1. TechNova DevOps Pipeline: Automated Deployment Framework for Web Applications

Project Overview

TechNova Solutions, a fast-growing e-commerce startup selling handcrafted goods, has been hampered by slow, error-prone releases because each code change—whether a bug fix, UI tweak, or new feature—requires manual server setup and configuration, leading to inconsistent environments, occasional downtime, and delayed time-to-market. To overcome these challenges, TechNova needs a fully automated pipeline that can reliably provision and configure cloud infrastructure, automatically build and test the application whenever developers push changes, package and store application artifacts, and deploy updates to production without human intervention. This solution must ensure consistent, version-controlled environments, faster delivery cycles, reduced deployment failures, and easy scalability to handle peak traffic periods—allowing TechNova to focus on innovating rather than managing manual deployment tasks.

Key Features

- **GitHub Repository Setup:** Establish a GitHub repository to manage the application's source code, enabling version control and collaboration.
- **Docker Containerization:** Create a Dockerfile to package the "Hello World" application into a container, ensuring consistency across environments.
- **Jenkins CI/CD Pipeline:** Configure a Jenkins pipeline to automate building, testing, and validation of the application on every commit.
- **Infrastructure Automation:** Utilize Terraform or Ansible scripts to provision an AWS EC2 instance, providing a scalable and repeatable infrastructure setup.
- **Automated Deployment:** Extend the Jenkins pipeline to deploy the Docker container to AWS, ensuring seamless updates to the running application.

Tech Stack

- **EC2/ECS:** Deployment platform
- **GitHub:** Version control
- **Docker:** Application packaging
- **Jenkins:** CI/CD automation server
- **Terraform/Ansible:** Infrastructure provisioning tools
- **Node.js/Python (Flask):** Lightweight front-end framework

---

2. RetailMax: E-Commerce Monolith Cloud Migration Platform

Project Overview

RetailMax, a well-established online retailer running a single, legacy e-commerce platform, struggles with lengthy, error-prone release cycles: every time they update their product catalogue, shopping cart, or checkout features, they must manually deploy and configure servers, resulting in unplanned downtime during peak sales and delayed feature rollouts. To modernize, RetailMax needs a solution that can automatically migrate their monolithic application to the cloud, package and version their entire system reliably, replace manual deployment steps with a repeatable CI/CD workflow, ensure zero-downtime releases during high-traffic events, provide continuous validation before going live, and offer real-time visibility into application performance—allowing them to accelerate releases, maintain consistent environments, and scale seamlessly without manual intervention.

Key Features

- **Infrastructure Provisioning:** Use Terraform to establish an AWS environment, including VPC, EC2 instances, and security groups, for a consistent setup.
- **Containerization:** Develop a Dockerfile to encapsulate the monolithic application into a Docker container, ensuring portability.
- **CI/CD Pipeline:** Implement a Jenkins pipeline to automate building the Docker image, running tests, and deploying to AWS.
- **Configuration Management:** Employ Ansible to configure server software (e.g., database, web server) and deploy the container.
- **Monitoring:** Set up monitoring with Prometheus/Grafana to visualize resource usage and performance metrics.

Tech Stack

- **GitHub:** Version control
- **Docker:** Containerization
- **Jenkins:** CI/CD automation
- **EC2/ECS:** Deployment platform
- **Terraform:** Infrastructure as code
- **Ansible:** Configuration management
- **Prometheus/Grafana:** Monitoring tools

---

3. TravelEase: Cloud-Native Microservices Deployment

Project Overview

TravelEase, a growing travel-tech startup offering online flight booking and payment services, struggles to coordinate independent teams as they each update their own microservice—deploying new features or fixes often involves manual coordination, inconsistent environments, and service outages that frustrate both customers and developers. To overcome these challenges, TravelEase needs a fully automated end-to-end deployment pipeline that can package and version each microservice independently, run automated tests and validations before updates are released, manage seamless, zero-downtime rollouts of individual services, ensure consistent environments across development, staging, and production, enable independent scaling of services to handle variable traffic (e.g., holiday travel spikes), and provide real-time visibility into service health and performance—allowing teams to innovate rapidly without risking reliability or customer experience.

Key Features

- **Single repository for collaboration:** All microservices are managed in one GitHub repo using feature branches and pull requests.
- **Containerization:** Each service is packaged using Docker for consistency.
- **AWS Deployment:** Services are deployed on AWS ECS or EC2 using Terraform.
- **CI/CD Pipelines:** Jenkins automates the build, tagging, pushing to ECR, and deployment process.
- **Service Connectivity:** Services communicate via AWS Load Balancers and ECS Service Discovery.
- **Monitoring & Logging:** Prometheus tracks metrics, Grafana visualizes data, and CloudWatch aggregates logs and sends alerts.

Tech Stack

- **GitHub**: Version control (monorepo structure)
- **Docker**: Containerization of each microservice
- **ECS/EC2**: Container orchestration and host management
- **Terraform**: Infrastructure provisioning
- **Jenkins**: CI/CD automation
- **Prometheus**: Metrics monitoring
- **Grafana**: Visualization dashboards
- **AWS CloudWatch:** Log aggregation and alerting

---

4. InventoWare: Legacy Application Modernization with IaC

Project Overview

InventoWare, a mid-sized manufacturing firm, has been running its critical inventory-management application on-premises for over a decade, relying on manual server setups and ad-hoc deployment scripts; as a result, every update—whether a new feature or a routine bug fix—takes weeks of coordination, leads to inconsistent environments, and risks downtime during busy procurement cycles. To modernise swiftly and reliably, InventoWare needs a solution that can seamlessly migrate their monolithic Java/Python system to the cloud, automatically provision and configure infrastructure in a version-controlled manner, package the application into a standardised format for deployment, enforce automated build, test, and validation steps before each release, integrate security scans to catch vulnerabilities early, and deploy updates without human intervention—ensuring consistent environments, reduced manual effort, faster turnaround for feature releases, and robust security hygiene throughout.

Key Features

- **Infrastructure as Code:** Employ Terraform to provision AWS resources for a scalable environment.
- **Configuration Automation:** Write Ansible playbooks to install software dependencies and deploy the application consistently.
- **Containerization:** Refactor the legacy app into a Docker container, enabling deployment flexibility on AWS.
- **CI/CD Pipeline:** Set up a Jenkins pipeline to build, test, and deploy the container image, incorporating basic validation tests.
- **Security Scanning:** Integrate a tool like Clair or Trivy to scan container images for vulnerabilities before deployment.

Tech Stack

- **GitHub:** Version control
- **Ansible:** Configuration automation
- **Docker:** Containerization
- **Jenkins:** CI/CD automation
- **EC2/ECS:** Deployment platform
- **Terraform:** Infrastructure provisioning
- **Clair/Trivy:** Security

---

5. FinSureTech: DevSecOps Pipeline for Financial Web App

Project Overview

FinSureTech, a fintech startup developing a customer-facing financial web application, must meet strict regulatory and security compliance requirements. Their current delivery pipeline lacks integrated security checks, leading to late detection of vulnerabilities and delayed releases—posing risks to both customer trust and audit readiness. To address this, FinSureTech needs a secure CI/CD pipeline that embeds security at every stage of the development lifecycle. This includes early detection of vulnerabilities in both source code and open-source dependencies, runtime security analysis to catch real-time issues, secure handling of sensitive credentials, and consistent deployment environments—all without slowing down delivery. The company seeks a solution that automates these processes, ensuring rapid and secure delivery of features while staying compliant with industry standards.

Key Features

- **CI/CD with Security Gates:** Configure a Jenkins pipeline with stages for Static Application Security Testing (SAST) using SonarQube and Software Composition Analysis (SCA) with Snyk.
- **Container Scanning:** Scan Docker images with Clair or Trivy to identify vulnerabilities in the operating system or libraries.
- **Dynamic Analysis:** Use OWASP ZAP for Dynamic Application Security Testing (DAST) in a staging environment to detect runtime issues.
- **Infrastructure Provisioning:** Provision a secure AWS environment with Terraform for staging and production deployments.
- **Secrets Management:** Securely manage sensitive data (e.g., API keys) using AWS Secrets Manager or Kubernetes secrets.

Tech Stack

- **GitHub:** Version control
- **Docker:** Containerization
- **Jenkins:** CI/CD automation
- **Terraform:** Infrastructure provisioning
- **SonarQube:** Static analysis
- **OWASP ZAP:** Dynamic scanning
- **EC2/ECS:** Deployment platform
- **Synk/Clair:** Additional security tools

---

6. FinCore: Monitoring and Observability for Containerized Microservices

Project Overview

FinCore Solutions, a digital-first banking startup, is rapidly expanding its suite of containerized microservices—handling functions like instant payments, personalized investment tracking, and automated account services. As these services multiply, the engineering team struggles to maintain visibility into system performance, identify bottlenecks, and respond promptly to disruptions. To ensure reliability and customer trust, FinCore Solutions needs a comprehensive monitoring and observability solution that delivers real-time insights, intuitive dashboards, and proactive alerts—empowering the team to detect issues early and guarantee uninterrupted service delivery across all containerized microservices.

Key Features:

- **Metrics Instrumentation:** Integrate Prometheus client libraries into each microservice container to expose metrics (CPU usage, memory, request rates).

- **Prometheus Setup:** Deploy a Prometheus server to scrape metrics from container endpoints, using service discovery or static targets.
- **Grafana Dashboards:** Create Grafana dashboards to visualize key metrics like latency, error rates, and resource utilization.
- **Alerting:** Define Prometheus or Grafana alerting rules for critical conditions (e.g., high error rates), with notifications via email or Slack.
- **Infrastructure as Code:** Use Terraform to provision the monitoring stack and configure container deployment targets.

Tech Stack:

- **EC2/ECS:** Deployment platform
- **Docker:** Containerization
- **Prometheus:** Metrics collection framework
- **Grafana:** Visualization and alerting platform
- **Terraform:** Infrastructure provisioning
- **GitHub:** Version control