Starter Architecture

Solution for a scalable and maintainable web application ecosystem

Architecture Overview

Key Points:

"Containerization using Docker Compose to ensure a consistent, isolated **development environment**, reducing setup complexities."

"**Infrastructure management** with Terraform, Server provisioning with Ansible and release engineering through Jenkins and Docker."

"The **web application** comprises Laravel for back-end API development, along with Vue.js/Nuxt for front-end single-page (SPA) and server-side rendering (SSR) applications."
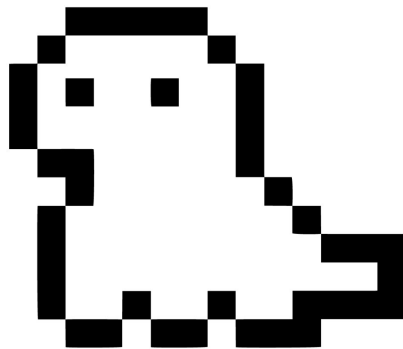
Technologies Used

Key Points:

Terraform - Infrastructure as Code

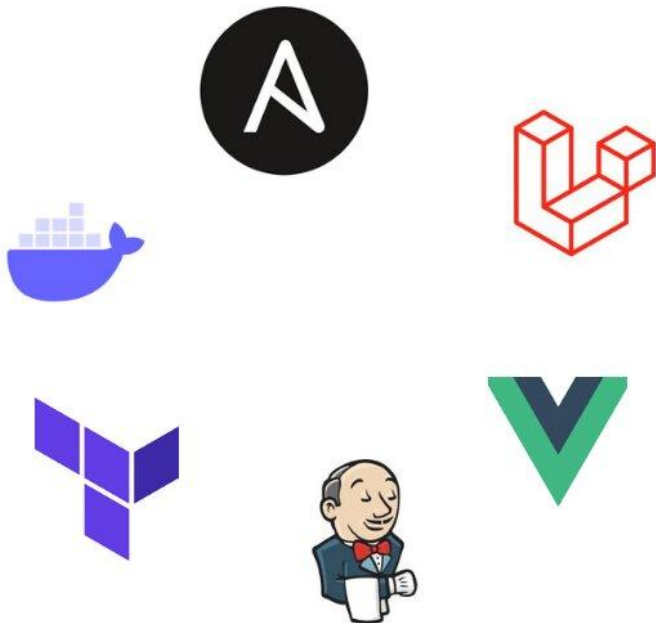**Ansible** - Configuration Management

Docker - Containerization

Docker Compose - Development Environment

Jenkins - Release Engineering

Laravel - Back-end API

Vue.js - Front-end SPA for Admin Panel

Nuxt - Front-end SSR for Public Content

Infrastructure as Code

Terraform: Orchestrating Infrastructure

Key points:

"**Automate** the provisioning and management of infrastructure resources"

"Consistency across all environments, **Version Control** and Reproducibility"

"Cost Control, Collaboration and Security. Infrastructure configurations **can be defined and audited** for security best practices."

Server Provisioning

4. Ansible: Configuration Management

Key points:

"Automates **server setup** and **application deployment**"

"Ansible playbooks are **idempotent**, meaning you can run them multiple times without causing unintended changes."

"Ansible allows you to define and enforce security policies and **compliance standards** across your infrastructure"

Containerization with Docker

Key points:

"Packaging applications into containers offers **consistency**, **isolation** and **version control**"

"Facilitating consistent runtime environments for **development environment** and **pipeline builds**"

"Enhancing **portability** and **scalability**"

Continuous Integration and Deployment

Key points:

"Automating and monitoring **builds** using **Docker Compose environment** and **deployments** using **Ansible playbooks**"

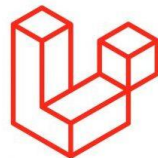"Docker Compose's containerization allows for **portability**, **isolation** and **scalability**"

"Ansible playbooks can enforce **security configurations** and policies and they also offer **rollback capability** in case of issues or failure"

6. CI/CD with Jenkins

Backend API

Key points:

"Laravel offers a robust and feature-rich framework for building **efficient** and **scalable** backend APIs"

"**Domain-Driven Design** architecture, separating the **Business Logic Layer** (Services) responsible for core business rules from the **Data Access Layer** (Repositories) responsible for data persistence."

"**Microservices-Ready**: Each entity within our Laravel backend resides in a separate folder with its Provider, Routes, Requests, Controllers, Services, Repositories, and Models.

Frontend SPA/SSR

Vue.js Admin Panel and Nuxt Public Content
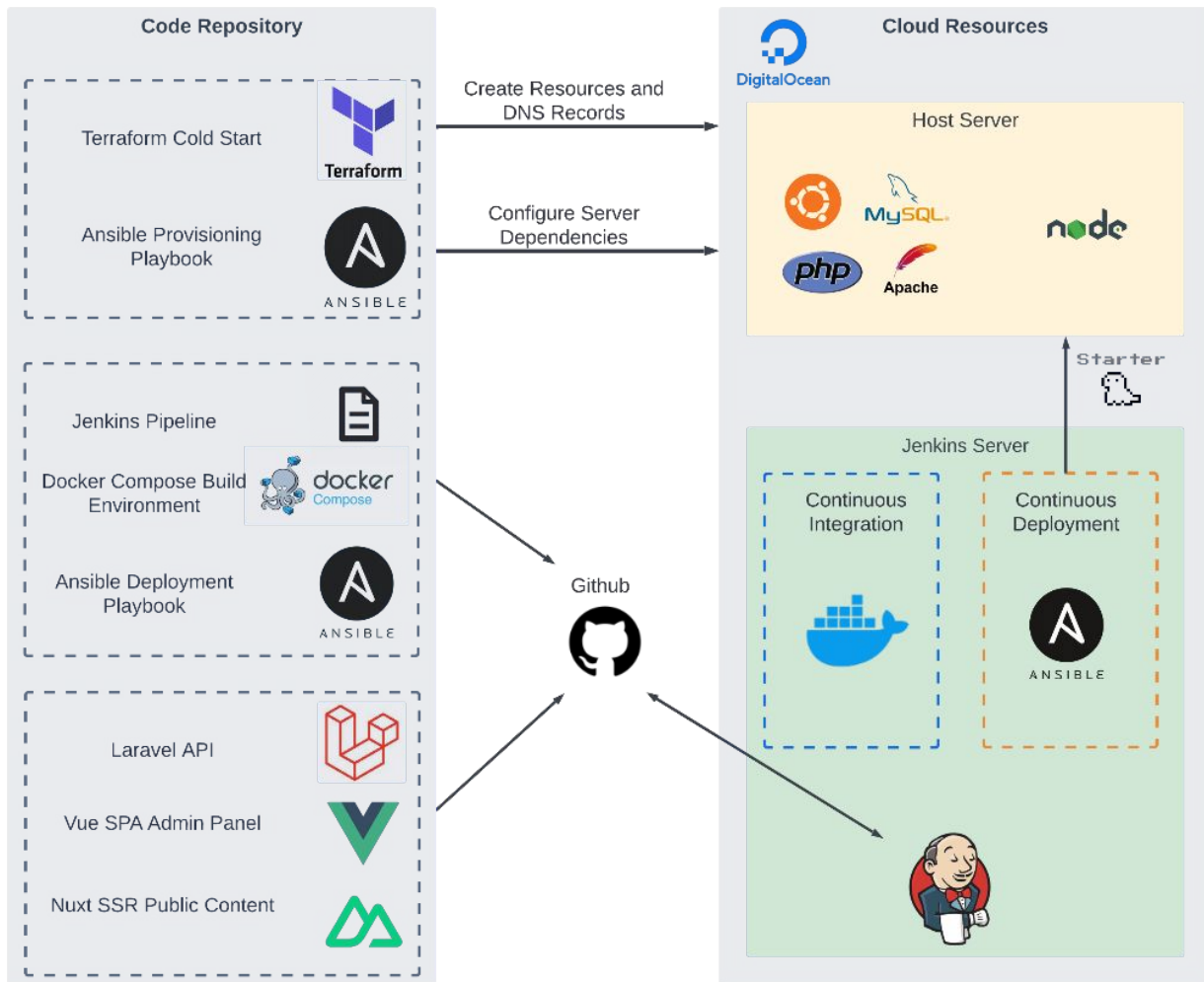
Key points:

"Separation between **administrative tasks** and **public-facing content delivery**, ensuring a responsive and SEO-friendly web application"

"Using **Composition API** syntax and **TypeScript** in Vue.js applications leads to code that is more maintainable, readable, and type-safe"

"We follow the well-established **7-1 Pattern** for organizing **SASS** codebase"
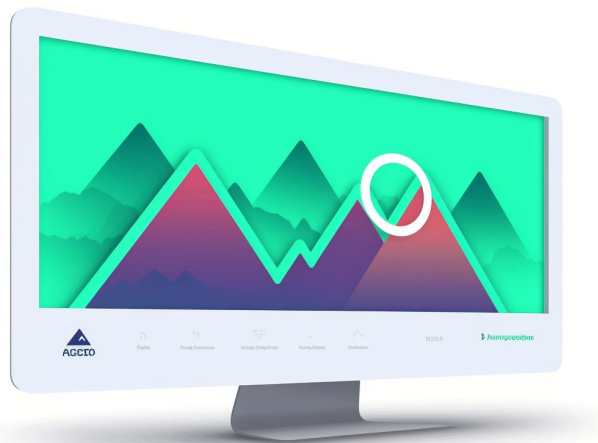
References

Technology Practices Manual

https://docs.google.com/document/d/1ELZcxtIbeMNzzRscpK55a_tI9MkxIvTIDZ38CiXCVfw/edit?usp=sharing

Onboarding Guide

https://docs.google.com/document/d/11Tr-RxFEPbgFKAoXsLrHA9CcN60WgEuKoerccC8HAnw/edit?usp=sharing

Github Repository

https://github.com/NikolovskiRatko/starter-architecture

Follow README.md files, for detailed usage instructions.

Usage Demonstration Video

https://www.youtube.com/watch?v=-ObfNN_rsPM

Engineering Best Practices

https://docs.google.com/presentation/d/1boOPjJzf6ecm7APt7Vr_uhQGT9TetWaaHB_CkCcU9EY/edit?usp=sharing