Rendering Sample Document

mattj23

October, 2019

1 Overview

This project is a LaTeX compiling and template rendering web service intended to run in a Docker container, and interacted with by other software through a REST-like api. It is written in Python3 and uses Flask.

This software was developed as a lightweight (as lightweight as one can reasonably call something housing a full LaTeX installation) infrastructure service for automated document generation. It is meant to be simple and reliable, able to be deployed once for an organization or group and provide the rendering of latex files for many other applications without requiring them to each maintain their own LaTeX toolchain.

This project is essentially a small Flask app built on top of an ubuntu docker image with ‘texlive-full‘ installed, inspired by ‘blang/latex:ubuntu‘ but derived from ‘ubuntu:bionic‘ for the considerable improvements in the 3.6 version of python.

2 Why as a service, and why Docker?

LaTeX, though quite powerful, can be a frustrating toolset to install and maintain, especially across platforms. A comprehensive installation can be several gigabytes in size, and seems to be easily broken. Online tools like Overleaf clearly show how much pain can be saved by not maintaining individual installations, but Overleaf itself is structured towards the concepts of users and projects and isn’t quite a lightweight service meant to be used by other services.

Over the past 10 years, I’ve often relied on LaTeX to generate business documents as part of automated information systems in small business environments. I would develop the code and the LaTeX templates on my windows workstation with a MikTeX installation, then end up having to re-install everything onto the target machine. Invaribly some considerable time would pass and either I’d have upgraded laptops or the deployment machine would become unreliable or broken by some update, and putting everything back together would become an all day affair. Adding another system that needed LaTeX compilation would never end up being as simple as using the existing machine with the already installed toolchain, as I would always end up breaking something.

Docker is great for a lot of reasons, but the one I'm most fond of is that it has largely eliminated the dependency hell involved in setting up machines to host applications that rely on less widely used components. In the SME world, it's difficult to overstate the value of being able to trivally redeploy an important service after the ancient tower running in a neglected closet that hosted it for the better part of the past decade finally decides it's too tired to carry on.

Figure 1: The Universe