Ruby (https://cloud.google.com/ruby/) Guides

Running Rails on Google Cloud

It's easy to get started developing Ruby on Rails apps that run on Google Cloud. Because the apps you create run on the same infrastructure that powers all of Google's products, you can be confident that they can scale to serve all of your users, whether there are a few or millions of them.

Hosting platforms

There are three main options for deploying Rails on Google Cloud.

Rails deployment option	Use if you want	Don't use if you need	Get started
App Engine flexible environment	• Most of the advantages of App Engine • System libraries and RubyGems that depend on them • Custom Docker runtimes	Total control over the underlying virtual machines.	Rails on App Engine flexible environment (https://cloud.google.com/ruby/rails/appengine)

Rails deployment option	Use if you want	Don't use if you need	Get started
Google Kubernetes Engine (GKE)	• Rails containers in a microservice environment • A toolkit to design your own container-based platform	A fully-featured platform-as-a-service (PaaS). For a container-based PaaS, consider App Engine flexible environment.	Rails on GKE (https://cloud.google.com/ruby/tutorials/bookshelf-on-kubernetes-engine)
Compute Engine	• Familiar infrastructure-as-a-service (IaaS) using VMs • Support for both Windows and Linux VMs	A serverless environment without the need to configure your own infrastructure	Rails in Compute Engine (https://cloud.google.com/ruby/tutorials/bookshelf-on-compute-engine)

Databases

Relational databases

Rails Active Record (http://guides.rubyonrails.org/active_record_basics.html) works best with a traditional SQL database. If you are starting a new project, Cloud SQL (https://cloud.google.com/sql/) is a good choice. With a few clicks you can create a MySQL or PostgreSQL database that is fully managed and scaled by Google, with no management on your end. Review the Using Cloud SQL for MySQL with Rails 5 (https://cloud.google.com/ruby/rails/using-cloudsql-mysql) tutorial or the Using Cloud SQL for PostgreSQL with Rails 5 (https://cloud.google.com/ruby/rails/using-cloudsql-postgres).

NoSQL databases

In many situations, there are compelling reasons to use a NoSQL database, for example, scalability or suitability for your data model. Although using Rails models with a NoSQL database can be challenging, it is possible with some limitations. For example, you can express many types of database joins in Rails, but joins aren't supported by Datastore or other NoSQL databases such as MongoDB. You can learn more by reviewing our Rails Bookshelf tutorial using Datastore (https://cloud.google.com/ruby/getting-started/using-cloud-datastore), which uses Datastore as its non-relational database. For a managed, massively-scalable NoSQL solution, consider Datastore (https://cloud.google.com/datastore/), which is a non-relational database that often scales better than a SQL solution.

If you choose to use MongoDB, you can deploy it by using Google Cloud Marketplace (https://cloud.google.com/marketplace/solution/click-to-deploy-images/mongodb) and do your own management, or you can use the managed MongoDB hosting service provided by mLab (https://mlab.com/).

Caches

To install Memcached on Compute Engine, you can use Google Cloud Marketplace (https://cloud.google.com/marketplace/solution/bitnami-launchpad/memcached?q=memcache). To install Memcached on either Compute Engine or GKE , you can use the Memcached Docker image (https://hub.docker.com/_/memcached/). Similarly, you can install Redis by using Google Cloud Marketplace (https://cloud.google.com/marketplace/solution/click-to-deploy-images/redis?hl=en) or the Redis docker image (https://hub.docker.com/_/redis/).

Last updated December 4, 2019.