

Time	~ 1/2 day
Learning Goals	<ul style="list-style-type: none"> • How to deploy a Rails app • Learn development best practices • Learn about concurrency, dyno hours and database connections

Woohoo! We've come to the stage of deploying your app! But let's first talk about fundamentals!

Development Best Practices

- 1 **Never store sensitive information** like API keys, secrets, passwords etc in your repo and push to Github. Use environment variables instead and make sure to include the file with those keys in `.gitignore`. We like to use `figaro` gem to manage our env variables because it integrates nicely with Heroku.
- 2 **Make sure you don't have debugger or byebug in your production code!** Never ever commit them.
- 3 **Write tests** to reduce bugs and ease refactoring (we will be doing lots of this in the next few challenges.)
- 4 Use the **foreman** gem to manage starting and stopping Rails server and Sidekiq with a [Procfile](#). [Read More](#).

The Various Tools We Use In Production:

Use Case	Service Provider/Tool
Version Control	Git
Code Storage	GitHub/BitBucket
Code Quality	Rails Best Practices /CodeClimate/Codacy/H
Continuous Integration	Codeship/Semaphore/Circle CI/Travis CI
Hosting	Heroku
DNS	Cloudflare/Amazon Route 53
CDN	Cloudflare/Amazon Cloudfront
Redis	Heroku Redis
ElasticSearch	Bonsai/SearchBox
Log Management	Logentries
Error Monitoring	HoneyBadger/AirBrake/Rollbar...
Performance Monitoring	Skylight/New Relic/AppSignal

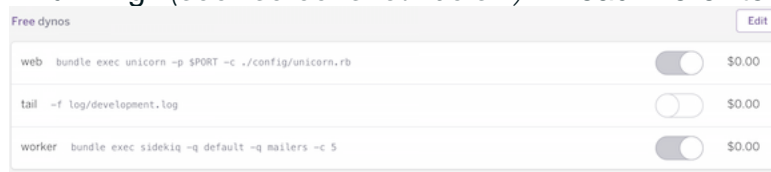
Preparing for Deployment

Most of these apply to various hosting providers but some are specific to Heroku

- Prepare a [Procfile](#). Read up on [foreman gem](#).
- Make sure `rails_12factor` gem is in your `Gemfile` and run `bundle install` Read more about the [12 Factor app](#) or the [official 12 Factor Doc](#).
- Use `unicorn` instead of `webbrick` as your production app server <https://devcenter.heroku.com/articles/rails-unicorn>
- Make sure your app uses `postgres` as database (not `mysql`, not `sqlite`)

Deploy

- 1 Have a Heroku account.
- 2 Have Heroku toolbelt installed on your computer.
- 3 Make sure your app's code base is managed with Git.
- 4 Create an app on Heroku `heroku create <app-name>`
- 5 Deploy your app: `git push heroku master`
- 6 Run database migrations (the database is automatically created) `heroku run rake db:migrate`
- 7 Go to heroku.com on your browser.
- 8 You can run on free dynos for now, but make sure both your rails server and sidekiq are both running (see screenshot below)! Read here to [understandHeroku](#)



Free dynos			Edit
web	bundle exec unicorn -p \$PORT -c ./config/unicorn.rb	<input checked="" type="checkbox"/>	\$0.00
tail	-f log/development.log	<input checked="" type="checkbox"/>	\$0.00
worker	bundle exec sidekiq -q default -q mailers -c \$	<input checked="" type="checkbox"/>	\$0.00

[Dynos.](#)

- 9
- 10 Add the following necessary services for your app:
<https://elements.heroku.com/addons/heroku-redis>
<https://elements.heroku.com/addons/searchbox>.
- 11 Enable database auto daily backup <https://devcenter.heroku.com/articles/heroku-postgres-backups#scheduling-backups>.
- 12 Change your app's domain if you want to buy your own domain! It's easy. You buy domain from places like **GoDaddy**, and then change your DNS settings to point to your Heroku app and within your Heroku settings, you need whitelist your URL.
- 13 If you run your own domain, it's advisable to use a CDN. We like to use Cloudfront as our DNS and CDN cause it's zero code configuration and it's free! [What is a CDN?](#)
- 14 Set up other services such as **Logentries** and **New Relic** from your Heroku app's resources page.
- 15 Set up **Skylight** from <https://www.skylight.io>.
- 16 An error monitoring service is also extremely important. It notifies you whenever an error occurs on your app. Trust us, a lot of errors can happen and are sometimes unpredictable! Unfortunately, we haven't yet encountered a freemium service.
- 17 Code Quality and CI services are also recommended! (You can choose free open-source versions for this, but as your team grows, you want hosted and automated solutions to keep workflow simple and be able to hold everyone on the team accountable.)

Submit Your App URL as solution!

Edit your **Readme.md** to include your app's link: <http://your-app-name.herokuapp.com> or <http://custom-domain.com>.

Other Mandatory and Important Configuration

- 1 Tweak **the number of unicorn workers** to not overuse memory! So the idea is this, your app will have more than 1 simultaneous request. This means that you need a way to handle concurrency to not have to wait for a request to be over before handling another. The way we do it in RoR is commonly by using a forking server such as Unicorn instead of multi-threading. So each unicorn worker will use up some memory. One heroku dyno is 512mb ram, so imagine if each of your unicorn worker uses 150mb, you can only run 3 unicorn workers. Or is it? No you can't! Unicorn has something called a master worker as well. So it's 3 + 1. Unfortunately, we will not be able to give you an answer to how many unicorn workers you should run, every app is different. This is where tools like **NewRelic** becomes important. You first set between 2 - 4 unicorn workers per 512mb RAM. Then you monitor memory usage. If it's too much, you reduce. Not hard at all!
- 2 Set the right number of database pool connections in **database.yml**
<https://devcenter.heroku.com/articles/concurrency-and-database-connections#connection-pool>

Really really good articles on scaling Rails

Don't listen to people who say Rails can't scale. That's so 2007. And people who don't learn anything can't scale anything.

- 1 <https://hackhands.com/ruby-rails-performance-tuning/>
<http://www.nateberkopec.com/2015/07/29/scaling-ruby-apps-to-1000-rpm.html>