



Python: The Best Language to Kickstart and Scale Your Business

Table of Contents

| | |
|--|----|
| <u>Foreword</u> | 3 |
| <u>Startups and Challenges</u> | 4 |
| <u>The Benefits of Python for Startups</u> | 8 |
| <u>Awesome Things Are Possible With Python: Django Stars Team Speaking</u> | 20 |
| <u>What's So Special About the Django Framework?</u> | 23 |
| <u>Examples of Startups Built with Python/Django</u> | 28 |
| <u>How to Build a Python Team For Your Startup</u> | 30 |
| <u>Conclusion</u> | 38 |

Foreword

Every type of business has its challenges and advantages. And since business and technologies are constantly developing, no single piece of advice can help them develop them the right way. However, there are basics that need to be covered, regardless of what kind of venture you want to work on.

Below, we're going to talk about:

1. Challenges new businesses face on their way to greatness and what programming language is best for startups and is capable of growing along with them.
(Spoiler alert: that language is Python!)
2. Tips on how to put together an effective Python development team.
3. Python tools and features that made us fall in love with it (the list is longer than you might think)
4. Some of the wonderful things you can do with this language.

Startups and Challenges

“Startup” has been a buzzword for a while now, so what is all the buzz about? Why is there so much talk about these companies?

Caring for a startup is a process full of unexpected turns, risks, and entirely new situations brimming with a degree of uncertainty and vulnerability. A startup needs all the support and survival tools it can get.

What Is a Startup?

The term “startup” is being used in many industries to describe everything from tiny living-room ventures or hip apps to massive tech companies. So, what is a startup, exactly?

Our favorite definition comes from Adora Cheung, co-founder and CEO of Homejoy, one of the [Hottest U.S. Startups](#) of 2013. She says,

Startup is a state of mind. It's when people join your company and are still making the explicit decision to forgo stability in exchange for the promise of tremendous growth and the excitement of making immediate impact.

– Adora Cheung

Anyway, most definitions agree that a startup is a company that has recently set up shop. Merriam-Webster, for instance, kindly calls it “a fledgling business enterprise.” One way or another, the key characteristic of a startup is its ability to scale – to grow fast, regardless of geography. This is what makes startups different from small businesses.

Startups aren't necessarily tech companies; however, in a world where many services are slowly but surely moving online, it's common to associate the term with technological solutions.

After some time, companies graduate from being startups. They attract investment or get acquired by larger companies, their revenues and number of employees grow, they open new offices. That is, if they survive long enough.

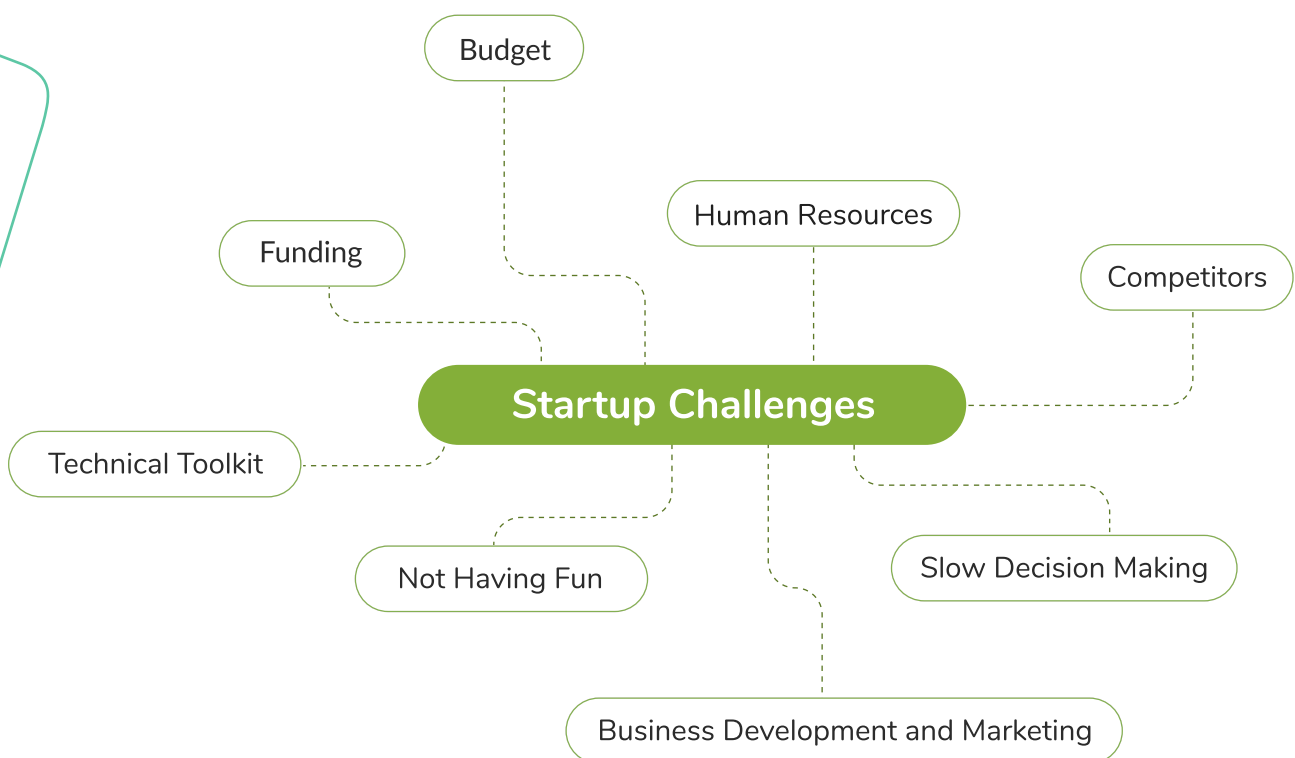
Startup Challenges

But what can happen to bring them out of balance? What are the real challenges that a startup can encounter in the first years of its existence?


Funding, budget, finance. Where do you get startup money? How do you create a budget? How do you manage cash flow? How do you plan your taxes?

Human resources. How do you build an effective team that will join you on this uncertain startup journey? Where and how do you find and recruit the right talent? How do you find the right partners?

Competitors. Who's your competition? What is your vantage point? Being a startup requires a certain ability to react and adapt quickly, and the right strategy.



Slow decision making. A startup is about making a difference, making an immediate impact, and creating a unique solution before anyone else.



If you can't make decisions fast, your startup will lag behind the rest and eventually fail.

Business development and marketing. What business model is best for your startup? How do you protect your intellectual property? How do you identify your users and build a product that solves a problem (even one they may not know they had)?

Not having fun. As we mentioned earlier, people who join a young team want to grow with you and your project, have fun, learn, and make a difference. And besides, what's the point of doing anything if you don't have fun in the meantime?

Technical toolkit. What kind of developers do you need? What programming language is best for your startup? For technical startups like digital platforms, choosing the technical arsenal is the most burning issue. It should be something with a simple syntax and that's easily to scale and maintain.

Since this is our main area of expertise, we'd like to share a couple of ideas on startups out there that got stuck dealing with the technical aspect of their work.

The Benefits of Python for Startups

Our first advice for a startup looking to find its place in the market would be to choose the right programming language for its website or platform. We ourselves chose to use the Django framework within Python, and it's been a great journey so far. Moreover, we keep discovering its true power again and again. It's a simple technology, as it is robust and flexible. On one hand, it can help a startup withstand the stress of perturbations in the market; on the other, it can adapt better to new changes and customers' needs.

We're not saying that Python is the only way to go, but we do know for certain it has a lot of advantages.

1. Python takes you to market a lot quicker


One of the benefits of Python for business is that it lets you build an MVP a lot faster. This increases your chances of finding your product/market fit. Nowadays, companies in every industry require the ability to get to market quick, toughen up in the face of real-life problems, and constantly improve and grow – in one word, to compete at a high level. To do that, they need to be able to test their product on their target audience and improve it, fast.

Startup Stages

1. Identifying market need
2. Conceptualizing the product
3. Building the product roadmap
4. Developing MVP(s)
5. Iterating based on feedback

These are the common stages of development, and at some point you can't go any further without a prototype, and, later, an MVP. You won't be able to sell your product to investors or users until you properly test it.


Python allows you do this quicker and less painfully by offering ready-to-use solutions from libraries (Numpy, Scipy, Scikit-learn, Statsmodels, Pandas, Matplotlib, Seaborn, etc.). You don't need to waste your time developing small things like authorisations or user management tools from scratch.



When your MVP is ready, Python allows you to adapt parts of its code. This means that after you've validated your MVP, you can easily change some lines or even add new ones, if required, without disturbing the rest of the code.

Users these days are used to living in a fast-paced world. Millennials feel like they have to be constantly productive, and they expect the same from the products they use. Basically, there's no time for error. They value maximum transparency and high-quality service.

2. Even if your product is complex, Python programming is not



Python allows you to transform complex products and algorithms into simple code that is fast and cheap to work with. This is possible thanks to Python's simple syntax.

And because it's so simple, it's easier to deal with intricate systems and make sure all the element relationships are working properly. This also makes communication between coders easier.

Python creator Guido van Rossum describes it as a “high-level programming language, and its core design philosophy is all about code readability and a syntax which allows programmers to express concepts in a few lines of code.”

This is as important for collaboration on the development level as it is with clients. As you can imagine, people on both sides of the development process have different levels of technical understanding. Python lets engineers explain the code much easier, and clients can better understand the development progress.

Unlike languages like Java, Python’s programming approach is less limited. It has multiple paradigms and supports multiple programming styles – including procedural, object-oriented, and functional. This makes Python perfect for startups, as they need the ability to change their approach at any given moment.

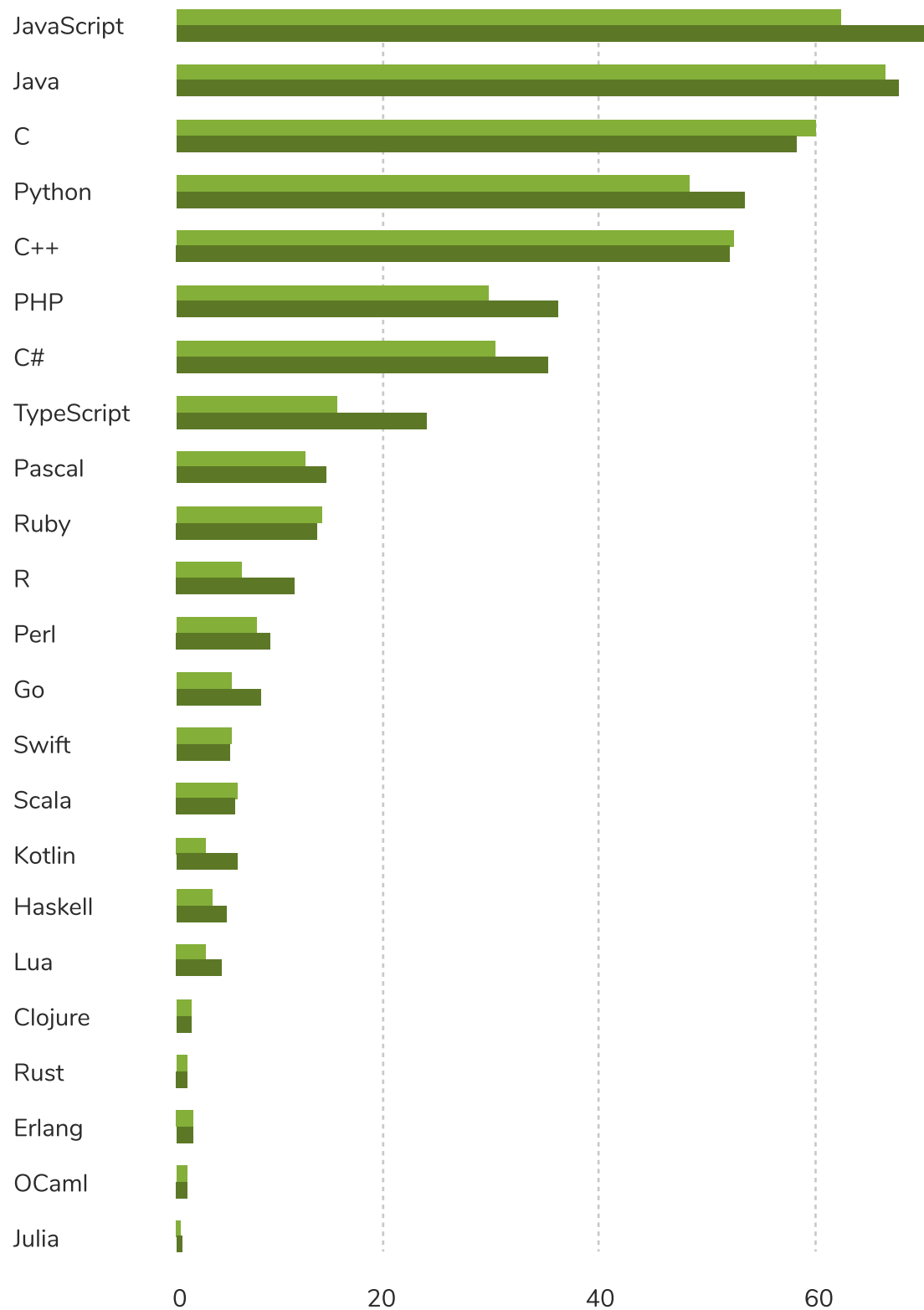
3. Python developers are easier to find

More and more developers want to learn Python. According to the 2019 HackerRank Developer Skills Report, in 2017 about 49.44% of developers learned Python, and in 2018 this number rose to 57.02%. This makes Python one of the four coding languages that most developers want to learn. As a result, companies that work with Python will probably never run out of specialists.

All this proves that you'll be able to find a team to work with easily and quickly, and there will be no shortage of specialists to develop and maintain your products in the future. On top of this, it will be easier to fill the gap in developers, which seems to be an issue in many countries.

Python is incredibly versatile – it can be used for all kinds of purposes, both traditional (web development) and cutting-edge (like AI). It's the go-to language for non-technical fields like business, as it works best for data analysis (especially financial analysis).

Languages Known in 2017 vs 2018



4. Python is open-source and has open libraries

With open libraries, you don't have to build your tools from scratch. This means you can develop the product and analyze large amounts of data in the shortest possible time (and save money as you go).

Python libraries include those for API integration, which makes the process a lot easier. API integration is especially valuable for fintech products, as it allows companies to collect and analyze data about users, real estate, and organizations.

As an open-source product, Python provides incredible flexibility: you can choose from an incredible number of libraries and pick packages for turnkey solutions. Python can be implemented anywhere you want and modified according to the needs of a particular project.

5. Python works for business analytics and business intelligence

One of the main purposes of business analytics is to describe what has already happened. To describe and categorize existing data, data analysts use Python.

This data allows the business to evaluate metrics over time and understand trends.

As a process, data analysis includes data profiling, visualizing results, and creating observations to plan next steps. Python allows you to manipulate data, distribute workflows, and create visualizations.

Furthermore, business analytics allows companies to predict what will happen. Machine Learning, for instance, uses streamlined mechanisms that predict the future using statistics and existing information.

This is where Python comes in. It's quickly becoming the #1 language of Machine Learning. It's used to create models for Bayesian networks and decision trees, among others. To access numerous supervised and unsupervised machine learning algorithms, data scientists use Google's TensorFlow, a popular Python library.

One more big application of Python is decision science, which is the final stage of business analytics. It anticipates outcomes, when and why they will happen, and determines *how to apply this information*.

Python and Data Science for Business

PYTHON

Business Analytics

Business Intelligence

Decision Science

What was happened?
What will happen?
How to apply this information?

Decision scientists align their data analysis with business problems, as their main goal is to make use of the insights they get. They use Python to create prescriptive analytics tools that use artificial neural networks to optimize outcomes.

Depending on who you ask, business analytics is either the child or the parent of business intelligence. Like business analytics, BI also collects and analyzes actionable data, but with a different purpose – *to provide insights into how to improve business operations and discover pain points in your workflows.*

6. Python is perfect for AI and ML

Artificial Intelligence and Machine Learning are the hottest trends in the IT industry these days. In light of the large amounts of data that need to be analyzed and processed, AI and ML have become a real-

world necessity rather than an idea from science fiction. Based on research, Jean Francois Puget from IBM's machine learning department once said that Python is the most popular language for AI and ML. How come?

First of all, as we've already established, it offers a great variety of libraries with pre-written code and base-level items, so that developers don't have to waste their time creating them again and again. Python libraries allow for continuous data processing, which (for instance) Machine Learning requires. On top of this, it has a low entry level, and thus, a growing supportive community. It's also very flexible and versatile, and can run on any platform.

Use cases of AI and ML include fintech, travel, healthcare and many others. Fintech uses the AI and ML to predict the best investments; the travel industry wants to provide their customers with customized offers; while healthcare uses AI and ML to detect injuries and predict and scan diseases.



[Read](#)

7. Python doesn't require a large team

Thanks to Python's simplicity and readability, a small development team can handle all the work and eventual scaling of the product.

With a relatively small team, it's easier to communicate and react to outside changes. This is especially important for startups, as they have to be nimble and adjust in order to survive. Also, it's obviously cheaper to maintain a smaller collective. The money you save can be invested into the growth of your startup!

8. Python is easily scalable

Every startup should be ready for any kind of change, including quick growth.

It's every startup's dream and goal to become big one day, but not everyone is ready for big changes, even when they think they are.

Python is a language well suited to scalability and can endure large amounts of data. As mentioned above, Python is very simple. This simplicity allows developers to scale a product up or down, without having to involve any time-consuming processes.

To conclude, let's summarize what makes Python the best programming language for startups:

The Benefits of Python for Startups



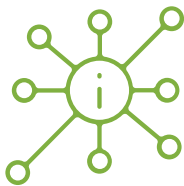
Business Analytics and
Business Intelligence



Open-source and
Open Libraries



Large Hiring Pool



Faster Time to Market



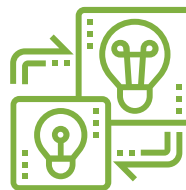
AI- and ML-compatibility



Simple Syntax



Small Teams




Scalability

Awesome Things Are Possible With Python: Django Stars Team Speaking

For us as a team, it was important to find a tool that we could work with on a daily basis (and love it) and reach our goals as easily as possible. We chose Python, and we use it every day. Over the years, we've definitely solved more than one problem that your business may encounter. We've learned that since it's highly universal, Python works with numerous other tools, which essentially means that there's nothing Python can't do. That's why we'd love to share our experience with you – and prove that, with Python, you can do anything.

Working with Docker

The truth is, Python cannot be compiled into an executable file like an application written in Java or C++, for instance. In a way, it's similar to how differently shaped goods can't really be stored or transported together. Which is why [Docker](#) can come to the rescue. Docker is essentially an open-source tool that provides a universal form for delivering your application with all its settings, databases, requirements, and environments.




On top of this, Docker works with all major Cloud providers, which allows you to migrate your application from your own physical or virtual server to the Cloud – and the other way around – without any problems.

Working with Asyncio

As you may know, Python is a sequential language. This means that all requests are executed one by one. But if you have complex code on your hands that requires simultaneous execution, you can use [Asyncio](#) to have all the work done at the same time.

For instance, imagine you have a script that requests data from 3 different servers. Requests to one of the servers can be done in no time, but the others require 10 or more seconds. And while you're waiting for that one request, the entire script is actually doing nothing.



With Asyncio, you can write a script that allows you to switch tasks and not lose valuable time. Some say Python being strictly sequential is a disadvantage, but it's so universal that even drawbacks can be turned into advantages.

Merging Django ORM with SQLAlchemy

This one is quite specific, and mostly used for data analysis applications that require extra- tricky database requests. In cases like this, the Django ORM flexibility isn't enough, and it's a lot easier to use external tools. Which only proves that, despite the Django ORM, it's possible to combine similar tools and benefit from them.

Using List Comprehension

Python is famous for its simple, elegant, and easily readable code. List comprehension is one of its features that allows you to create powerful and versatile functionality within a single line of code. Like no other language, Python enables you to optimize code and do big things with just a couple of lines.

Testing Third-Party Integration Using Mock Data

It's common practice for third-party services to charge for every request sent to them. Which is why testing application integration can get somewhat costly. To avoid these expenses, we create a mock service that serves the same purpose. This way, using fake data, we

can do the testing for free. As soon as the process runs smoothly, we replace the mock service with the real one – and voila! This is one of the reasons why Python is so good to work with: you don't need to build the integration from scratch, but simply redirect the requests to the real service.

What's So Special About the Django Framework?

If you're not familiar with Django, a short introduction is in order.

Django is an open-source framework for Python-based backend web applications. In fact, it's one of the top web development languages.

Django preaches flexibility, simplicity, reliability, and scalability.

For all functions and components, Django has a naming system of its own – for instance, HTTP responses are called “views”. Also, Django has an admin panel that's widely considered to be easier to work with compared to Lavarel or Yii. Django's other technical features include:

- Simple syntax;
- Its own web server;

- MVC (Model-View-Controller) core architecture;
- All the essentials needed to solve common cases;
- An ORM (Object Relational Mapper);
- HTTP libraries;
- Middleware support;
- Python unit test framework.

In addition, Django provides a dynamic CRUD (create – read – update – delete) interface, configured with admin models and generated via introspection.

So why do we use the Django Framework?

Safety and simplicity

The framework uses the principles of rapid development. This means that developers can do more than one iteration at a time. Also, the Django framework uses the DRY — Don't Repeat Yourself — philosophy. It means that developers can reuse existing code and focus on creating new, unique code. All of this results in a faster time to market.

Security

Since security is always a top priority, Django has one of the best out-of-the-box security systems. It helps developers avoid common security issues like clickjacking, cross-site scripting, and SQL injection. Django is known for releasing new security patches fast, and it's normally the first to respond to danger and alert other frameworks.

Suitability for any web application project

Whether it's a simple website or a high-load web application, with Django you can take on projects of any size and complexity. The framework has enough extras for making applications that handle heavy traffic and very large volumes of information. Django is cross-platform, so your project can be based on Mac, Linux or PC. On top of this, it works with most major databases, and allows for the use of any database you find suitable, and even multiple databases at the same time.

Popularity

Django has one of the largest communities out there, and it's been properly time- and crowd-tested. Numerous forums and dedicated websites provide access to information, and contact with other fellow developers for advice or collaboration. Django has the best documentation among most open-source frameworks, which is still maintained at the highest level. The software is constantly updated, and any bugs and glitches can be fixed in the blink of an eye.


Fun fact: Django was first created to support a web application for a news publisher, the Lawrence Journal-World. It gloriously handle large volumes of texts, media files, heavy traffic, and anything else that works like a web-based periodical

Soon enough, other companies realized the brilliance of Django, and now it's being used to build eCommerce websites, health care and financial applications, as well as apps for transportation, booking, and much more. Here's what you can build using the Django framework:

- Financial platforms that can analyze and calculate approximate results based on personal data, risk tolerance, and probability of achieving goals;
- Built-in custom CRM systems for internal data;
- B2B CRM systems for handling communication between businesses;
- Platforms that facilitate communication between the two parties, like a business and a consumer;
- High-load booking engines or shopping platforms;
- Android and iOS mobile apps that support web applications;
- Real estate property evaluation systems;
- Document management systems;
- Platforms for handling legal issues, such as verifying mortgage conditions or lease status.



travel.padi.com



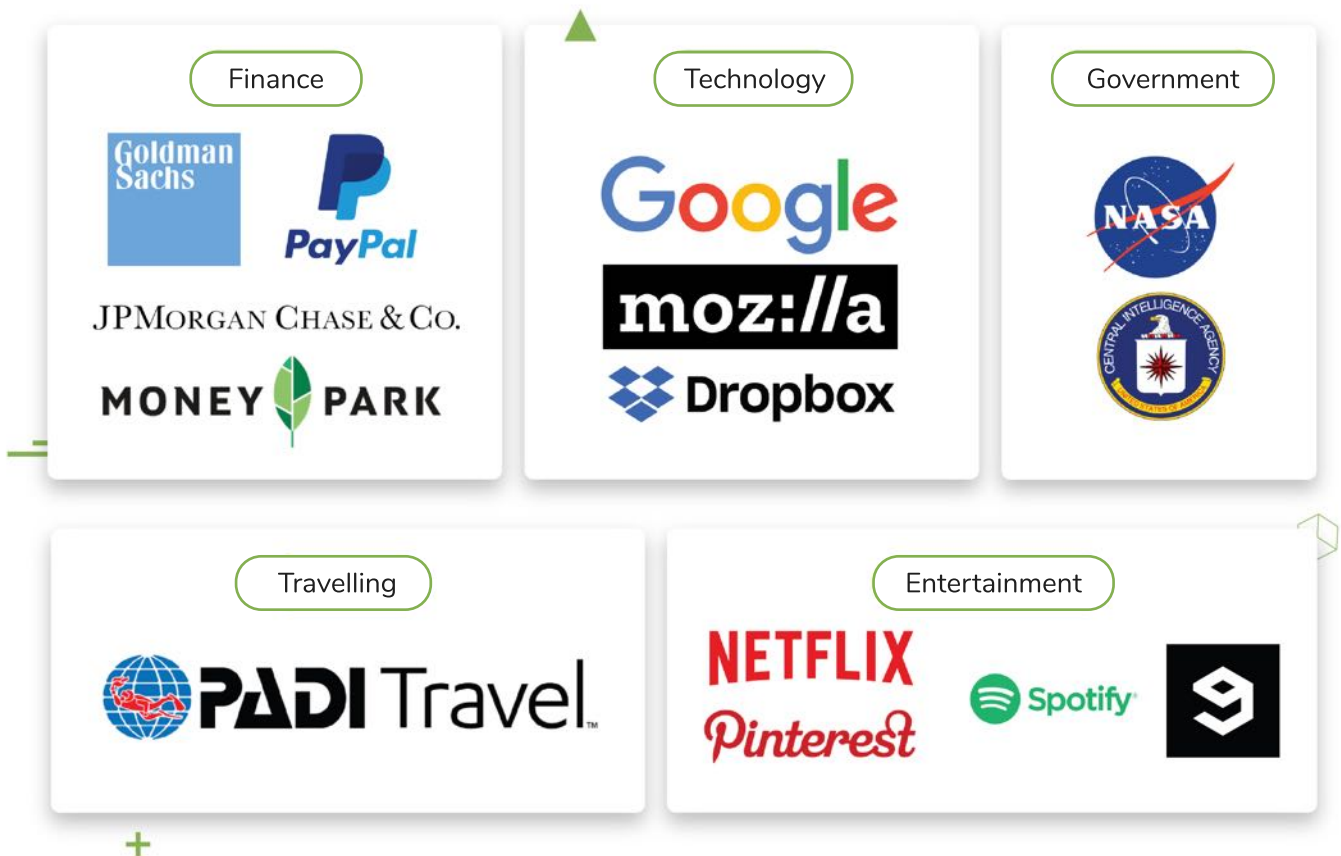
Some companies base their projects on more than one framework, which means that different frameworks are used to create different features. You can use Django to create the following separate features:

- An email system for sending notifications to users;
- A filtering system with advanced logic and dynamically changing rules;
- Algorithm-based generators;
- Data-analysis tools;
- Interfaces for managing investment funds;
- Admin dashboards;
- Photo-based verification systems;
- Other features that facilitate the development of CRM and B2B platforms, online marketplaces, booking systems, and more.

Examples of Startups Built with Python/Django

As you can see, using the Python/Django combo gives you endless possibilities, and a lot of freedom and security.

Many successful companies like Google, Instagram, Spotify, Dropbox, The Washington Post, Mozilla, and NASA appreciate this flexibility and rely on code written in Python using the Django framework.



It's hard to compete with giants, but they're not the only ones who use Python/Django to their advantage. Others include:

[PADI Travel](#) – the world's biggest online resource for divers, which includes a booking platform, travel guide, and social network.

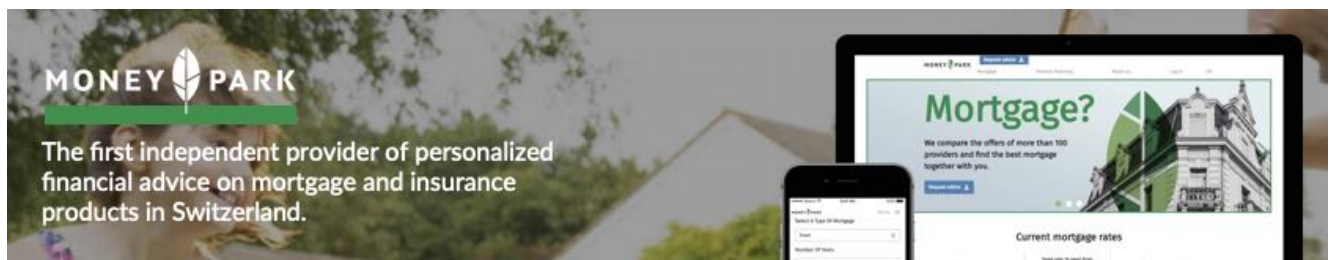
[Festicket](#) – a search engine for music festival tickets, and one of the world's largest music communities.

Password Boss – a digital wallet and password management system.

Hudson – a global recruitment company that used Django to create its own CRM system.

Prezi – an online drag-and-drop tool for creating presentations.

MoneyPark – the first independent provider of customized financial advice on mortgage and insurance products in Switzerland.



moneypark.ch

How to Build a Python Team For Your Startup

Looking at these companies, one can get discouraged: “How can I possibly reach such heights?” But with the right team, everything is possible. Just remember: it’s not enough to find the best specialists, lock them in a room, and give them a deadline. Your team is like a movie cast. They must know their roles, they have to work well

together, they must feel each other, and there must be chemistry between the team members.

Every person on your team has to be good at their job, and know their responsibilities and all aspects of the process. They also have to share your company's ideas – that's when all of you will be able to reach goals and develop successfully.

If you fill in the project roles with the right people who know what they're doing and share your devotion, your project will run smoothly, and you won't waste time and money on settling miscommunications or disputes, explaining over and over again what to do, or redoing development stages.

Team Roles

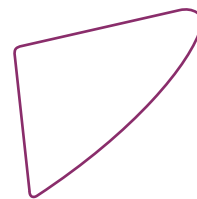
To understand who you're looking for, you have to be clear about the size of the product, your goals, and the ways to reach them. This knowledge will also allow you to know what roles you need on your team and how many specialists to look for.

To use Python on projects, we usually use the Scrum method. The Scrum Guide describes three important roles:

- **Product Owner** – usually, a project's key stakeholder;
- **Scrum Master** – a facilitator, a process owner who coordinates the team's work;
- **The Development Team** – a group of in-house or dedicated developers that work on the project together.

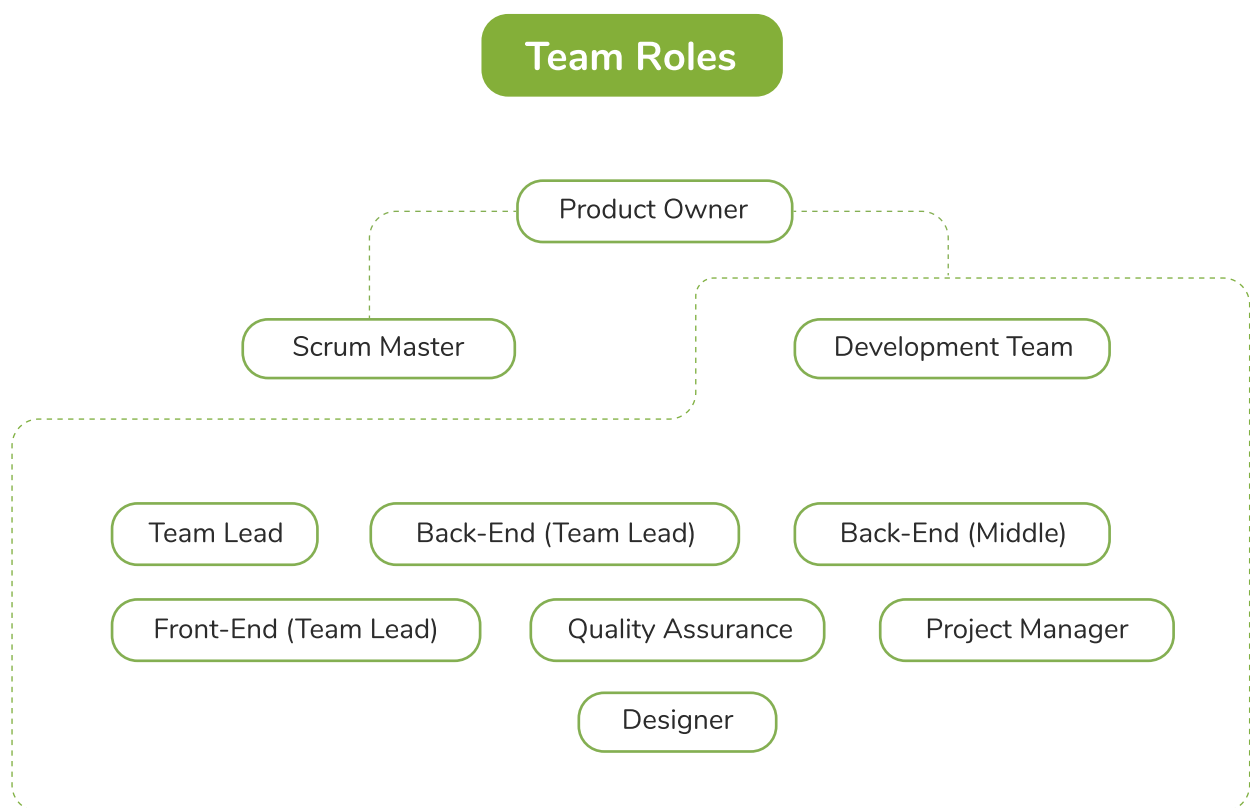
Other than this, there's a bare minimum of people required for a team working on a Python project:

- Team Lead with relevant experience
- Back-end – Team Lead
- Back-end – Middle
- Front-end – Middle
- Quality Assurance
- Project Manager/Scrum Master
- Designer (depending on the project's tasks)



Also, remember:

- There should not be two leaders;
- No people without experience in the domain;
- To involve architects or leaders who analyze the project at the initial stage.



Recruiting

It might not seem important, but your recruiting strategies may affect the quality of the specialists you hire. We advise you to avoid passive recruiting by simply posting a job listing on your company's website or social media page. It's okay to do this, but not as the primary source of candidates. If you do, you may end up sitting and waiting forever for good specialists to apply (and they may never come). Show some initiative – go and hunt down your dream team on LinkedIn, for instance. Check out developers' profiles to see what projects they're working on and when they should be done.

Be sure to properly test your candidates' expertise. If the company owner doesn't have enough technical knowledge, they should think of hiring a Chief Technical Officer who will take control of this process.

Now, when we're past the recruiting stage, what's next? There are some important qualities of teamwork to keep in mind.

Team Spirit

Fill them in on your product, and on the company mission and values. Educate them about your project's goals, benefits, target audience, and how it will improve people's lives. This should raise team morale and motivate your team. On top of this, don't forget to share positive news about your product – such as good reviews and growth in user numbers.

Knowing and Appreciating Responsibilities

For any complex creation to work, its parts have to be done well and match each other. In this case, it means that everyone has to know what everyone else is doing, and how it fits into the big picture.

Knowing how all the parts of your project work together will help the developers appreciate each other's work.

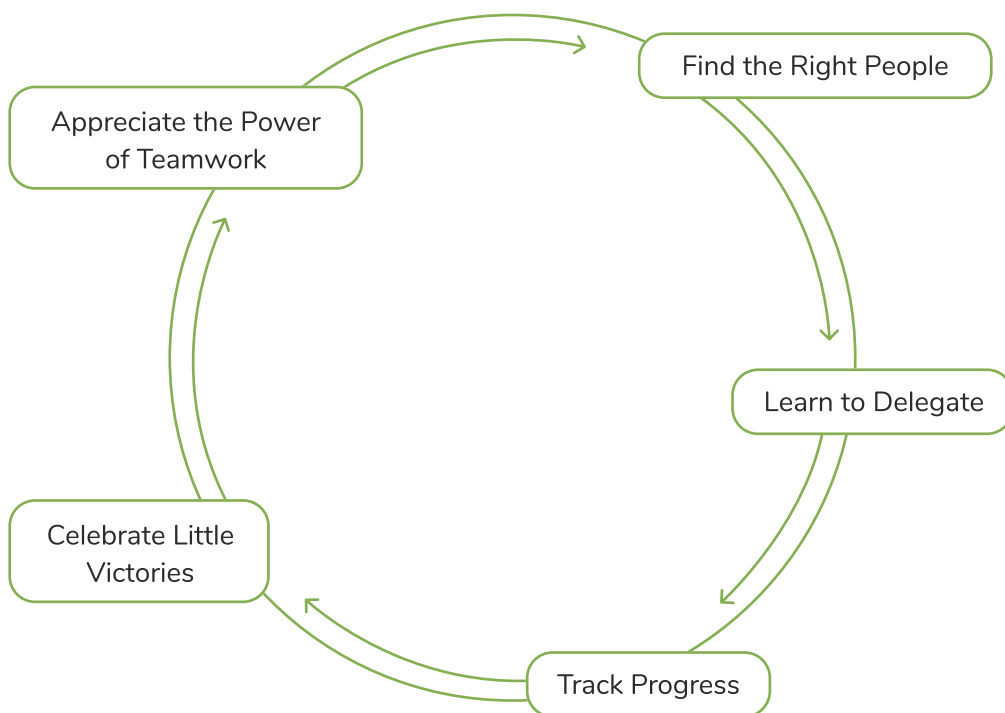
Monitoring Progress


Always keep abreast of what's going on with the project. This will help you quickly fix any problems that arise. Plan regular meetings to discuss progress and challenges. But don't be too controlling – as soon as you organize the process and explain how it's supposed to go, your team should be fine. Learn to trust your developers.

Motivation and Praise

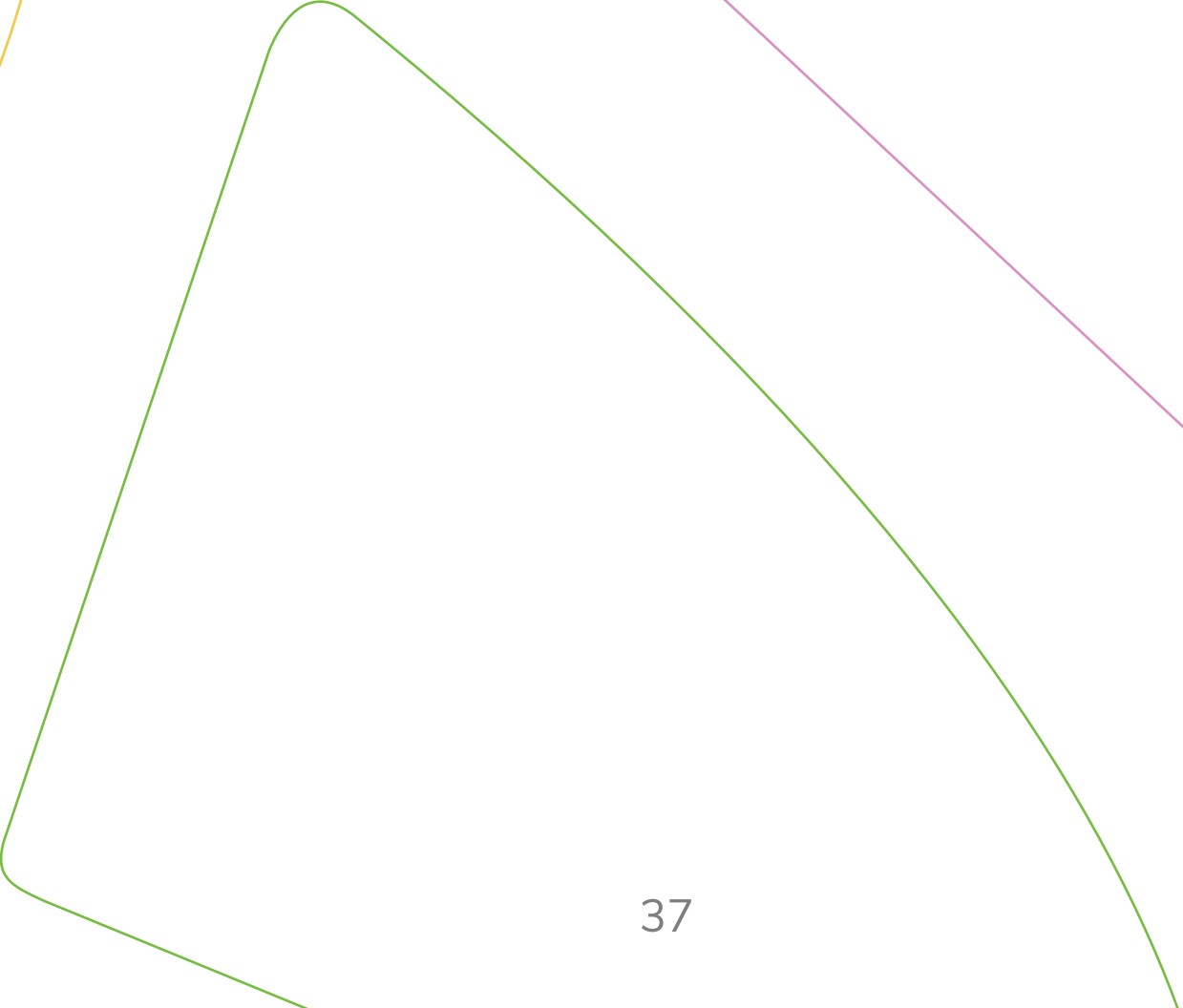
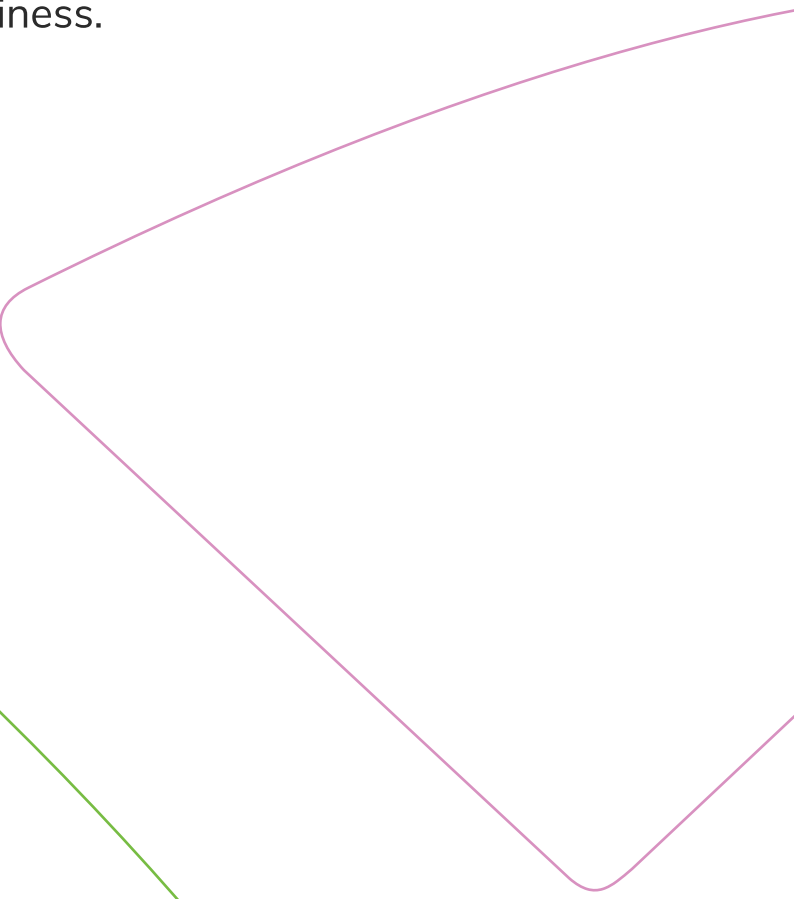

Respect your team's work – and show them that you do! Reward your team for successfully coping with complicated tasks, reaching a certain milestone, or developing a top-notch feature. It may be just a word of praise, or a common activity – but either way, they will appreciate it. All this will help you build an effective team.

Building Really Effective Team





As you can see, there are many facets to creating a startup. But no matter what you do in your business, there should be a solid base that will help you grow. In development, this base is the programming language you choose. It should be easy to learn and work with, and it should allow you to face all the challenges of a newly built company and all the hardships that come along with powerful competition. Most importantly, it should give you the necessary power to scale your business.



Conclusion

We've come to realize that Python is the language that ticks every box, for our company at least. Of course, you should make sure Python meets your product and your business goals. If it does, you're in for a wonderful ride. And if you want to know more, check out our [blog](#) or [get in touch](#).

Python is simple, versatile, allows integration with numerous great tools, and works perfectly for different industries, be it entertainment or fintech. You can explore various functionalities for your product without having to write a lot of code. Which, of course, is good for bringing newbies to the project, as well as product maintenance. And when it's time for your startup to “graduate”, with Python you'll easily scale your product and kick off an entirely new and exciting era for your business.

**CONTACT US TO IMPLEMENT
YOUR IDEAS**

info@djangostars.com