ΧΑΡΟΚΟΠΕΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΜΑΤΙΚΗΣ

Τμήμα Πληροφορικής και Τηλεματικής
Μεταπτυχιακό πρόγραμμα
Πληροφορική & Τηλεματική

Εργασία μαθήματος
«Cloud Native Εφαρμογές»

Επί των φοιτητών:
Πετρίδη Αχιλλέα itp24105
Χαρίτο Δημήτρη itp24108

Διδάσκοντες:
Ανάργυρος Τσαδήμας
Γεώργιος Κουσιουρής

Καλλιθέα, 2025

Πίνακας Περιεχομένων

Εισαγωγή

Η παρούσα εργασία παρουσιάζει την υλοποίηση μιας σύγχρονης cloud-native e-commerce εφαρμογής (SimpleEshop) που εκμεταλλεύεται τις τεχνολογίες και πρακτικές που διδάχθηκαν στο μάθημα "Cloud Native Εφαρμογές".

Η αρχιτεκτονική βασίζεται σε microservices που εκτελούνται σε Kubernetes cluster, με ολοκληρωμένο CI/CD pipeline, monitoring, και serverless υπολογιστικά στοιχεία. Η υλοποίηση συμπεριλαμβάνει όλες τις απαιτούμενες τεχνολογίες και πρακτικές που ορίστηκαν στην εκφώνηση της εργασίας.

Technologies used

Containerization & Orchestration

- **Docker**: Containerization όλων των microservices
- **Kubernetes (MicroK8s)**: Orchestration και management του cluster
- **Docker Hub**: Container registry για image storage

CI/CD & GitOps

- **Jenkins**: Continuous Integration pipeline
- **ArgoCD**: GitOps-based continuous deployment
- **GitHub**: Version control και webhook integration

Infrastructure as Code

- **Terraform (OpenTofu)**: Infrastructure provisioning στο Azure
- **Ansible**: Configuration management και service deployment

Monitoring & Observability

- **Grafana**: Visualization και monitoring dashboards
- **Prometheus**: Metrics collection
- **Loki**: Log aggregation και analysis

Event-Driven Architecture

- **MinIO**: Object storage με webhook capabilities
- **Mailpit**: Email service για notifications
- **Serverless Functions**: Event-triggered processing

Design Patterns Υλοποιημένα

- **Circuit Breaker**: Για fault tolerance
- **Retry Pattern**: Για resilient service calls
- **Event Sourcing**: Μέσω MinIO webhooks
- **Stateless Services**: Emailing services

Υλοποίηση

Ανάλυση Απαιτήσεων: Κατά τη φάση έναρξης πραγματοποιήθηκε εκτενής ανάλυση των απαιτήσεων της εργασίας. Προσδιορίστηκαν οι τεχνολογίες που θα χρησιμοποιηθούν και δημιουργήθηκε το αρχικό architectural design.

Τεχνολογικές Επιλογές:

- Επιλογή Kubernetes ως container orchestration platform
- Απόφαση για χρήση Azure ως cloud provider
- Καθορισμός των microservices που θα αναπτυχθούν

Σχεδιασμός Αρχιτεκτονικής: Δημιουργία high-level architecture diagram που περιλαμβάνει όλα τα components και τις μεταξύ τους σχέσεις.

Εκπόνηση Μελέτης

Infrastructure Design: Σχεδιασμός της cloud infrastructure χρησιμοποιώντας Terraform modules. Καθορισμός των virtual machines, networking configuration, και security rules.

Service Architecture: Λεπτομερής σχεδιασμός των microservices, των APIs που θα εκθέτουν, και των communication patterns μεταξύ τους.

CI/CD Pipeline Design: Σχεδιασμός της pipeline που θα υποστηρίζει automated testing, building, και deployment των εφαρμογών.

Monitoring Strategy: Καθορισμός των μετρικών που θα παρακολουθούνται.

Κατασκευή

Infrastructure Provisioning: Υλοποίηση των Terraform configurations για τη δημιουργία της Azure infrastructure. Δημιουργία των virtual machines, networks, και security groups.

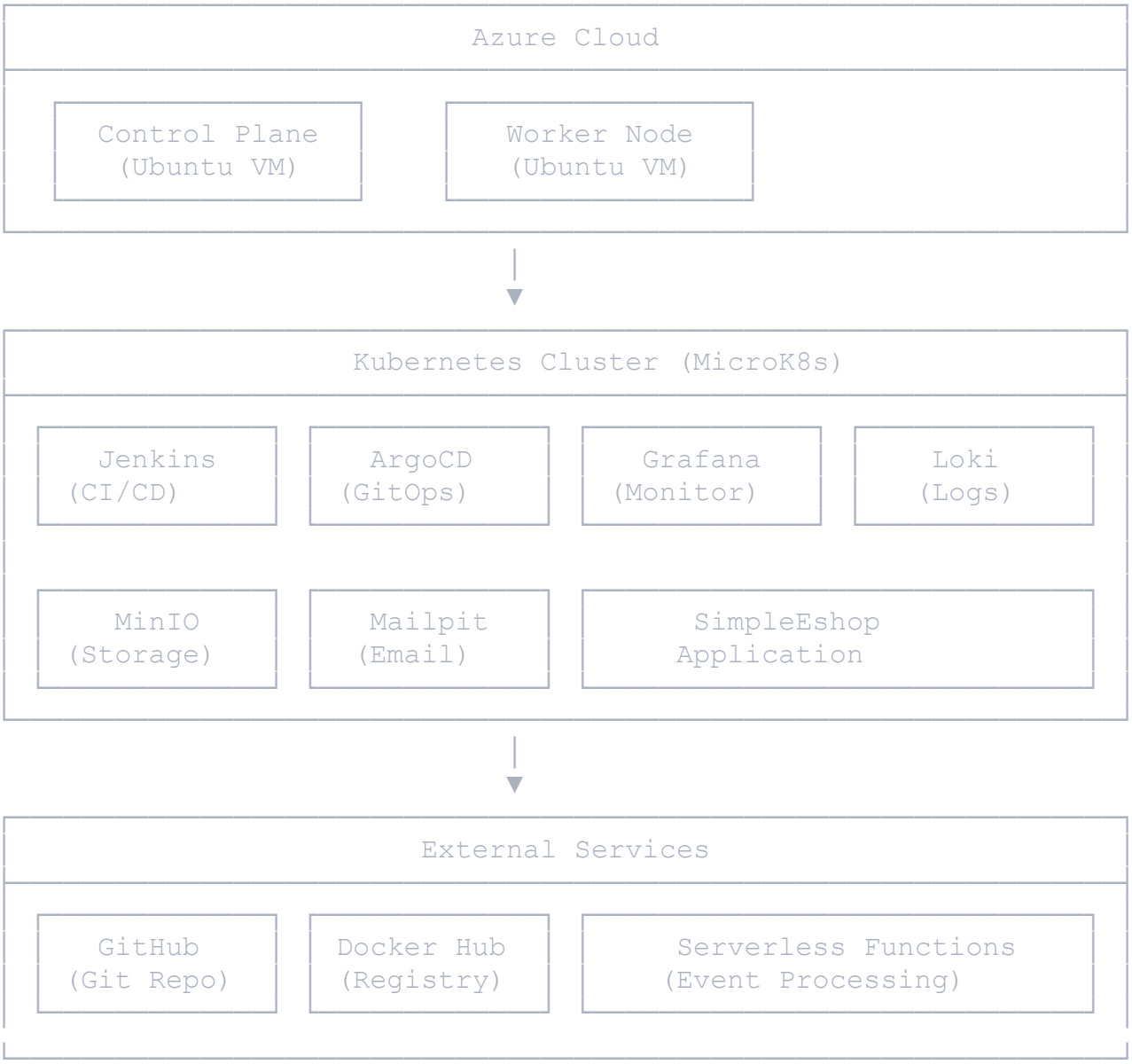
Kubernetes Cluster Setup: Εγκατάσταση και configuration του MicroK8s cluster μέσω Ansible playbooks. Setup των control plane και worker nodes.

Application Development: Ανάπτυξη των microservices που αποτελούν την SimpleEshop εφαρμογή. Δημιουργία των Dockerfiles και των Kubernetes manifests.

CI/CD Implementation: Υλοποίηση των Jenkins pipelines και configuration του ArgoCD για GitOps deployment.

6. Διαγράμματα

Αρχιτεκτονική Συστήματος



Ροή Δεδομένων

Λεπτομερής Περιγραφή Ροής:

1. **Development Workflow:**
   - Developer commits code στο GitHub repository
   - GitHub webhook triggers Jenkins pipeline
   - Jenkins builds Docker image και pushes στο Docker Hub
   - Jenkins updates Kubernetes manifests με το νέο image tag
2. **Deployment Workflow:**
   - ArgoCD monitors το GitHub repository για changes
   - ArgoCD automatically syncs και deploys τα updated manifests
   - Kubernetes scheduler deployes τα νέα pods
3. **Runtime Data Flow:**
   - User requests προς SimpleEshop application
   - Application processes orders και stores data στο MinIO
   - MinIO webhooks trigger serverless functions για data processing

- Email notifications στέλνονται μέσω Mailpit service

4. **Monitoring Flow:**
   - Application metrics συλλέγονται από Prometheus
   - Logs aggregated από Loki
   - Grafana visualizes metrics και logs για monitoring

7. Υλοποίηση

Βασικά Χαρακτηριστικά Κώδικα

Design Patterns Implementados:

1. **Circuit Breaker Pattern:**

yaml
```yaml
# Υλοποίηση στο API Gateway
resilience4j:
  circuitbreaker:
    instances:
      payment-service:
        failure-rate-threshold: 50
        wait-duration-in-open-state: 30s
```

2. **Retry Pattern:**

yaml
```yaml
# Configuration για retry logic
resilience4j:
  retry:
    instances:
      external-api:
        max-attempts: 3
        wait-duration: 1s
```

3. **Stateless Services:** Όλα τα microservices είναι stateless με external state management μέσω databases και caching services.

Event-Driven Processing:

javascript
```
// Serverless function triggered από MinIO webhook
exports.processOrder = async (event) => {
    const orderData = JSON.parse(event.body);

    // Process order metadata
    const processedOrder = await processOrderData(orderData);

    // Update MinIO object με metadata
    await updateOrderMetadata(processedOrder);

    // Trigger email notification
    await triggerEmailNotification(processedOrder);
};
```

Ρύθμιση Vm

Infrastructure Components:

1. **Azure Virtual Machines:**
    - **Control Plane VM**: Standard_B2s (2 vCPUs, 4GB RAM, Ubuntu 24.04 LTS)
    - **Worker VM**: Standard_B2s (2 vCPUs, 4GB RAM, Ubuntu 24.04 LTS)
2. **Networking Configuration:**

hcl
```
# Terraform configuration
resource "azurerm_virtual_network" "main" {
  name                = "simpleeshop-vnet"
  address_space       = ["10.0.0.0/16"]
  location            = var.location
  resource_group_name = azurerm_resource_group.main.name
}

resource "azurerm_subnet" "control_plane" {
  name                 = "control-plane-subnet"
  resource_group_name  = azurerm_resource_group.main.name
  virtual_network_name = azurerm_virtual_network.main.name
  address_prefixes     = ["10.0.1.0/24"]
}
```

3. **Security Configuration:**

yaml
```yaml
# Network Security Group rules
- name: Allow SSH
  protocol: Tcp
  source_port_range: "*"
  destination_port_range: "22"
  access: Allow

- name: Allow Kubernetes API
  protocol: Tcp
  destination_port_range: "6443"
  access: Allow

- name: Allow NodePort Services
  protocol: Tcp
  destination_port_range: "30000-32767"
  access: Allow
```

Kubernetes Cluster Configuration:

yaml
```yaml
# MicroK8s addons enabled
addons:
  - dns
  - ingress
  - storage
  - metrics-server
  - prometheus
```

Service Deployment Configuration:

yaml
```yaml
# Jenkins deployment
apiVersion: apps/v1
kind: Deployment
metadata:
  name: jenkins
spec:
  replicas: 1
  selector:
    matchLabels:
      app: jenkins
  template:
    spec:
      containers:
      - name: jenkins
        image: jenkins/jenkins:lts
        ports:
```

```
        - containerPort: 8080
        volumeMounts:
        - name: jenkins-data
          mountPath: /var/jenkins_home
```

Monitoring Configuration:

yaml

```
# Grafana dashboard configuration
apiVersion: v1
kind: ConfigMap
metadata:
  name: grafana-dashboards
data:
  simpleeshop-dashboard.json: |
    {
      "dashboard": {
        "title": "SimpleEshop Metrics",
        "panels": [
          {
            "title": "Request Rate",
            "type": "graph",
            "targets": [
              {
                "expr": "rate(http_requests_total[5m])",
                "legendFormat": "{{service}}"
              }
            ]
          }
        ]
      }
    }
```

Συμπεράσματα

Η υλοποίηση του SimpleEshop επιτυγχάνει όλους τους στόχους που τέθηκαν στην εκφώνηση της εργασίας:

1. **Τεχνολογική Κάλυψη**: Υλοποιήθηκαν 4 απαιτούμενες τεχνολογίες
2. **Design Patterns**: Εφαρμόστηκαν Circuit Breaker, Retry, Event Sourcing, και Stateless patterns
3. **Monitoring**: Παρακολούθηση μετρικών με Grafana και Prometheus
4. **Serverless**: Event-driven processing με MinIO webhooks
5. **Automation**: Πλήρης αυτοματοποίηση με Terraform και Ansible
6. **GitOps**: CI/CD με Jenkins και ArgoCD

Η εφαρμογή αποτελεί ένα πλήρες παράδειγμα σύγχρονης cloud-native αρχιτεκτονικής που μπορεί να χρησιμοποιηθεί ως reference για μελλοντικές υλοποιήσεις.