The Lean Deployment Playbook

Host, Secure, and Scale Real-Time Apps with Docker, Nginx, and Redis for Under $10/Month

Stanley Amaziro

The Lean Deployment Playbook

Host, Secure, and Scale Real-Time Apps with Docker, Nginx, and Redis for Under $10/Month

Stanley Amaziro

This book is available at https://leanpub.com/leandeploymentplaybook

This version was published on 2026-05-30

This is a Leanpub book. Leanpub empowers authors and publishers with the Lean Publishing process. Lean Publishing is the act of publishing an in-progress ebook using lightweight tools and many iterations to get reader feedback, pivot until you have the right book and build traction once you do.

Contents

Getting Started

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/leandeploymentplaybook.

Unsure How to Get Started? Try our Book Workshop!

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/leandeploymentplaybook.

How to Write on Leanpub

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/leandeploymentplaybook.

Previewing and publishing

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/leandeploymentplaybook.

Basic formatting

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/leandeploymentplaybook.

Markdown and Markua

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/leandeploymentplaybook.

Generate a preview version of your book

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/leandeploymentplaybook.

Either read a tutorial, or just go for it!

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/leandeploymentplaybook.

Read the tutorial...

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/leandeploymentplaybook.

...or just go for it!

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/leandeploymentplaybook.

Thanks for being a Leanpub author!

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/leandeploymentplaybook.

Writing in Markua

Writing in Markua is easy! You can learn most of what you need to know with just a few examples.

To make *italic text* you surround it with single asterisks. To make **bold text** you surround it with double asterisks.

Section One

You can start new sections by starting a line with two # signs and a space, and then typing your section title.

Sub-Section One

You can start new sub-sections by starting a line with three # signs and a space, and then typing your sub-section title.

Including a Chapter in the Sample Book

At the top of this file, you will also see a line at the top:

```
1  {sample: true}
```

Leanpub has the ability to make a sample book, which interested readers can download or read online. If you add this line above a chapter heading, then when you publish your book, this chapter will be included in a separate sample book for these interested readers.

Links

You can add web links easily.

Here's a link to the Leanpub homepage.

Images

You can add an image to your book in a similar way.

First, add the image to the “Resources” folder for your book. You will find the “Resources” folder under the “Manuscript” menu to the left.

If you look in your book’s “Resources” folder right now, you will see that there is an example image there with the file name “palm-trees.jpg”. Here’s how you can add this image to your book:

If you want to add a figure title, you put it in quotes:

Figure 1. Palm Trees

If you want to add descriptive alt text, which is good for accessibility, you put it between the square brackets:

Figure 2. Palm Trees

You can also set the alt text and/or the figure title in an attribute list:

Figure 3. Palm Trees

Finally, if no title is provided, and the `alt-title` document setting is the default of `all`, the alt text will be used as the figure title instead of as alt text.

Figure 4. Palm Trees

You can set the important document settings at Settings > Generation Settings.

Lists

Numbered Lists

You make a numbered list like this:

1. kale
2. carrot
3. ginger

Bulleted Lists

You make a bulleted list like this:

- kale

- carrot
- ginger

Definition Lists

You can even have definition lists!

term 1
: definition 1a

: definition 1b

term 2
: definition 2

Page Breaks

We don't recommend that you manually break pages, since that is brittle and can lead to unexpected formatting if you edit text earlier in your chapter and forget about the manual page breaks. But if you really want to add a page break, you use the `{pagebreak}` directive on a line by itself, with blank lines above it and below it.

Code Samples

You can add code samples really easily. Code can be in separate files (a "local" resource) or in the manuscript itself (an "inline" resource).

Local Code Samples

Here's a local code resource:

Figure 5. Hello World in Ruby

```ruby
require 'time'

# This is just some pointless code so you can see the syntax highlighting...
def display_info
  pi = Math::PI.round(10)
  time_last_year = (Time.now - 365 * 24 * 60 * 60).getlocal("-08:00")
  formatted_time = time_last_year.strftime("%Y-%m-%d %H:%M:%S")
  puts "Pi to 10 decimal places: #{pi}"
  puts "The time 1 year ago in Pacific Time: #{formatted_time}"
end
```

Inline Code Samples

Inline code samples can either be spans or figures.

A span looks like `puts "hello world"` this.

A figure looks like this:

```ruby
require 'time'

# This is just some pointless code so you can see the syntax highlighting...
def display_info
  pi = Math::PI.round(10)
  time_last_year = (Time.now - 365 * 24 * 60 * 60).getlocal("-08:00")
  formatted_time = time_last_year.strftime("%Y-%m-%d %H:%M:%S")
  puts "Pi to 10 decimal places: #{pi}"
  puts "The time 1 year ago in Pacific Time: #{formatted_time}"
end
```

You can also add a figure title using the title attribute:

Figure 6. Hello World in Ruby

```ruby
require 'time'

# This is just some pointless code so you can see the syntax highlighting...
def display_info
  pi = Math::PI.round(10)
  time_last_year = (Time.now - 365 * 24 * 60 * 60).getlocal("-08:00")
  formatted_time = time_last_year.strftime("%Y-%m-%d %H:%M:%S")
  puts "Pi to 10 decimal places: #{pi}"
  puts "The time 1 year ago in Pacific Time: #{formatted_time}"
end
```

Tables

You can insert tables easily inline, using the GitHub Flavored Markdown (GFM) table syntax:

Header 1	Header 2
Content 1	Content 2
Content 3	Content 4 Can be Different Length

Tables work best for numeric tabular data involving a small number of columns containing small numbers:

Central Bank	Rate
JPY	-0.10%
EUR	0.00%
USD	0.00%
CAD	0.25%

Definition lists are preferred to tables for most use cases, since reading a large table with many columns is terrible on phones and since typing text in a table quickly gets annoying.

Math

You can easily insert math equations inline using either spans or figures.

Here's one of the kinematic equations $d = v_i t + \frac{1}{2}at^2$ inserted as a span inside a sentence.

Here's some math inserted as a figure.

$$\left|\sum_{i=1}^{n} a_i b_i\right| \le \left(\sum_{i=1}^{n} a_i^2\right)^{1/2} \left(\sum_{i=1}^{n} b_i^2\right)^{1/2}$$

Figure 7. Something Involving Sums

Headings

Markua supports both of Markdown's heading styles.

The preferred style, called atx headers, has the following meaning in Markua:

```
{class: part}
# Part

This is a paragraph.

# Chapter

This is a paragraph.

## Section

This is a paragraph.

### Sub-section

This is a paragraph.

#### Sub-sub-section

This is a paragraph.

##### Sub-sub-sub-section

```

```
This is a paragraph.

###### Sub-sub-sub-sub-section

This is a paragraph.
```

Note the use of three backticks in the above example, to treat the Markua like inline code (instead of actually like headers).

The other style of headers, called Setext headers, has the following headings:

```
{class: part}
Part
====

This is a paragraph.

Chapter
=======

This is a paragraph.

Section
-------

This is a paragraph.
```

Setext headers look nice, but only if you're only using chapters and sections. If you want to add sub-sections (or lower), you'll be using atx headers for at least some of your headers. My advice is to just use atx headers all the time. (The `{class: part}` attribute list on a chapter header to make a part header does actually work with Setext headers, but it's really ugly.)

Note that while it is confusing and ugly to mix and match using atx and Setext headers for chapters and sections in the same document, you can do it. However, please don't.

Block quotes, Asides and Blurbs

> Block quotes are really easy too.
>
> —Peter Armstrong, *Markua Spec*

> Asides are useful for longer text.
>
> But typing them like this isn't fun.

> Asides can be written this way, since adding a bunch of A> stuff at the beginning of each line can get annoying with longer asides.

Blurbs are useful

Blurbs are useful

There are many types of blurbs, which will be familiar to you if you've ever read a computer programming book.

This is a discussion.

You can also specify them this way:

This is a discussion

This is an error.

This is information.

This is a question. (Not a question in a Markua course; those are done differently!)

This is a tip.

This is a warning.

This is an exercise. (Not an exercise in a Markua course; those are done differently!)

Good luck, have fun!

If you've read this far, you're definitely the right type of person to be here!

Our last piece of advice is simple: once you have a couple chapters completed, publish your book in-progress!

This approach is called Lean Publishing. It's why Leanpub is called Leanpub.

If you want to learn more about Lean Publishing, read this or watch this.

As a backend and systems engineer, my development philosophy centers on a single truth: **production environments are unforgiving.**

Early in my engineering journey, I relied heavily on local setups. Everything always worked seamlessly on my machine. But the moment those systems hit a live staging or production server, underlying environmental mismatches would tear the application apart. I spent long, exhausting nights tracking down silent runtime bugs caused by operating system disparities—like local environments masking case-sensitivity differences in file naming conventions or configuration trees that completely broke critical scripts and real-time process loops the moment they hit Unix-based production Linux nodes.

Worse yet, I watched techpreneurs and indie hackers bleed venture capital and lean startup budgets dry by deploying to managed cloud platforms that charge extortionate markups for basic scaling.

That is why I wrote this micro-playbook. It removes the guesswork from deployment by providing a definitive, containerized blueprint to keep real-time apps fast, secure, and running flawlessly for less than the cost of a cup of coffee a month.

The Cloud Pricing Trap

As a techpreneur, your greatest enemy early on is fixed overhead. When you launch a new application, the temptation is to deploy it to a managed cloud platform that promises "one-click deployment."

It starts at $7/month. Then you need a database—add $15. Then you need a background worker—add another $12. Suddenly, before you even have your first 100 active users, you are looking at a $50 to $100 monthly bill. If your app handles real-time data features like live tracking, WebSockets, or continuous state updates, those managed platforms will quickly throttle your connections or charge you massive premiums for concurrency.

The reality? You are paying a 500% markup for someone else to manage basic software configurations that you, as a developer, can set up yourself in less than an hour.

The Alternative: Raw Compute

By taking control of your own infrastructure using a Virtual Private Server (VPS) from providers like DigitalOcean, Hetzner, or Linode, you can get a dedicated virtual machine with its own IPv4 address for $5 to

$10 a month.

A single $5/month VPS possesses more than enough raw processing power and RAM to handle thousands of active users *if* your application stack is containerized, cached, and proxied correctly.

The Stack Trifecta

To turn a cheap server into an enterprise-grade engine, we use three industry-standard tools:

- **Docker:** Ensures your app runs exactly the same way on your server as it does on your local machine. No more "it worked on my machine" deployment nightmares.
- **Nginx:** Acts as the traffic cop (Reverse Proxy). It sits at the edge of your server, accepts public traffic, handles SSL decryption, and routes it directly to your running app containers.
- **Redis:** An in-memory data structure store. Because it lives entirely in RAM, it handles real-time state, message brokering, and caching at sub-millisecond speeds, taking the massive read/write load off your primary database.

Many developers make the mistake of using a development Docker setup in production. They end up with 1.5 GB image sizes, security vulnerabilities, and containers that lose all their data the second the server restarts. In this chapter, we will build a lean, secure, multi-stage Docker environment designed specifically for production.

The Multi-Stage Build Pattern

In production, every megabyte of RAM and disk space counts on a $5 VPS. We use the **Multi-Stage Build Pattern** to decouple our source code compilation from the final lightweight runtime container. We compile our application in an explicit, dependency-heavy build stage, and then ship *only* the compiled, executable outputs into a razor-thin production environment running Alpine Linux.

--- STAGE 1: The Build Environment ---FROM node:20-alpine AS builder

WORKDIR /app

COPY package*.json ./ RUN npm ci

COPY . .

RUN npm run build

--- STAGE 2: The Production Runner ---FROM node:20-alpine AS runner

WORKDIR /app

ENV NODE_ENV=production

RUN npm ci --only=production

COPY --from=builder /app/dist ./dist

EXPOSE 3000

USER node

CMD [“node”, “dist/index.js”]

Techpreneur Tip: Principle of Least Privilege

Notice the USER node configuration at the bottom of Stage 2. By default, Docker containers execute processes inside the kernel namespace as the root user. Switching explicitly to a non-privileged user closes this massive execution gap, ensuring an application-level exploit cannot escape container limits and compromise your underlying host filesystem.

Orchestrating the Stack: docker-compose.yml

Instead of running long, annoying terminal commands to start services separately, we use Docker Compose to define our entire architecture in a single file.

version: ‘3.8’

services: web_app:

build:

context: . dockerfile: Dockerfile

restart: always environment:

- PORT=3000

- REDIS_URL=redis://redis_service:6379
- NODE_ENV=production expose:

- “3000”

depends_on:

- redis_service networks:
- app_network

redis_service:

image: redis:7-alpine

command: redis-server --appendonly yes --requirepass \${REDIS_PASSWORD} restart: always

expose:

- “6379”

volumes:

- redis_data:/data networks:
- app_network

networks: app_network:

driver: bridge

volumes: redis_data:

driver: local

Chapter 3: The Edge Proxy & Gateway Design Pattern

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/leandeploymentplaybook.

3.1 Understanding Real-Time Traffic Needs

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/leandeploymentplaybook.

Network Isolation and Boundary Config

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/leandeploymentplaybook.

Pub/Sub Messaging Pattern Architecture

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/leandeploymentplaybook.

Automating Free SSL via Let's Encrypt

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/leandeploymentplaybook.

Host Server Hardening

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/leandeploymentplaybook.

Crash Triage Checklist

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/leandeploymentplaybook.

title: The Lean Deployment Playbook

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/leandeploymentplaybook.

The Cloud Pricing Trap

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/leandeploymentplaybook.

The Alternative: Raw Compute

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/leandeploymentplaybook.

The Stack Trifecta

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/leandeploymentplaybook.

The Multi-Stage Build Pattern

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/leandeploymentplaybook.

Orchestrating the Stack: docker-compose.yml

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/leandeploymentplaybook.

Chapter 3: The Edge Proxy & Gateway Design Pattern

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/leandeploymentplaybook.

3.1 Understanding Real-Time Traffic Needs

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/leandeploymentplaybook.

Network Isolation and Boundary Config

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/leandeploymentplaybook.

Pub/Sub Messaging Pattern Architecture

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/leandeploymentplaybook.

Automating Free SSL via Let's Encrypt

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/leandeploymentplaybook.

Host Server Hardening

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/leandeploymentplaybook.

Crash Triage Checklist

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/leandeploymentplaybook.

title: The Lean Deployment Playbook

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/leandeploymentplaybook.

The Cloud Pricing Trap

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/leandeploymentplaybook.

The Alternative: Raw Compute

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/leandeploymentplaybook.

The Stack Trifecta

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/leandeploymentplaybook.

The Multi-Stage Build Pattern

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/leandeploymentplaybook.

Orchestrating the Stack: docker-compose.yml

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/leandeploymentplaybook.

Chapter 3: The Edge Proxy & Gateway Design Pattern

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/leandeploymentplaybook.

3.1 Understanding Real-Time Traffic Needs

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/leandeploymentplaybook.

Network Isolation and Boundary Config

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/leandeploymentplaybook.

Pub/Sub Messaging Pattern Architecture

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/leandeploymentplaybook.

Automating Free SSL via Let's Encrypt

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/leandeploymentplaybook.

Host Server Hardening

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/leandeploymentplaybook.

Crash Triage Checklist

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/leandeploymentplaybook.