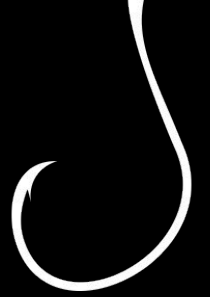
The Modern Full-Stack **Owning Your Deployment Pipeline**

Developer ownership of the pipeline is the key to unlocking value

Jerry Reghunadh

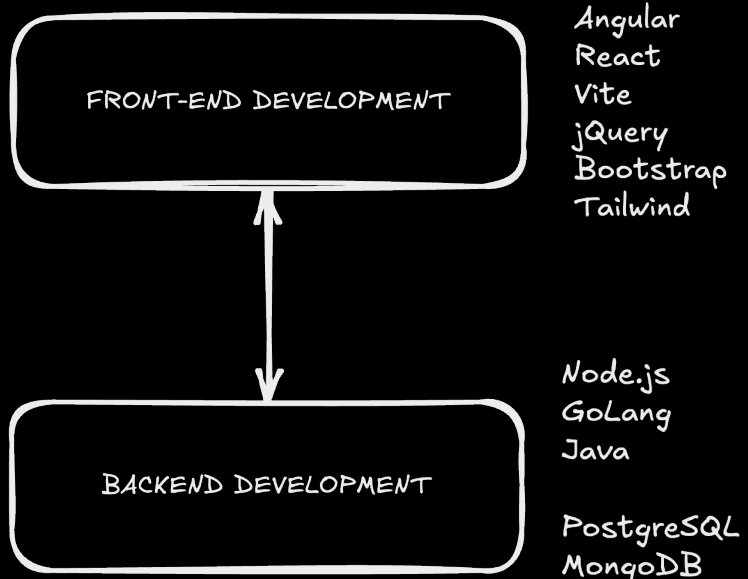
https://jerrymannel.me/

```
{
  "name": "Jerry",
  "whatIDo": [
    "Senior Director",
    "Built multiple 0->1 products",
    "Helped build and maintain solutions for clients",
    "Loves to learn, code, read, and play videogames",
    "Follows 🏎️ Formula 1 & ⚽ Football (the real football)"
  ],
  "pet": "🐶 Joey",
  "recentAchievement": "Wrote the book - The Developer's Guide to AI"
}
```

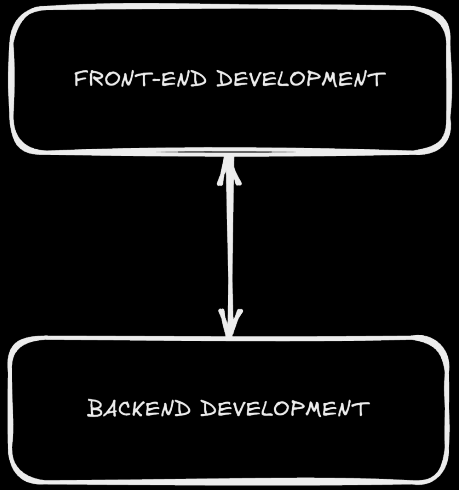


Are You "Half-Stack"?

The "Full-Stack" Engineer



The "~~Full~~ Half-Stack" Engineer



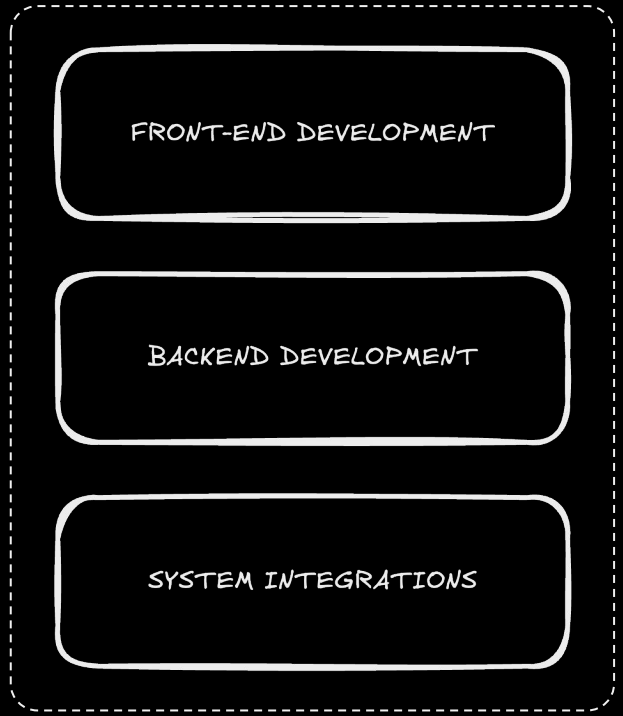
Because no one's bothered about,

- Deployments
- Infrastructure
- Integration

The 2024 State of Developer Productivity Report

"Developers lose 5 to 15 hours per week to unproductive work that could be automated or eliminated."

The "Full-Stack" Engineer



Angular
React
Vite
jQuery
Bootstrap
Tailwind

Node.js
GoLang
Java

PostgreSQL
MongoDB
MySQL

Integration
Deployment
Infrastructure

What’s Holding Us Back?

Fear, Uncertainty, and "The Wall of Not My Job"

Fear of the Unknown

- Learning to live in the terminal
- Not code
- Configurations

What-if Scenarios

- What if I break production?
- What if I take down the whole system?
- What if everyone finds out I don't know Kubernetes?

Organizational Silos

- The Wall of 'Not My Job'
- Do you have a ticket for that?
- Don't touch that; it's not your responsibility.

How?

The Core Principles

The Core Principles

Continuous Integration (CI)

Automates builds and tests on every code push.

Merges developer code frequently into a central repository.

Finds and fixes bugs and integration errors early.

Continuous Delivery/Deployment (CD)

Automates the software release process.

Enables consistent and reliable deployments.

Delivers new features to users faster and with less risk.

Infrastructure as Code (IaC)

Defines and manages infrastructure using code.

Creates consistent and reproducible environments automatically.

Eliminates manual server configuration and errors.

Observability

Collects detailed logs, metrics, and traces from your app.

Allows you to ask deep questions about system behavior.

Helps you understand *why* an issue is happening, not just *what*.

The Core Principles

Continuous Integration (CI)

GitHub Actions
GitLab CI
Jenkins
CircleCI
Azure Pipelines

Continuous Delivery/Deployment (CD)

Argo CD (for Kubernetes)
Spinnaker
Octopus Deploy
often built-in to CI tools like GitHub Actions, GitLab CI, etc.

Infrastructure as Code (IaC)

Terraform
AWS CloudFormation
Azure Resource Manager
Ansible

Observability

Platforms - Datadog, New Relic, Honeycomb

Open Source:

- Prometheus (Metrics),
- Grafana (Visualization),
- OpenTelemetry (Standards),
- Jaeger (Tracing),
- ELK Stack (Logging)

Examples

Dockerfile

```
# Use an official Node.js 20 image as the base.
FROM node:20-slim

# Set the working directory inside the container.
WORKDIR /app

# Copy the package files and install dependencies first
# This layer is cached unless package.json changes.
COPY package*.json ./
RUN npm install

# Copy the rest of your application code.
COPY . .

# Expose port 3000 to the outside world.
EXPOSE 3000

# The command to run when the container starts.
CMD [ "node", "server.js" ]
```

Examples

Kubernetes Deployment

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: mongo
  namespace: jerry
spec:
  replicas: 1
  selector:
    matchLabels:
      app: mongo
  template:
    metadata:
      labels:
        app: mongo
    spec:
      containers:
        - name: mongo
          image: mongo:6.0
          args:
            - --replSet
            - rs0
          ports:
            - containerPort: 27017
          resources:
            limits:
              cpu: 1
              memory: 1Gi
            requests:
              cpu: 500m
              memory: 500Mi
          volumeMounts:
          - name: data-directory
            mountPath: /data/db
      volumes:
        - name: data-directory
          hostPath:
            path: /Volumes/M01/DATA/mongo/db6
```

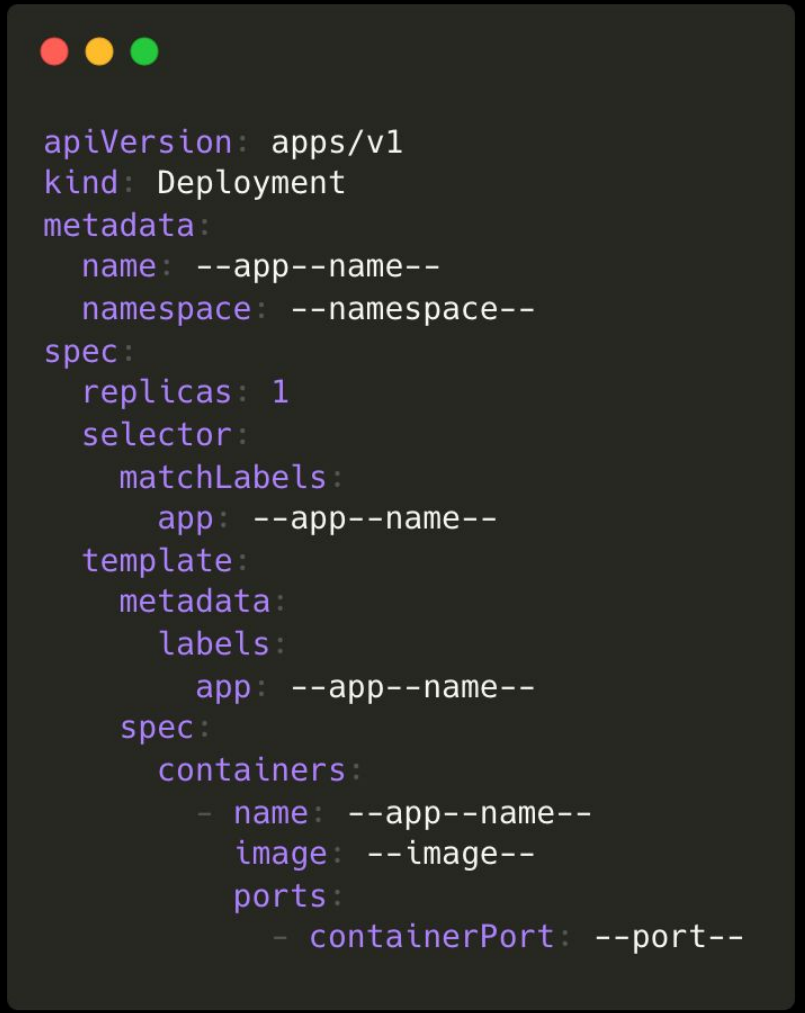
Examples

Ansible playbook

```yaml
- name: Install and Configure Docker on Ubuntu
  hosts: k8s_machines
  become: true
  tasks:
    - name: Get the docker install file
      ansible.builtin.get_url:
        url: https://get.docker.com/
        dest: /home/ubuntu/get-docker.sh
        mode: "u=rwx,g=rx,o=rx"

    - name: Install docker
      ansible.builtin.command: sh get-docker.sh
      changed_when: false

    - name: Set permission for ubuntu user to run docker
      ansible.builtin.command: usermod -aG docker ubuntu
      changed_when: false

    - name: Daemon reload
      ansible.builtin.command: systemctl daemon-reload
      changed_when: false

    - name: Start docker
      ansible.builtin.service:
        name: docker
        state: started
        enabled: true
```

Build Templates

1. Build once
2. *Templatize*
3. Repeat



```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: --app--name--
  namespace: --namespace--
spec:
  replicas: 1
  selector:
    matchLabels:
      app: --app--name--
  template:
    metadata:
      labels:
        app: --app--name--
    spec:
      containers:
        - name: --app--name--
          image: --image--
          ports:
            - containerPort: --port--
```

Best Practices

Best Practices

Shift Security Left

Make Small, Reversible Changes

Use Blueprints, Not Sculptures

Build for Observability

Wrap up

Your 3-Step Challenge to Start ~~Monday~~ Today

Step 01

Get Curious

Step 02

Learn The Fundamentals

Step 03

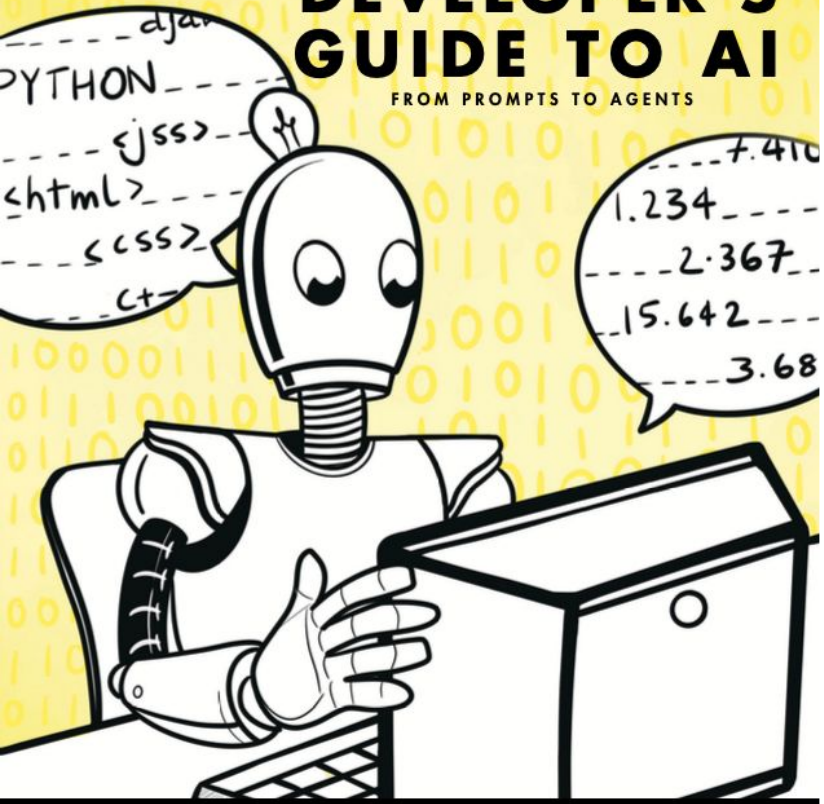
Raise Your Hand

roadmap.sh/devops

Stop being a developer who just** writes code**.
Become the engineer who confidently** delivers value**.
The journey begins when you decide to own your deployment pipeline.

THE DEVELOPER'S GUIDE TO AI

FROM PROMPTS TO AGENTS



JACOB ORSHALICK, JERRY MANNEL REGHUNADH, AND DANNY THOMPSON

THE COMMIT YOUR CODE EDITION

EARLY RELEASE COPY

https://thedevelopersguidetoai.com/

no starch press

Thank you