🚀 Sarawak Energy CI/CD Project - Complete Deployment Guide

📋 Prerequisites Checklist

Before starting, ensure you have:

- [ ] **Jenkins Master**: AWS EC2 instance (18.143.157.100:8080) running
- [ ] **Worker Node 1**: Linux-Docker environment on Windows
- [ ] **Worker Node 2**: AWS EC2 instance
- [ ] **Git repository** (GitHub/GitLab/Bitbucket) access
- [ ] **SSH access** to all EC2 instances
- [ ] **Admin access** to Jenkins master

🗂️ PHASE 1: Repository Setup

Step 1.1: Create Git Repository

1. **Create a new repository** on GitHub/GitLab:

   ```
   Repository Name: sarawak-energy-cicd
   Description: Sarawak Energy CI/CD Training Project
   Visibility: Private (recommended) or Public
   ```

2. **Clone repository to your local machine**:

   ```bash
   git clone https://github.com/your-username/sarawak-energy-cicd.git
   cd sarawak-energy-cicd
   ```

Step 1.2: Create Directory Structure

3. **Create the complete project structure**:

   ```bash
   # Main directories
   mkdir -p part1-verification/{templates}
   mkdir -p part2-sensor-ml/{data,models}
   mkdir -p part3-robots/{templates}
   mkdir -p jenkins-pipelines
   mkdir -p setup-scripts
   mkdir -p shared-data/{archive,logs,reports}
   mkdir -p documentation
   ```

   Your directory structure should look like:

```
sarawak-energy-cicd/
├── docker-compose.yml
├── README.md
├── part1-verification/
│   ├── Dockerfile
│   ├── requirements.txt
│   ├── app.py
│   ├── database.py
│   ├── face_verification.py
│   └── templates/
│       └── index.html
├── part2-sensor-ml/
│   ├── Dockerfile
│   ├── requirements.txt
│   ├── sensor_monitor.py
│   ├── ml_pipeline.py
│   └── data/
├── part3-robots/
│   ├── Dockerfile
│   ├── requirements.txt
│   ├── robot_controller.py
│   ├── dashboard.py
│   └── templates/
│       └── dashboard.html
├── jenkins-pipelines/
│   ├── Jenkinsfile-verification
│   ├── Jenkinsfile-sensor-ml
│   └── Jenkinsfile-robots
├── setup-scripts/
│   ├── setup-master-node.sh
│   ├── setup-worker-node-1.sh
│   └── setup-worker-node-2.sh
└── shared-data/
    ├── archive/
    ├── logs/
    └── reports/
```

Step 1.3: Add Files to Repository

4. **Copy and save each file** in the correct location: **Main Configuration:**

```bash
# Save docker-compose.yml in root directory
nano docker-compose.yml
# Copy content from the docker-compose.yml artifact
```

Part 1 Files:

```bash
# Verification system files
nano part1-verification/Dockerfile
nano part1-verification/requirements.txt
nano part1-verification/app.py
nano part1-verification/database.py
nano part1-verification/face_verification.py
nano part1-verification/templates/index.html
```

Part 2 Files:

```bash
# Sensor ML system files
nano part2-sensor-ml/Dockerfile
nano part2-sensor-ml/requirements.txt
nano part2-sensor-ml/sensor_monitor.py
nano part2-sensor-ml/ml_pipeline.py
```

Part 3 Files:

```bash
# Robot system files
nano part3-robots/Dockerfile
nano part3-robots/requirements.txt
nano part3-robots/robot_controller.py
nano part3-robots/dashboard.py
```

Jenkins Pipeline Files:

```bash
# Jenkins pipelines
nano jenkins-pipelines/Jenkinsfile-verification
nano jenkins-pipelines/Jenkinsfile-sensor-ml
nano jenkins-pipelines/Jenkinsfile-robots
```

Setup Scripts:

```bash
```

```
# Setup scripts
nano setup-scripts/setup-master-node.sh
nano setup-scripts/setup-worker-node-1.sh
nano setup-scripts/setup-worker-node-2.sh

# Make scripts executable
chmod +x setup-scripts/*.sh
```

5. **Commit and push to repository**:

```bash
git add .
git commit -m "Initial commit: Sarawak Energy CI/CD project setup"
git push origin main
```

🖥️ PHASE 2: Master Node Setup (Jenkins Master)

Step 2.1: Connect to Jenkins Master

6. **SSH into Jenkins Master EC2**:

```bash
ssh -i your-key.pem ec2-user@18.143.157.100
```

Step 2.2: Clone Repository and Setup

7. **Clone repository on master node**:

```bash
sudo yum install -y git
cd /home/ec2-user
git clone https://github.com/your-username/sarawak-energy-cicd.git
cd sarawak-energy-cicd
```

8. **Run master node setup**:

```bash
chmod +x setup-scripts/setup-master-node.sh
./setup-scripts/setup-master-node.sh
```

9. **Verify services are running**:

```bash
```

```
docker ps
# Should see PostgreSQL and Redis containers running

# Test database connection
docker exec sarawak-postgres-db psql -U postgres -d visitors -c "SELECT version();"
```

Step 2.3: Configure Jenkins Security Groups

10. **Update AWS Security Groups** for Jenkins Master:

```
Inbound Rules:
- Port 8080: Jenkins Web (0.0.0.0/0)
- Port 5432: PostgreSQL (Worker nodes IPs)
- Port 6379: Redis (Worker nodes IPs)
- Port 5001: Verification System (0.0.0.0/0)
- Port 5002: ML System (Worker nodes IPs)
- Port 5003: Robot System (0.0.0.0/0)
- Port 22: SSH (Your IP)
```

💻 PHASE 3: Worker Node 1 Setup (Linux-Docker)

Step 3.1: Connect to Worker Node 1

11. **Access your Linux-Docker environment** (Windows machine):

```bash
# Open your Linux terminal/WSL
cd ~
```

Step 3.2: Clone Repository and Setup

12. **Clone repository on worker node 1**:

```bash
git clone https://github.com/your-username/sarawak-energy-cicd.git
cd sarawak-energy-cicd
```

13. **Run worker node 1 setup**:

```bash
chmod +x setup-scripts/setup-worker-node-1.sh
./setup-scripts/setup-worker-node-1.sh
```

14. **Deploy ML system**:

```bash
```

```
cd /opt/sarawak-energy

# Clone project if not done by script
git clone https://github.com/your-username/sarawak-energy-cicd.git sarawak-project

# Deploy ML system
./deploy-ml-system.sh

# Verify deployment
curl http://localhost:5002/health
```

🌐 PHASE 4: Worker Node 2 Setup (EC2 Worker)

Step 4.1: Connect to Worker Node 2

15. **SSH into EC2 Worker Node**:

```
bash

ssh -i your-key.pem ec2-user@YOUR-WORKER-NODE-2-IP
```

Step 4.2: Clone Repository and Setup

16. **Clone repository on worker node 2**:

```
bash

git clone https://github.com/your-username/sarawak-energy-cicd.git
cd sarawak-energy-cicd
```

17. **Run worker node 2 setup**:

```
bash

chmod +x setup-scripts/setup-worker-node-2.sh
./setup-scripts/setup-worker-node-2.sh
```

18. **Deploy web services**:

```
bash
```

```
cd /opt/sarawak-energy

# Clone project if not done by script
git clone https://github.com/your-username/sarawak-energy-cicd.git sarawak-project

# Deploy web services
./deploy-web-services.sh

# Verify deployments
curl http://localhost:5001/health  # Verification system
curl http://localhost:5003/health  # Robot system
```

Step 4.3: Update Security Groups for Worker Node 2

19. **Configure security groups** for Worker Node 2:

```
Inbound Rules:
- Port 5001: Verification System (0.0.0.0/0)
- Port 5003: Robot Dashboard (0.0.0.0/0)
- Port 22: SSH (Your IP)
```

🔧 PHASE 5: Jenkins Configuration

Step 5.1: Access Jenkins

20. **Open Jenkins in browser**:

```
URL: http://18.143.157.100:8080
```

21. **Get initial admin password**:

```
bash

# SSH into Jenkins master and run:
sudo docker exec jenkins-master cat /var/jenkins_home/secrets/initialAdminPassword
```

Step 5.2: Configure Jenkins Nodes

22. **Add Worker Node 1** (Linux-Docker):
    - Go to: **Manage Jenkins → Manage Nodes and Clouds → New Node**
    - Node name: `worker`
    - Type: **Permanent Agent**
    - Configuration:

Remote root directory: /opt/sarawak-energy/jenkins-workspace
Labels: worker linux-docker ml-processing
Usage: Use this node as much as possible
Launch method: Launch agent via Java Web Start

- **Copy the secret** from the node configuration page

23. **Add Worker Node 2** (EC2 Worker):
    - Go to: **Manage Jenkins** → **Manage Nodes and Clouds** → **New Node**
    - Node name: `worker-node-ec2`
    - Type: **Permanent Agent**
    - Configuration:

      Remote root directory: /opt/sarawak-energy/jenkins-workspace
      Labels: worker-node-ec2 ec2-worker web-services
      Usage: Use this node as much as possible
      Launch method: Launch agent via Java Web Start

    - **Copy the secret** from the node configuration page

Step 5.3: Start Jenkins Agents

24. **Configure Worker Node 1 agent**:

```bash
# SSH/access Worker Node 1
cd /opt/sarawak-energy
nano start-jenkins-agent.sh

# Update the SECRET variable with the secret from Jenkins UI
SECRET="your-worker-node-1-secret-here"

# Start the agent
./start-jenkins-agent.sh
```

25. **Configure Worker Node 2 agent**:

```bash
```

```
# SSH into Worker Node 2
cd /opt/sarawak-energy
nano start-jenkins-agent.sh

# Update the SECRET variable with the secret from Jenkins UI
SECRET="your-worker-node-2-secret-here"

# Start the agent
./start-jenkins-agent.sh
```

Step 5.4: Create Jenkins Pipelines

26. **Create Verification Pipeline**:
    - **New Item** → **Pipeline** → Name: `sarawak-verification-pipeline`
    - **Pipeline Configuration**:
      - Definition: **Pipeline script from SCM**
      - SCM: **Git**
      - Repository URL: `https://github.com/your-username/sarawak-energy-cicd.git`
      - Script Path: `jenkins-pipelines/Jenkinsfile-verification`
      - **Build Triggers**: Poll SCM `H/1 * * * *` (every minute)

27. **Create ML Pipeline**:
    - **New Item** → **Pipeline** → Name: `sarawak-sensor-ml-pipeline`
    - **Pipeline Configuration**:
      - Definition: **Pipeline script from SCM**
      - SCM: **Git**
      - Repository URL: `https://github.com/your-username/sarawak-energy-cicd.git`
      - Script Path: `jenkins-pipelines/Jenkinsfile-sensor-ml`
      - **Build Triggers**: Poll SCM `H/1 * * * *` (every minute)

28. **Create Robot Pipeline**:
    - **New Item** → **Pipeline** → Name: `sarawak-robot-pipeline`
    - **Pipeline Configuration**:
      - Definition: **Pipeline script from SCM**
      - SCM: **Git**
      - Repository URL: `https://github.com/your-username/sarawak-energy-cicd.git`
      - Script Path: `jenkins-pipelines/Jenkinsfile-robots`
      - **Build Triggers**: Poll SCM `H/1 * * * *` (every minute)

🧪 PHASE 6: System Testing

Step 6.1: Verify All Services

29. **Check all service endpoints**:

```
bash

# From your browser or curl:
http://18.143.157.100:8080   # Jenkins
http://18.143.157.100:5001   # Verification System
http://18.143.157.100:5002   # ML System API
http://18.143.157.100:5003   # Robot Dashboard
```

Step 6.2: Test Individual Systems

30. **Test Verification System**:
    - Open: `http://18.143.157.100:5001`
    - Fill form with test data:

      ```
      Name: John Doe
      Floor: 3 (Engineering)
      Purpose: Testing CI/CD system
      Duration: 2 hours
      ```

    - **Allow camera access** and capture photo
    - Submit form and verify success

31. **Test ML System**:

```
bash

# Check sensor data generation
curl http://18.143.157.100:5002/sensor_data

# Check model info
curl http://18.143.157.100:5002/model_info

# Trigger manual training
curl -X POST http://18.143.157.100:5002/train_model
```

32. **Test Robot Dashboard**:
    - Open: `http://18.143.157.100:5003`
    - Verify real-time data updates
    - Check robot status and alerts

Step 6.3: Test CI/CD Pipeline Integration

33. **Manual Pipeline Triggers**:

```bash
# SSH into Jenkins master
ssh -i your-key.pem ec2-user@18.143.157.100

cd /opt/sarawak-energy/shared-data

# Create test visitor trigger
echo '{
  "visitor_id": 999,
  "name": "Test User",
  "destination_floor": 2,
  "purpose": "CI/CD Testing",
  "entry_time": "'$(date -Iseconds)'",
  "status": "approved"
}' > visitor_test_999.json

# Create ML trigger
echo "Test ML trigger" > trigger_sensor-ml-pipeline_$(date +%s).txt

# Create robot trigger
echo "Test robot trigger" > trigger_robot-pipeline_$(date +%s).txt
```

34. **Monitor Pipeline Execution**:
    - Go to Jenkins: http://18.143.157.100:8080
    - Watch pipelines execute automatically
    - Check console outputs for each pipeline
    - Verify file processing and data flow

Step 6.4: End-to-End Flow Test

35. **Complete System Test**:
    1. **Submit visitor** through verification system
    2. **Monitor Jenkins** for automatic pipeline triggers
    3. **Check robot dashboard** for visitor notifications
    4. **Wait for sensor data** and threshold alerts
    5. **Verify ML model** retraining triggers
    6. **Confirm robot responses** to threshold violations

📊 PHASE 7: Monitoring and Maintenance

Step 7.1: Set Up Monitoring

36. **Enable system monitoring**:

```
bash

# On each node, monitoring scripts are automatically set up
# Manual health checks:

# Master node:
ssh ec2-user@18.143.157.100
cd /opt/sarawak-energy
./monitor-system.sh

# Worker node 1:
cd /opt/sarawak-energy
./monitor-worker1.sh

# Worker node 2:
ssh ec2-user@worker-node-2-ip
cd /opt/sarawak-energy
./monitor-worker2.sh
```

Step 7.2: Backup Configuration

37. **Verify backup systems**:

```
bash

# Manual backups (automatic backups are scheduled):
# Master node:
./backup-system.sh

# Worker node 2:
./backup-worker2.sh
```

🎯 PHASE 8: Training Exercises

Exercise 1: Visitor Flow Test

38. **Process multiple visitors**:
    - Submit 5 different visitors through the kiosk
    - Monitor Jenkins pipeline executions
    - Verify robot notifications
    - Check data archival

Exercise 2: Threshold Alert Test

39. **Simulate sensor alerts**:

```
bash

# Modify thresholds in part2-sensor-ml/sensor_monitor.py
# Lower thresholds to trigger alerts faster:
TEMPERATURE_THRESHOLD = 70.0  # Lower from 85.0

# Commit and push changes
git add part2-sensor-ml/sensor_monitor.py
git commit -m "Lower thresholds for testing"
git push origin main

# Wait for automatic pipeline to redeploy
```

Exercise 3: Pipeline Customization

40. **Add custom features**:
    - Add email notifications to pipelines
    - Create custom deployment stages
    - Implement rollback mechanisms
    - Add integration tests

⚠️ Troubleshooting Guide

Common Issues:

1. Services Not Starting:

```
bash

# Check Docker status
docker ps
docker logs container-name

# Restart services
docker restart container-name
```

2. Jenkins Agents Not Connecting:

```
bash

# Check firewall rules
# Verify Jenkins node configuration
# Check agent logs in Jenkins UI
```

3. Database Connection Issues:

```bash
# Test from worker nodes
telnet 18.143.157.100 5432

# Check security groups
# Verify database is running on master
```

4. Pipeline Failures:

```bash
# Check Jenkins console output
# Verify file permissions on shared-data
# Check Docker socket permissions
```

📚 Additional Resources

Useful Commands:

```bash
# Service management
./manage-services.sh {start|stop|restart|status|logs|health}

# System monitoring
./monitor-worker1.sh
./monitor-worker2.sh

# Data cleanup
./cleanup-data.sh

# Manual testing
curl -X POST http://18.143.157.100:5002/train_model
curl http://18.143.157.100:5001/health
```

File Locations:

- **Master Node**: `/opt/sarawak-energy/`
- **Worker Node 1**: `/opt/sarawak-energy/`
- **Worker Node 2**: `/opt/sarawak-energy/`
- **Shared Data**: `/opt/sarawak-energy/shared-data/`
- **Logs**: `/opt/sarawak-energy/logs/`

✅ Completion Checklist

- [ ] Repository created and files committed
- [ ] Master node setup completed
- [ ] Worker node 1 setup completed
- [ ] Worker node 2 setup completed
- [ ] Jenkins agents connected
- [ ] All pipelines configured
- [ ] Services health checks passed
- [ ] End-to-end testing completed
- [ ] Monitoring systems active
- [ ] Training exercises completed

🎉 **Congratulations! Your Sarawak Energy CI/CD system is now fully operational!**

For support or questions about this deployment, refer to the individual README files in each service directory or check the system logs using the monitoring scripts.