Learning Objectives

- **Define the different methods of production hosting**
- **Explain the pros and cons of each method**
- **Describe the different factors that you should consider when selecting a hosting solution**

Production Hosting

Different Hosting Methods

Strangely, it often seems easier to build your Django application than to actually get it up and running in production. There are many different ways to get your Django application online, and the specific method for each vendor could probably each be an entire module on its own. Hence we'll break them down into two groups: app hosting services or virtual server hosting. Each of these has its pros and cons.

App Hosting Service Intro

You've probably heard of an app hosting service before, many different vendors have them, each with their own special name: AWS has Elastic Beanstalk, Google Cloud has App Engine, Azure's is App Service and there's also Heroku, along with many more. While the nuances of their configuration differ, at a high level they behave the same. The vendor provides a tool (usually command-line based) that you run to upload your code to their platform. There might be a web-UI to configure environment variables, and the vendor might also provide load balancing services or other extra monitoring tools. You'll probably have to use the vendor's database (and other) services, which might cost extra.

Virtual Servers Intro

A virtual server, or virtual private server, is a cloud-hosted virtual server which you have full access to. It's like a computer, hosted somewhere, that you can log into and run whatever you like on. You'll need to configure all the services that you need including web and application servers, databases and so on. Security is also your responsibility, so things like firewalls, OS updates and patches need to be managed. You get to decide how your application is uploaded and updated on the server, which gives you more flexibility but can be a little more difficult – there's no vendor-provided command to do it all. AWS, Google Cloud and Azure all have virtual server offerings, plus there are a plethora of other providers.

▼ **Which vendor should you choose?**
You might make a choice by picking a regional provider, if your application is mostly for locals. Otherwise, you can check out some tutorials to see which one feels best for you, and of course consider the pricing. But if your company is already using a particular vendor, then the choice may have

been made for you!

So in summary, app hosting is less flexible but easier, while virtual servers are more work but give you more options, right? It's not quite that simple. Let's look at a few factors.

Cost

App hosting services can seem cheaper, as they may give a small monthly cost for running your app. But often, that cost doesn't include extra things you might need, like a database, which the vendor may provide as a cloud-hosted service. Furthermore you can only host one app per app-service instance, so each app you want to host will increase the cost by the same factor.

Compare that to a virtual server. You can host as many applications as its resources will allow. You can also set up and run other services like a database on the same server, so you don't need to pay extra for the vendor provided database. This will have an impact on performance, but for a small site it can be negligible.

Ease of Setup

App hosting services can be easier to set up, with their graphical configuration options and single-command deployments. However, these are quite vendor specific. Someone who knows the steps and nuances for setting up on AWS can't directly translate those to Google Cloud, for example. Therefore it might take extra time to figure out a new platform. This can lead to vendor lock in, too.

Compare that to virtual servers which are more standardized. A Linux virtual server from AWS is pretty similar to a Linux virtual server from any other provider, so the same process can be used to set up hosting regardless of the vendor. Tools like Salt or Ansible can be used to automate the setup process.

Scaling

If you need more resources to run your application, app hosting services are definitely easier to scale. Usually by changing a few options in the GUI your app can scale to a more powerful server or more instances. Some providers even allow automatic scaling up and down, to reduce costs during periods of lower demand.

Scaling virtual servers is not so easy, you have to set up a more powerful server and migrate all your data to it. Or manually create more servers and set up load balancing. Both of these methods can be automated, but it's something you'd have to figure out yourself.

Data Persistence and System Services

Some app hosting services are set up to run using containers which aren't designed to persist data. Your application might write data to the file system but you might unexpectedly lose it when the container restarts or you re-deploy your application. If you want to persist data, then you'll have to mount an external persistent volume into the application container or use a vendor-provided object store.

Compare this to a virtual server, the file system is right there and ready to use, and anything you write to it will persist across reboots.

A follow on from this problem is that sometimes it can be difficult to make changes to configuration files. Rather than just tweaking a few settings to a file on disk, you'll need to come up with a script that writes out the configuration on a deployment. This can increase the time to alter and test configuration, as you need a deployment each time; it can also take some time to figure out how to escape variables in the config-writing scripts.

Again, with a virtual server, just make the config changes you need and they're persisted.

Conclusion

There's no one-size-fits-all approach to hosting your Django application. If you're hosting a single Django application with no extra services, and are happy to use vendor provide databases and data stores, then an app hosting service might be the best approach. Otherwise, if you're looking to save costs and are happy with server administration duties then a virtual server could be the want. Either way, this introduction should have given you some extra context in which to make your decision.

In the next module, we'll look at some ways to speed up your Django application with performance optimizations.