding new developers.

The key difference between success and failure here is the way you compose your first message, or your **outreach.** A good first contact message reflects the best practices found in the best sales emails; it must be interesting and effective. Ask yourself:

"What can I include in the message to make it impossible to ignore?"

Here are a couple of handy outreach tips to get you started:

a) Include the salary range in your message.

This is a must, since it has become an industry standard by now.

b) Include the location of your job posting.

Keep in mind that most developers aren't looking to relocate and you will have to make your deal even sweeter to get them to do it.

c) Tell the candidate about the technologies they'll be working with.

Any bleeding edge technologies you may be using could turn the tide in your favor and convince the developer to choose you.

d) End the message with one, crystal-clear call to action.

This could be either "email us," "visit our website," or "schedule a call." Again, the next step should be clear as day; **don't force the candidate to think.**

If there are other aspects of your offer you'd like to highlight in the outreach, here's one simple rule to follow: **choose the benefits that separate you from the competition.** Those are your key selling points and you should point them out first. If your company has great

employee benefits, add a few words about them. If it's a modern and exciting project, make sure the candidates are made aware of that.

However, make sure that the content of your outreach is all killer and no filler. **The purpose of this message is to impress the candidate, not inform them of all the details**. That comes later.

2) Recruitment through sharing knowledge: conferences, workshops, and hackathons

A less straightforward, but nonetheless effective, recruitment method is sharing your organization's tech knowledge at conferences, workshops, and hackathons.

It's a win-win situation: you give the event attendants a chance to educate themselves, while at the same time gaining an opportunity to grow your network and expose new people to your organization.

The first issue to take into account here is **location, location, location. Think small and local.** You have a much higher chance of success if you address a more focused, more motivated group of candidates during a smaller event. Attending larger events may raise awareness and prestige, but the audience may come from all over the continent. There's not a lot of hope that you will find someone who fits your location.

Second, keep in mind that **this recruitment tactic requires technical knowledge.** You may turn to your CTO to help you here. Their combination of technical and business savvy gives them the right skillset to ensure an interesting conference presentation or practical workshop content.

Whenever you organize or participate in an event, you should think in terms of a **recruitment funnel.** You need enough people attending the event (filling the top of the funnel) to sift out those that will be either uninterested or unqualified, and get the gems you can hire.

Let's dig a little deeper into how you can ensure success for each type of event.

3) Effective conference presentations

When it comes to making an impact during a conference presentation, we have two key nuggets of knowledge to share with you.

First of all: **don't be selfish!**

When preparing your content, you should ask yourself:

"**How will the audience benefit from attending my presentation?"**

Keep in mind that this is very different from looking into how *you* will benefit from doing the presentation. That should *not* be your main focus; the attendants will catch on to it.

Our second tip is much more specific and relates to live coding at conferences. Again, we'll give it to you straight: **just don't do it.**

Live demos during a presentation create a lot of problems for little to no benefit. Even if it's smooth, it takes up way too much of your precious time; the audience is just watching you type. Additionally, it's incredibly easy to make a mistake in your code, and then end up wasting even more time on debugging, hunting for that one mistyped line as your authority crumbles.

For those reasons, it's much better to **show ready-made code snippets and keep the show going smoothly.**

Depending on your target audience, you should also choose appropriate content. **Juniors** seek basic knowledge. They don't want lectures or speeches; they're mostly looking for knowledge akin to a programming course. **Advanced attendants** will be more interested in case studies of interesting projects. Make sure to present something concrete and focused.

The matter of junior vs. advanced attendants also comes into play when you're looking to organize a workshop.

4) Hosting workshops: junior vs. advanced

Organizing a practical workshop yourself can be even more effective than attending a conference with your presentation. The downside, of course, is that it requires much more work and preparation involving many people in your organization. You are the host here, after all.

The most pivotal decision to make is whether to gear your event toward juniors or advanced developers. It's easier to conduct workshops for junior attendees—but do you want to recruit junior developers? Ask yourself this question in advance, or you'll end up with an unproductive event.

On the other hand, you can conduct advanced workshops. Keep in mind, however, that those are much more time-consuming to put together. You could spend many hours of work—even a whole month—preparing to pull off just one short event.

Moreover, you'll need to bring in experienced speakers and your audience will be harder to convince—both to attend the workshops and to take an interest in your job offers.

Another issue to tackle with junior workshops is **how you filter potential attendants.** Usually your workshops will require at least a minimum of starting knowledge, even if they're geared toward relative beginners.

To process the influx of submissions and get worthwhile attendants, you need to establish an **automated skills test.** Anything else will create too much work to process the event smoothly.

As an example, the Has Power workshops hosted by STX Next include a short, four-question quiz. The attendants are invited based on their quiz answers and their LinkedIn profile. The need for careful selection is very real for us, since we usually have two or three people for each spot at workshops such as Python Has Power.

5) Hackathons: managing expectations

Another idea that has been gaining prominence among software houses and tech companies is **organizing hackathons to attract programming talent.**

The idea of a hackathon is quite simple. You book a venue. You order food and beverages for your attendants (beer is absolutely fine!). You invite potential candidates to participate in a fun coding project. Finally, you try to hire the ones who stand out the most.

Hackathons work especially great if you're a startup. You can even establish a theme for your hackathon, such as VR, IoT, AI, Big Data—anything that's exciting and fits your startup profile. Invite developers to come, code something, show off their idea, and get prizes for the best projects. This way you can pre-select the developers who have skills you could use.

Keep in mind, however, that while hackathons can be easier to organize than workshops, they are also more expensive. Without an appropriate budget for food, drinks, and prizes, you might fail to meet expectations.

Remember, **your hackathon must be fun!** These events are all about creativity, so leave plenty of room for the attendants to run with their ideas. Don't impose stiff limitations.

Oh, and one last thing: **don't expect people to come and create amazing software you can then sell.** The event shouldn't support your product; it should bolster your recruitment. We've seen hackathons that were all about "cheap coding labor"—not fun.

Summary

Outbound methods of recruitment are a complex subject, and an equally exhaustive inside look at the inbound approach awaits you in the next section of the ebook.

Bearing that in mind, here's a short recap of what we've covered so far:

LinkedIn

- Your best bet to find good developers is still via direct LinkedIn outreach.
- A good outreach should be impressive and interesting. Put emphasis on your key advantages over the competition.
- Information such as salary, location, tech stack, and the next action to take are a must.

Conference presentations

- Prepare basic, course-like knowledge for junior attendants and case studies for advanced guests.
- Think first about how the audience—not you—will benefit from seeing your presentation.
- Don't do live coding—at best you will waste time; at worst you might lose authority.

Workshops

- Keep in mind that a good workshop can take up to a month to organize properly.
- You should organize junior workshops only if you want junior developers.
- Make sure to pre-filter your attendants using an automated skills test.

Hackathons

- Include funds for food, drinks, and prizes for the attendants in your budget.
- Feel free to give your hackathon a theme that fits your profile to attract the right programmers.
- Don't treat the hackathon as a source of free labor to build your product.

Now that we're all caught up on the outbound approach, let's move on to **inbound methods of recruitment**—because you can't go running after every candidate and hitting them over the head with your offer. Once you scale up, such practices become unsustainable.

Inbound Methods Of Recruitment

It's no revelation that people have different means of achieving their goals. This is as true for job-seeking as it is for anything else.

It stands to reason, then, that even if you exhaust all of the outbound options we've gone over in the previous section, there will still be developers who never get hired at your company if you limit yourself only to outbound methods.

After all, some people prefer to take matters into their own hands. They want to find the absolute best job for them through their own efforts. Maybe they feel that this way they can get a better offer than if they're just "reactive" and responding to outbound outreach.

And that's all okay. What you need to do is **make sure you're out there for the candidates to find.** Once you have a sound system of inbound recruitment opportunities in place, the developers should come to you and ask to join your team.

Below you will find four methods STX Next uses to ensure a strong presence on the job market and a steady stream of passive job submissions.

1) The classic approach: job postings on job portals

As before, we'll start with the classic, perhaps obvious, method first: **listing jobs on job portals.**

Also as before, this tactic is included in the list for a reason. **Job postings can be as effective as ever, even for developer positions**—if you put in the effort to prepare them correctly.

Multiple factors affect the effectiveness of your job posting, especially when you're targeting developers. We've compiled some key guidelines for you:

- **Always include information about the salary, segregated by seniority if possible.**
- **List the technologies the developer will be working with.**
- **Highlight the perks and benefits of employment at your company.**
- **Consider separating your requirements into "must have" and "nice to have" categories.**
- **Outline your recruitment process to set clear expectations right from the start.**
- **Encourage the reader to contact your HR department and ask questions in case of confusion.**
- **Do not gloss over crucial details—only communicate the truth.**
- **Update the job posting based on feedback or questions during the interview.**
- **Lastly, use numbered or bulleted lists to make the content easier to digest! (See what we did there?)**

Once you have the content of your job offer ready, it's time to post it around the web and wait for inbound candidates. If you're looking for inspiration, here's a short list of job portals to post your software engineer openings on:

- Indeed,
- Monster,
- LinkedIn Jobs,
- AngelList,
- Dice,
- Glassdoor.

Even after you've covered the portals above, stay on the lookout for new ones where you can promote your offers. Niche websites may get less traffic from potential candidates, but at the same time you're likely to encounter less competition there.

2) The Careers page on your website

If someone hears about your company, **they will visit your website.** That's especially true if they'd like to work for you. It's simple common sense. I mean, would you even want to hire someone who hasn't bothered to google you? Didn't think so.

This is why you should assume that every single person you recruit will go through your website, or at the very least your Careers page.

To avoid losing their interest at this stage, **make sure you avoid two common pitfalls:**

a) Not having a Careers page in the first place

Some companies, especially startups, take a peculiar approach to recruitment. Believe it or not, they assume all they have to do is put up a vague teaser page showing their awesome company culture and developers will flock to join them.

Sure, an interesting main page may pique the visitors' interest, but taking on a new job is a major life decision. Candidates want a clearly labeled place where they can get cold, hard information about potential job positions. And if they want to get such information, it's your task to provide it—with a **dedicated Careers page.**

b) Having a huge Careers page the developers can't easily scan

Of course, there's the other side of the coin. Large companies tend to have a multitude of job openings at any point in time. As such, their Careers pages tend to be filled with numerous job positions, most of which have nothing to do with software development. The IT gets lost in the crowd.

Try to empathize with a potential candidate. You visit your company's website. Do you feel like sifting through all the content just to find that one position labeled "Python Developer"? Or do you keep looking elsewhere, perhaps in search of a company that can communicate their needs more clearly? Exactly.

Luckily, there's a simple fix: **create a dedicated landing page on your website specifically for IT positions,** or at least the key IT jobs you want to fill. Then you can direct traffic from other pages, such as your main page or social media posts, to this dedicated landing page. No more digging through irrelevant postings.

3) Posting job offers on social media: LinkedIn, Facebook, Twitter

If you're anything like us, you're spending quite a bit of time and energy building a following on social channels such as LinkedIn, Facebook, and Twitter. It's entirely reasonable that you use that following not only to attract new clients, but also new hires, including software developers.

Once again, the trick is in the execution. **A well-structured post can mean the difference between a candidate clicking with curiosity or scrolling down past your offer.**

To grab their attention, your offer should follow some of the same rules as the outreach we've discussed in the outbound section. **The offer must be brief and include key selling points:** salary, work requirements, technologies used, benefits, location, etc.

As a general rule of thumb, **visual content works best,** so presenting your offer using images instead of just text will help you stand out. If possible, use a consistent visual theme for your software developer job openings—see our example below.

Note: the salaries here may be outdated. Visit our Careers page for current salaries.

Posting on your company's Facebook fanpage or LinkedIn company page is just the tip of the iceberg, though; **it's not nearly enough.** To really start getting results, you need to go deeper and take your search one step further.

A great opportunity here are **groups for jobseekers.** Look for groups that fit your tech stack, such as "JavaScript Jobs" or "Python Jobseekers," to identify an audience that will certainly pay attention when you share your job posting.

Speaking from experience, **Facebook groups** have been the most effective at getting STX Next inbound developer candidates. Here are two groups you can start with: Jobs in IT and Python Developers.

4) Ad campaigns

The final tactic to consider for your inbound recruitment are **paid campaigns using Facebook and Adwords.** You will need to set aside a budget for such operations, of course, but a well-profiled campaign can be very effective.

As an example, STX Next entered the Gdańsk market a while back, after establishing our new STX Next office in the Polish Tricity. With a small expense of around 150 PLN (€35/$42) per campaign, we created ads aimed specifically at Python developers in the region, aiding our recruitment efforts.

Bear in mind: **this is not a "last but not least" entry on the list.** Ad campaigns *will* cost you, especially if they're poorly targeted. You should only use them to support the other operations.

Think of the ad campaign as a closing mechanism for on-the-fence candidates. A large success factor in inbound operations is having multiple points of contact. You want to have many channels for potential applicants to hear about your company.

Once your candidates realize that the paid job ad they're seeing isn't coming from a completely unknown employer—but one they've already discovered on Facebook, LinkedIn, or Twitter—they will be much more likely to click and apply.

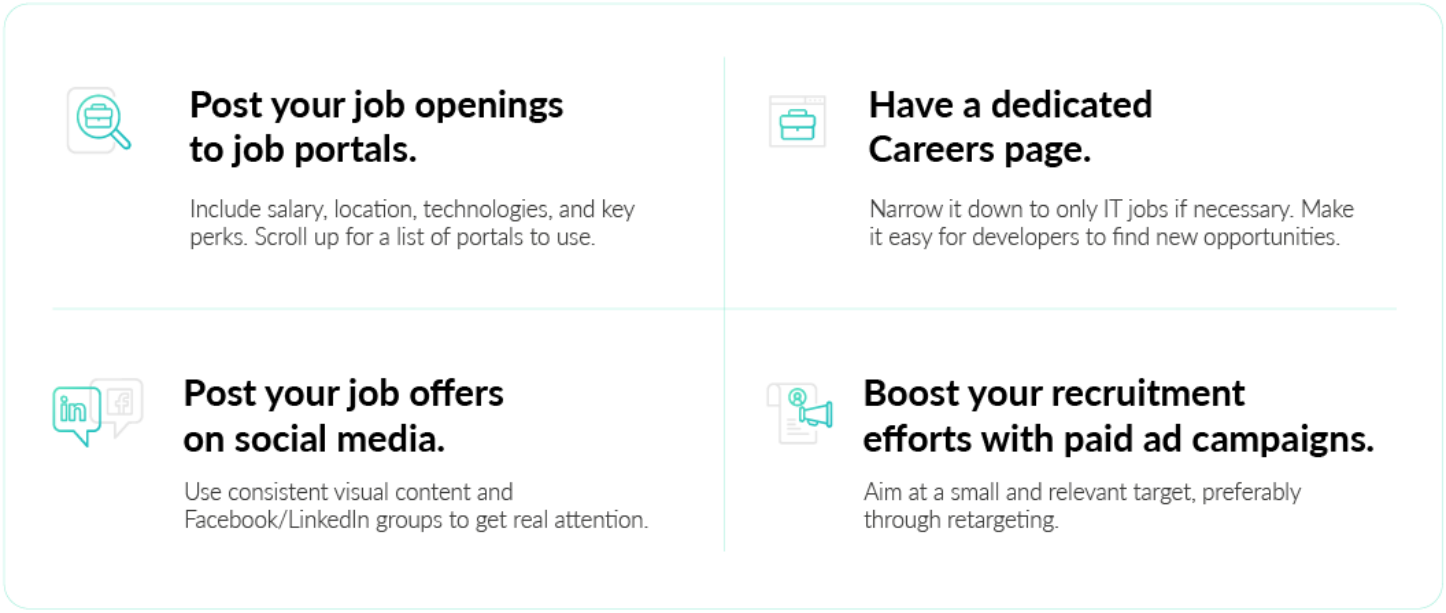
Using ads in synergy with other operations can lead to excellent results; using ads in isolation is a recipe for financial waste.

Another successful tactic is **retargeting (also known as remarketing).** The idea is to display your ads to the people who have already visited your website and give them an extra push to finally take action.

Such campaigns can be quite cheap and directed toward a very small group with a high chance of success. Once again, you're looking at paid campaigns to work as a closing mechanism, the cherry on top of your prior efforts.

Summary

Hoping for a CliffsNotes version of what we've covered? Look no further! Here are the essential takeaways from our four steps to a thriving inbound funnel for recruiting software engineers:



Of course, your hiring journey doesn't end here. Far from it, really—we've only just begun.

Read on to learn all the other ins and outs of recruiting. A big part of those is everything you need to take care of once your candidates' CVs start flooding your company inbox. Don't forget you still have five of the six stages of the recruitment process ahead of you.

Yes, we have only covered the first stage in the last two sections.

What can we say? Hiring software engineers is no joke.

Without further ado, let's move on.

Five Fundamental Dos And Dont's Of Recruiting

It's relatively common knowledge that a lot of people on the job market genuinely *hate* going through recruitment. Ever wondered why that is the case? If you have, you're in the right place.

Before we move on to an in-depth discussion of the six stages of the recruitment process and how to avoid failing at each of them, let's shed some light on the most common complaints candidates have about recruiting:

1) Too much to do

Don't make recruiting unnecessarily hard on your candidates. If it's too strenuous, a nuisance, or has too many steps, you will discourage them from putting their best foot forward before you even hire them. If you make your process effortless, your candidates will thank you for it.

2) Wasting time

If you value your applicants' time, you value them as people—now, *that* is a good start to any relationship. This goes both ways, too; their time is precious, but so is yours. Don't take too long. Be efficient, plan ahead, make the process as swift as possible.

3) Homework

Candidates resent being forced to go through a bunch of useless, pointless motions just because. If there's one thing people absolutely can't stand, it's getting bogged down in bureaucracy. Ask only as much as you really need from them, and they will happily oblige.

Plus, it bodes quite poorly for your future cooperation if you swamp them with a million things to do this early in the process.

4) Vagueness

Be transparent, not vague. Keeping your candidates in the dark isn't doing you or them any favors. Provide them with a complete set of information from the get-go, be upfront about your expectations, and deal in specifics only.

The salary formula, tech stack, or work experience are but a few examples where being honest and straightforward is in both yours and your candidate's best interest.

5) Poor feedback

"Poor" doesn't mean "negative," mind you. Positive or negative, it's the *quality* of your feedback that matters. Give it **always**, give it **well,** and give it **fast**—or not at all. Point out specific reasons that led to the ultimate decision to sign a contract or not.

Both sides stand to gain a lot from giving and receiving feedback, but only if it's done properly. Stay in touch, set a date, then stick to it.

Keep these fundamental concerns in mind when figuring out your recruitment process. Address them beforehand, and both you and your candidates will greatly benefit from it.

Even the people you don't end up hiring will then likely leave on a somewhat positive note, with a sense of clarity and closure, instead of feeling sour and bitter. This, in turn, should produce favorable word of mouth for your company, which is the greatest silver lining you could hope for under the circumstances.

Six Stages Of The Recruitment Process

It comes as no surprise that adding new members to your development teams requires putting in place a carefully constructed system; that's why it's called a recruitment *process*. The process varies from company to company, but it's always made up of several steps designed to be followed closely.

Your recruitment process speaks volumes of your company and plays a big part in your reputation on the job market. You stand to gain or lose more than you may think depending on how you conduct your recruiting.

We suggest a recruitment process made up of six stages to ensure that the experience is as efficient and beneficial as possible for *all* parties involved.

1) Job offers

First things first: you need to make it known that you're looking for fresh developers. At this stage, two methods of recruitment apply—**outbound and inbound**—as we have discussed earlier on in this ebook.

To recap, suffice it to say that outbound recruitment means **you find the developers,** while inbound recruitment means **the developers find you.**

Common mistakes: job offers

A common mistake at this stage is **mismatching the offer to the candidate and not including enough information about the position.** Candidates should have a clear picture of your tech stack, your projects, and your team composition. By this point you should have identified the strengths and weaknesses of your team; here is where you look for solutions to your potential problems.

Most importantly, **after reading the job offer, your candidates should know precisely what your expectations and requirements are,** so that they can decide whether submitting their application is even worth their time—or yours, for that matter.

2) Collecting CVs

Once your job offer is out, people will (hopefully) start turning their applications in. If your posting is attractive enough and you promote the opening properly, you should expect significant traffic with many candidates fighting for the job.

Common mistakes: collecting CVs

The biggest mistake you can make at this stage is having a **leaky system for collecting the CVs.** This essentially means "losing" a candidate's CV and never even responding to their application. Most of the time, you only learn that this is happening long after the fact, when it's too late and the damage is already done.

It's a huge blunder and very bad practice, resulting in highly negative word of mouth from the candidates you mistreat like that. Not to mention you potentially losing valuable team members for good due to simple sloppiness and negligence, since they're not likely to want to work for you ever again.

Contacting *all* the candidates is essential—even the most ridiculous, absurd cases. Even if they are a very poor fit for the job, each candidate has taken the time out of their busy schedule to reach out to you in hopes of professional cooperation. The least they deserve for that is a rejection with an explanation and thanks.

Your failure to provide each candidate with this bare minimum will cause justifiable anger and bitterness, but it goes even further than that; **it gives your whole company a bad name,** making you look incompetent and careless. Naturally, that's the last thing you want people to associate your business with.

So to prevent such unpleasantries from happening, **make sure you have a tight, fail-safe system for CV collection, and that you reply to each and every application.**

3) Screening

At this stage you screen your candidates, meaning you go over their CVs and LinkedIn profiles, and decide whether you want to move on to the next stage of the recruitment process with them.

Screening is advantageous because it verifies whether a candidate checks all of the most important boxes for the job of a software engineer, such as:

- technical skills,
- professional background,
- salary expectations,
- knowledge of English,
- willingness to work on a B2B basis (if your company uses this business model with developers).

If a candidate doesn't meet those basic requirements, you can safely turn them down already at this stage. It saves you, your HR recruiters, and your IT recruiters precious time later on.

However, if you find a candidate who is to your liking, this is when you schedule an appointment for the interview with your HR department.

Common mistakes: screening

A couple of mistakes sometimes happen here:

a) **Setting up the meeting for the next day after reaching out to your candidate and not giving them any room to move it.**

This is simply inconsiderate of your applicants' time. You have no way of knowing what their current situation is, professionally or personally. As such, it's possible that they simply won't be able to adjust to a schedule this tight on such short notice.

Being inflexible about this may cost you a potentially valuable developer. It also gives your company an unfavorable appearance of not caring about your employees' well-being too much.

On top of that, it suggests poor management skills on your part, if you're so desperate and hard-pressed to hire someone new.

b) **Promising contact and never delivering.**

You can't have it both ways; either you reject an applicant during screening (and explain why!) or you show interest to continue talking, promise contact, and **follow up.**

It's completely unacceptable to respond saying you'll "get back to them soon" and then never do. They have taken the time to talk to you and you have already given them hope

that you're at the very least considering interviewing them for the job. Leaving the candidates hanging at this point is unprofessional and disrespectful, showing no regard for them or their time.

4) The HR interview

The main purpose of the HR interview is to **get rid of the candidates who are clearly not right for the job.** They may have checked all the boxes at the CV-level, but actually sitting down with a person face-to-face and hearing them answer your recruiters' specific questions is something else entirely.

Since conducting interviews is centered around coming into contact with new people, the greatest quality you want in your recruiters is people skills, chief among those being empathy.

Both your recruiters and your candidates are living, breathing individuals with thoughts and feelings of their own. A truism, to be sure, but one that plays very strongly into human interaction, also in the professional context.

Though the setting is formal and serious, there should always be room left for a hint of partnership and cooperation**,** even this early in the process. After all, the shared interest is undeniable: one side wants to hire, the other wants to get hired.

Common mistakes: the HR interview

At the HR interview stage, a couple of mistakes can also be observed frequently:

a) **The recruiters are ill-informed about the position, the company, or the software engineering community.**

We'll be delving deeper into what makes a good recruiter in the next section of the ebook, but for now suffice it to say that they need to be competent**.** This means both asking the right questions and giving the right answers to any questions the interviewees may have.

From the perspective of the applicants, **your recruiters are the face of your company.** As such, they need to have the basic know-how to represent your brand in a worthy manner.

b) **The recruiters have no control over the interview.**

When interviewing candidates, recruiters need to be in charge of the situation. It's important not to create a hostile environment—this is an *interview*, not an *interrogation*,

after all—but don't lose track of the fact that **it is the recruiter who dictates the terms of the arrangement,** no matter how friendly or inviting they may otherwise appear.

Make sure your recruiters come prepared and on time. The questions they ask need to be matter-of-fact and to-the-point. If it helps them, using extra materials is not a crime, but they can't be reading from them all the time. A general aura of authority and professionalism is in order for the interview to go well.

Side note: if a candidate has over a year of commercial experience, it's common practice to conduct the HR interview by phone, then follow up with a longer IT interview should the candidate pass this stage.

5) The IT interview

For the purpose of hiring developers, having professionals with the technical know-how conduct this stage of the recruitment process **is a must.** While basic involvement and interest in the software engineering world should be expected of your HR department, they do not have the knowledge necessary to test the specific competences of your applicants. This is why this stage of the process exists in the first place.

First, your IT recruiters get information from the HR recruiters about the candidates who have passed the HR interview. Then, they sit down with an applicant and describe to them in greater detail the ins and outs of working at your company. They discuss past and present projects, the tech stack, the challenges that come with the job, etc.

The bulk of the IT interview consists of **test tasks evaluating the technical skills and competences of the candidate.** Regardless of the programming language your company specializes in, **the test tasks should reflect the actual, everyday work** the candidates would be expected to do at your company. Abstractions don't do anybody any favors.

Common mistakes: the IT interview

The greatest mistake you can make at this stage is **having the wrong people conduct the IT interview.** Granted, they need to be excellent coders in order to properly assess the candidates, but even more so they need to be social. Programmers or not, this is still a person talking to a person; **the ability to express yourself and show your feelings** goes a long way.

Once again, it all boils down to **being prepared—both professionally and personally.** The candidates may need help from your IT recruiters, and the recruiters should be able to provide that help.

Rudeness or unpleasantness help nobody, and will only have negative effects later on, whether a candidate gets the job or not. Just like with the HR interview, it's empathy above all else.

6) Feedback

When all is said and done, it is time to make the decision whether you hire a candidate or turn them down. What comes with that is the last stage of recruitment, often neglected, yet absolutely crucial: feedback.

Regardless of the choice you make about the hire, **giving feedback is equally important.**

If you make the hire—tell your candidate why, what they did well, what they could've done better.

If you *don't* make the hire—tell your candidate why, what they did well, what they could've done better.

***Always* give feedback.** And when you do, make it specific and to the point, don't beat around the bush. Talk about both the good and the bad. Be critical, yet constructive. Look back on their past, but also ahead into their future.

Don't forget that **positive feedback is just as valuable as negative feedback.** While pointing to potential weaknesses and shortcomings may seem like a priority, appreciating your candidate and commending them on what they bring to the table is a great morale boost and makes them feel valued from the start.

Your HR recruiters and your IT recruiters should also exchange feedback among themselves to make each individual recruitment process better than the last.

Lastly, **get feedback about your recruitment process from the candidates themselves.** Ask them what they liked and didn't like about it and why. Encourage them to be transparent and honest while they're at it. Their opinion is the one that counts the most.

It all comes full circle. The work never ends, and there is always room to get better. Adapt the principle of **continuous improvement; give feedback and get feedback all around.** It will pay dividends in more ways than you may expect.

Common mistakes: feedback

Unsurprisingly, the most common mistake made at the final stage of recruitment is *not* giving feedback or giving insufficient feedback. The insight your recruiters can give the candidates into their performance during recruiting is priceless, especially if you don't hire them.

Recruiters are often professional educators for candidates who wish to get into a new line of work or climb their career ladder, but are not yet in a position to successfully do either. Explaining at length and in detail what they're missing and what would need to happen for you to hire them is the greatest favor you can grant them—and one you also stand to gain a lot from in the long run.

If a candidate you turn down has a positive recruiting experience at your company and gets helpful feedback from your recruiters, chances are they will feel motivated to improve their skills and try to apply for a job at your company again once they get better.

Imagine their satisfaction and enthusiasm if their second attempt to work with you is successful. Also, imagine the merits of adding such a developer to your team.

That being said, you and your company will also benefit from giving **equally thorough and comprehensive feedback to the candidates you do end up hiring,** as well. This lays good groundwork for your future cooperation and provides a solid foundation for effective communication down the line.

Five Qualities Your Recruiters Need To Recruit Great Software Developers

At the end of the day, **the success or failure of your recruitment process depends on the people doing the recruiting.**

You can prepare the stages of the process with extreme precision and scrutiny. You can read guides like this one and formulate your unique vision of how recruitment should work. You can spend a lot of time and effort planning ahead to leave no stone unturned before you get down to business.

None of it will matter if your recruiters are not up to the task. Who they are personally and how they conduct themselves professionally translates directly into your candidates' experience with your company.

Which is why the people recruiting your new developers should possess the following qualities:

1) Professional

You know what they say about first impressions, right?

Your recruiters are the first point of contact between you and your applicants. They speak for your entire company and represent everything your brand stands for. For the people you turn down, your recruiters will be the extent of their experience with your company.

That is why it is absolutely crucial that you choose the people doing your recruiting very, very carefully. **How your candidates see them is how your candidates will see you.**

Above all else, this means that your recruiters need to be completely professional. “Well-informed” is a bit of an understatement; they really should know the basics of everything there is to know about your company. Any question an interviewee may have, they should be ready to answer. Preparedness is essential.

2) Organized

Think of organization as the “technical side” of being professional.

Punctuality is one of the big components here. If your recruiters are late for a scheduled appointment, or barely make it at the last minute, it casts a proportional shadow of poor time management on your company. They should be there ahead of time, waiting for the interviewee to come, with everything already in place.

Being organized also means **being in control of the interview.** Have the questions prepared beforehand. Make sure your questions are the right ones. Ask the questions in a sensible order that agrees with the flow of the conversation. All of this speaks volumes of your recruiters’ competence and strongly informs how the interview is going to go.

If your recruiter needs to use notes or a laptop during the interview, it’s their job to make sure this doesn’t break the flow of the meeting. Such **extra materials are acceptable only if they facilitate the meeting, not disrupt it.** Also, it’s bad practice to rely too heavily on notes, since it makes your recruiters look unprofessional.

3) Empathetic

Empathy is generally one of the most wonderful qualities a person can have. It can also be extremely useful in the workplace, especially if it’s someone working closely with other people, or whose job requires them to constantly meet new people. Like, say, a recruiter.

Recruiters benefit from being empathetic in a variety of ways, but never more so than when an interviewee is a particularly poor fit for the job. Politely turning someone down can sometimes be a tremendous challenge, and that’s when empathy is needed the most.

The reason why empathy is so important is because **you have no idea who the person sitting in front of you is.** You don’t know their life story, where they’re coming from, or what they’re going through, and all of that can influence their performance during the interview. Professional competences or lack thereof may very well be just the tip of the iceberg.

A recruiter needs to be understanding. People are people, and circumstances change everything.

4) Respectful

Similarly to professionalism and organization, a parallel may be drawn between empathy and respect—with subtle distinctions.

The key thing to remember is **we're all human.** We fail. We have bad days. Things don't always go our way. Sometimes this affects our day-to-day lives in ways we can't anticipate. It can also make us perform miserably during a job interview.

None of that means we don't deserve to be respected.

Let's assume a job interview does go horribly wrong. For whatever reason, the candidate is tragically underqualified. There is exactly zero chance of your recruiters changing their minds about him or her.

The appropriate response is to calmly, but decisively, turn them down—and then offer help. Give them recommendations, point them to the areas most in need of improvement, suggest workshops or other forms of honing their skills. **The value of individual feedback cannot be overstated. Provide it *always*.**

Who knows, maybe down the line they will reach out again, this time coming from a much different place? Time will tell, but even if this is the last you'll see of each other, **they will always remember being treated like a human being. And likely pass the good word on.**

5) Involved

This is something that often gets ignored, though it really shouldn't: **your recruiters need to be involved in the software engineering community.** Hosting and attending workshops and hackathons, following influential figures within the community on social media, or staying up to speed on the latest and greatest in the tech world are all great.

Granted, the technical side of recruitment belongs chiefly to the developers interviewing your candidates during the IT interview, but your HR department also has to be familiar with the fundamentals of IT.

No matter what a recruiter's exact role at your company is, they are recruiting developers, and as such should be prepared to answer questions that can specifically come from developers applying for a software engineering position.

Even if said questions won't be extremely detailed or technical in nature, **developers have their own community culture,** so it would be awkward if your recruiters had no interest in this culture or knowledge of it.

Every company is more than the sum of its individual parts, and each employee should act like an integral piece of a larger whole. While falling short in this regard may be forgiven in the case of employees who don't have a strictly representative function, recruiters must represent.

Soft Skills Vs. Hard Skills

In this section, we'll switch gears from recruiters to candidates.

Before we explain why the distinction between soft and hard skills matters, let's start with the basics: what are they?

Soft skills are a **loose collection of personality traits** relating to how you conduct yourself around other people. It's an umbrella term grouping together a wide array of characteristics, including work ethic, creative thinking, social graces, personal habits, and the like—all referring to the relationships with one's colleagues in this context.

Hard skills, on the other hand, are **quantifiable, acquired abilities** that require specific knowledge and training to perform certain tasks. They need to be learned and can be clearly defined, making them essentially the formal requirements your candidate needs to meet in order to get hired.

The two types of skills often go hand in hand, but never more so than in the workplace. True, your developers are there because they have a job to do, but they're also people who share the same space for a number of hours every day.

How well your developers get along with one another greatly influences the quality of their work, leading us to the key dilemma:

Which skills are more important?

A question frequently neglected, since the answer may appear obvious on the surface: of course it's the hard skills, right? They're the reason you hired somebody in the first place, what your developers need to know to do their jobs, why wouldn't they be your top priority?

You may be surprised to learn that the seemingly obvious answer is actually the wrong one. Turns out, **soft skills are more important than hard skills.** Yes, really.

Of course you can't hire someone for their soft skills alone; they need to be able to do their job and do it well—preferably very well.

But strong hard skills simply aren't enough, since they can be taught and honed, and in the long run, a severe shortage of soft skills could become a serious detriment to the whole team and, consequently, your company.

Why Soft Skills Are Essential In Your Developers

If you remain unconvinced, consider the following aspects to understand why soft skills are essential in your developers:

1) Communication

Communication skills make up a big, big part of soft skills, and for good reason. **An uncommunicative developer is an uncooperative developer.** If they're excessively timid or withdrawn, they will not be a good fit for an environment that relies on teamwork and problem solving.

2) Retention

When new talent joins your team, you need to train them to some extent. It's an investment—sometimes significant, though always smart and necessary.

People are the most valuable resource, so the idea is to attract or mould professionals and keep them working for you.

However, imagine you've whipped your fresh developers into shape, but their insufficient soft skills make it impossible to work with them. What can you do? Nothing.

You are forced to part ways, and all the precious time and money you've invested in their training goes to waste on your end.

3) Synergy

We all know the saying, “The whole is greater than the sum of its parts.” The same is true for your development team.

Every individual on the team plays a role in making it what it is. A healthy and friendly work environment is a very fragile ecosystem, easily disturbed by a single developer who is a particularly poor fit. This is not an exaggeration: **one “rotten apple” may be an influence bad enough to cause some of the other team members to leave.**

Unfortunately, this can be difficult to spot before making the hire. STX Next has dealt with candidates who seemed like a good fit during the interview, but later, when they joined the team, it turned out that some of the other team members had worked with the new developer before and the experience had been an extremely unpleasant one.

You should watch out for any clear red flags, though sometimes there’s just no way to know before it’s already too late and the hire is made.

4) Feedback

Teamwork comes with conflict; it’s unavoidable. Different people have different opinions, and the workplace is where they clash the most often. **To avoid tension and friction, your developers need to be open to feedback, otherwise nothing will get done.**

This is especially true in the software engineering community, due to **peer review** being very common practice. Mistakes happen all the time, and it’s vital that your employees take constructive criticism in stride. They are all in this together and they need to be aware of it.

5) Respect

To top it all off, we have a highly elusive, but invaluable quality: respect. **Respect for one’s superiors, but even more so for their subordinates.**

It’s not surprising that your developers would be on their toes when addressing someone in a position of authority who can fire them. It is the way they treat those who can’t do much for them that makes for a truly civil and pleasant atmosphere in the workplace.

After all, we are all people and deserve to be treated as such, even if we sometimes fall short of expectations. It’s important to have people on your team who understand that.

Five Reasons To Turn Down A Candidate

There are multiple reasons why you wouldn't want to hire certain developers. We could group these reasons into categories and apply a bunch of theory to it, but the simplest and clearest way to think about this is the stage of the recruitment process where the problems should be identified.

Some red flags can only come to light at the IT interview stage, due to the technical nature of this step, but a great deal of issues with your applicant can easily be picked up as early as during the HR interview.

Here are five immediate disqualifiers your HR department should be on the lookout for during their interview. Trust us, you do ***not*** want any of those in the developers you hire.

1) Lack of soft skills

We've already explained the differences between soft and hard skills, and why the former are arguably even more important than the latter. If a candidate appears to be lacking in those during recruitment, there is a strong possibility they won't be a good fit for your team.

2) Excessive financial expectations

The salary formula exists for a very good reason. It sets clear, reasonable expectations for your developers and realistically outlines their career paths for them. The formula is transparent and fair, preventing earnings from dividing your team members and spoiling their relationship. If you start bending the rules for one candidate, you risk losing the trust and confidence of your current employees.

3) Insufficient work experience

This naturally depends greatly on whether you're looking for a junior, regular, or senior developer. However, even at the junior level you need to hold your developers up to certain standards and have some basic, inflexible requirements.

Providing software engineering newcomers with an environment to learn and grow is commendable and can be good practice, but a complete amateur may hinder your workflow, not improve it.

4) Inadequate knowledge of the English language

This is a must if you're based in a country where English is not an official language and you're looking to hire local talent. Your candidate's command of English has to be good enough to use it on a daily basis and communicate in it effectively.

A vast majority of online activity in the software engineering world happens exclusively in English, and your developer will never be on the same level as the others if this is an issue for them.

5) Unwillingness to sign a B2B contract

Like it or not, the reality of the tech world is that a great deal of business is done via the B2B model. Your candidate must be aware and accepting of this, as well as willing to set up their own company to do business with you if they don't yet have one during the recruitment process.

You should also be understanding—and appreciative—of their setting this up solely for the purpose of working for you, but their reluctance to agree to B2B should immediately eliminate any possibility of further talks on your end.

Last Piece Of Advice… Okay, Two

To conclude our ultimate guide to hiring software developers, we'd like to leave you with not one, but two last pieces of advice:

1) Welcome your candidates with open arms.

It's true that you should set clear expectations for your candidates from the start. But in no way does that mean being uninviting toward applicants who do not meet your requirements.

Never alienate your candidates. Always encourage them to apply.

Granted, in some extreme cases they may end up pointlessly wasting your time, but a sincere effort to work at your company generally goes a long way. If anything, you have succeeded in making them *want* to be a part of what you're doing, which is exactly what you're going for.

This is why feedback is so important for the good image and reputation of your company. You have no obligation to hire a candidate whose skills are lacking, but why not help them get better and get feedback about your process, too?

They chose you. They came to you. They tried. That alone deserves praise and gratitude. **It's fine to turn them down, just don't put them down.**

A candidate may be underqualified now, but that may change in the future. Once their skills grow, it's likely they'll reach out to you again.

Why? Because they'll remember the positive recruitment experience at your company. And when they do come back for another interview, the conversation may go quite differently. It might even lead to a successful hire.

Regardless, all of the above can only happen if people apply. Your job is to convince them to get their foot in the door.

And then… just talk. See what happens. Nothing beats a good old face-to-face sit-down.

2) Don’t lose track of the competition.

The competition never sleeps. The rules of the game are ever-changing. Keep up with the latest developments to stay one step ahead of the others.

This is especially true in the software engineering world. Your developers will be expecting to work with new technologies and constantly think of new ways to build, execute, and deliver. Otherwise, they will burn out and leave your company in search of other opportunities and challenges.

Always keep things fresh and interesting if you want your teams to be productive and satisfied.

Final Thoughts

There you have it: everything you need to know about hiring software developers.

If you're reading this, you should realize full well why paying extra attention to every aspect of the recruitment process will pay dividends for your company in the long run.

Keep the methodology we have outlined for you in mind, and you will find more and more developers applying to work with you, leaving positive feedback whether you ultimately sign a contract or not.

With that said, some companies decide their efforts are better directed elsewhere. Instead of spending precious time and money on hiring, they turn to companies such as STX Next for support.

To learn what results you can get by outsourcing your software development, feel free to browse the many stories of successful cooperation in our Portfolio.

Hire an exclusive Python development team

Accelerate your software project with Europe's largest Python software house.
For companies with big projects and fast deadlines.

Team Extension

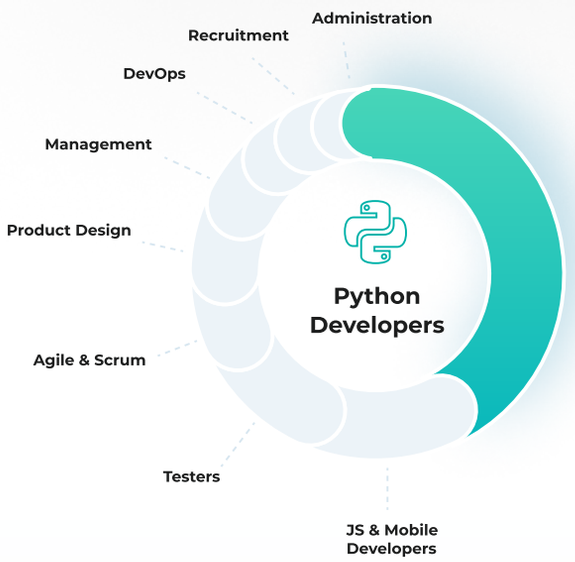
Additional developers or experts supporting your development efforts within 14 days

End-to-End Development

Full development team taking your project all the way from discovery to deployment

Consulting & Expertise

Solving your problems or improving your product with the help of subject matter experts



Over 400 developers

Ready to empower any project with well-reviewed code and a results-driven Agile process

17+ years
market experience

750+
projects delivered

3.5+ years
average partnership

300+
clients served

550+
professionals on board

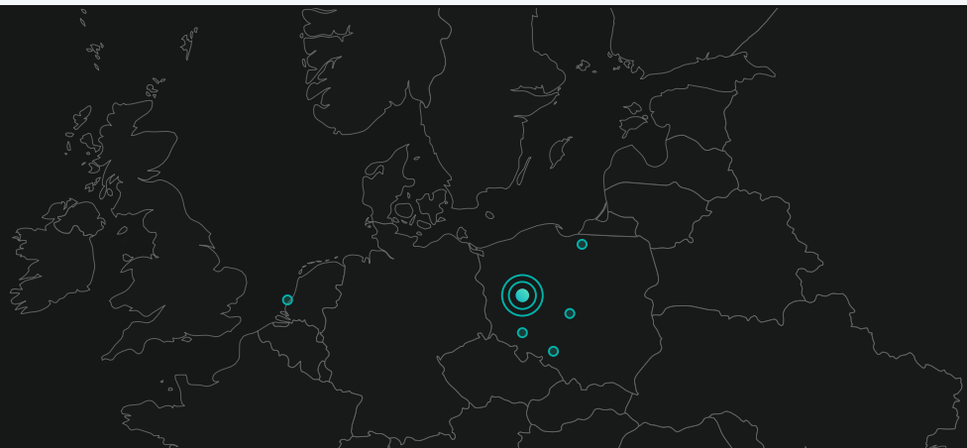
6.5+ years
average experience of our developers

Locations

Poznań (HQ)

Mostowa 38
61-854 Poznań, Poland
+48 61 610 01 92

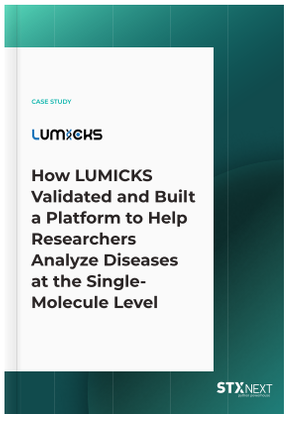
- Wrocław
- Olsztyn
- Katowice
- Łódź
- Hague (Netherlands)

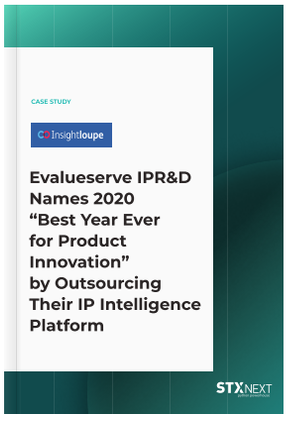


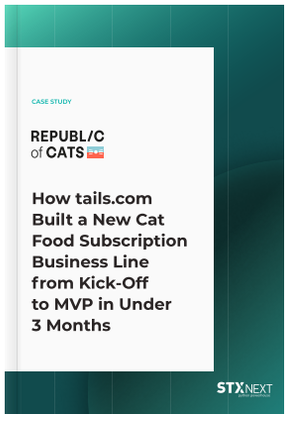
Resources

Arm yourself with the expert knowledge you need to successfully deliver software projects.
Get free in-depth resources, templates, and checklists—all based on 17+ years
of software development experience.

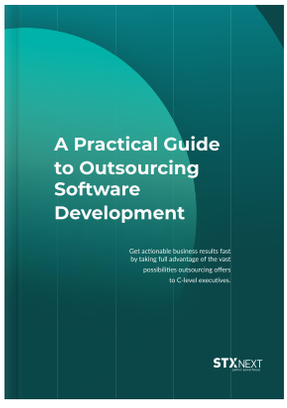
Reports and case studies

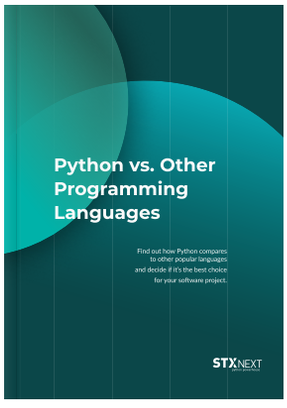




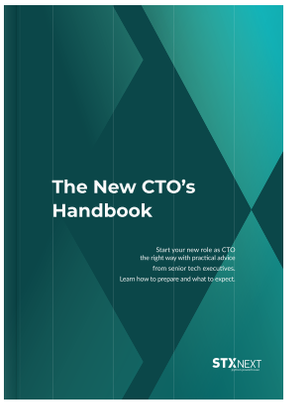


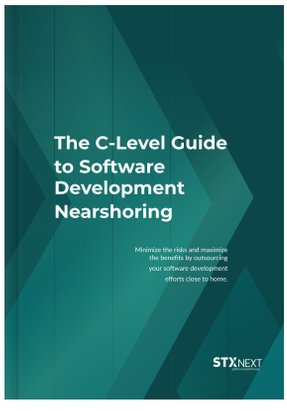
Guides





Ebooks





DISCOVER MORE →

Services

Your project is all that matters. We'll build it like it was our own. Whether it's team extension, end-to-end product development, or expert consulting you're after, we'll do everything in our power to meet your needs.

- Python Development
- JavaScript Development
- .NET Development
- Web Development
- Software Testing & QA
- Mobile Development
- Django Development
- Node.js Development
- React Native Development
- Fintech Development
- Machine Learning
- Data Engineering
- DevOps
- Product Design
- Discovery Workshops

REVIEWED ON Clutch ★★★★★ 82 REVIEWS

python SOFTWARE FOUNDATION

Tell us about your project

Speed up work on your software projects and outpace the competition.

HIRE US →

Marta Błażejewska
DIRECTOR OF SALES
marta@stxnext.com
+48 506 154 343

Sebastian Resz
HEAD OF SALES
sebastian@stxnext.com
+48 690 433 578

Follow us