# Implementation Plan: Campus Event Hub Deployment

**Document Purpose**: Deployment implementation plan for college project **Project**: Campus Event Hub Management System **Target**: Production deployment on VPS/Cloud Server

---

## 1. OBJECT PLAN (Software Deliverables)

### 1.1 Build Artifacts

| Component | Build Output | Description |
|---|---|---|
| **Frontend** | `/packages/client/dist/` | React application (HTML, CSS, JS bundles) |
| **Backend API** | `/packages/server/dist/` | Express.js server (compiled JavaScript) |
| **Database** | `app.db` | SQLite database with schema |

### 1.2 Deployment Package

**Package Name**: `campus-event-hub-v1.0.0.tar.gz`

**Contents**:

- Built frontend static files
- Compiled backend server code
- Database initialization scripts
- Nginx web server configuration
- PM2 process manager configuration
- Environment configuration template

---

## 2. ACTION & CONFIGURATION PLAN

### 2.1 Pre-Deployment Build

```
# Build all components
pnpm install
pnpm build

# Create deployment package
tar -czf campus-event-hub-v1.0.0.tar.gz \
  packages/client/dist/ \
  packages/server/dist/ \
  packages/notification/dist/
```

### 2.2 Server Setup & Deployment

**Step 1: Connect to Azure VM**

```
ssh pplkelompok1@57.158.27.31
```

**Step 2: Update System**

```
sudo apt update && sudo apt upgrade -y
```

**Step 3: Install Node.js (v20 LTS)**

```
curl -fsSL https://deb.nodesource.com/setup_20.x | sudo -E bash -
sudo apt install -y nodejs
```

**Step 4: Install pnpm**

```
sudo npm install -g pnpm
```

**Step 5: Install Nginx (Reverse Proxy)**

```
sudo apt install -y nginx
```

**Step 6: Clone Repository**

```
git clone https://github.com/ppl-kelompok-1/campus-event-hub.git
cd campus-event-hub
```

**Step 7: Install Dependencies**

```
pnpm install
```

**Step 8: Setup Environment Variables**

```
# Create .env file for server
vim packages/server/.env
```

**Server .env contents**:

```
PORT=3000
DATABASE_PATH=./data/app.db
JWT_SECRET=your-super-secret-jwt-key-change-in-production
JWT_EXPIRES_IN=7d

# Email Configuration (optional)
SMTP_HOST=smtp.gmail.com
SMTP_PORT=587
SMTP_USER=your-email@gmail.com
SMTP_PASS=your-app-password
EMAIL_FROM=noreply@campus-event-hub.com

# Application URL
APP_URL=http://57.158.27.31
```

```
# Create .env file for client
vim packages/client/.env
```

**Client .env contents**:

```
VITE_API_URL=http://57.158.27.31/api/v1
```

**Step 9: Build Application**

```
# Build both client and server
pnpm build
```

**Step 10: Initialize Database**

```
cd packages/server
pnpm init-superadmin
cd ../..
```

**Step 11: Install PM2 for Process Management**

```
sudo npm install -g pm2
```

**Step 12: Start Backend Server**

```
cd packages/server
pm2 start "pnpm start" --name server
cd ../..
```

**Step 13: Configure Nginx**

```
sudo vim /etc/nginx/sites-available/default
```

**Nginx Configuration**:

```
server {
    listen 80;
    server_name 57.158.27.31;

    # Frontend static files
    location / {
        root /home/pplkelompok1/campus-event-hub/packages/client/dist;
        try_files $uri $uri/ /index.html;
    }

    # API reverse proxy
    location /api/ {
        proxy_pass http://localhost:3000/api/;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection 'upgrade';
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
        proxy_cache_bypass $http_upgrade;
    }

    # Serve uploaded files
    location /uploads/ {
        alias /home/pplkelompok1/campus-event-hub/packages/server/uploads/;
    }
}
```

**Step 14: Test and Reload Nginx**

```bash
sudo nginx -t
sudo systemctl reload nginx
```

**Step 15: Setup PM2 Auto-start on Reboot**

```bash
pm2 startup
# Run the command it outputs (sudo env PATH=...)
pm2 save
```

**Step 16: Verify Deployment**

```bash
# Check PM2 processes
pm2 list

# Check Nginx status
sudo systemctl status nginx

# Test API
curl http://localhost:3000/api/v1/health

# Test from browser
# Frontend: http://57.158.27.31
# API: http://57.158.27.31/api/v1/health
```

### 2.3 Quick Reference Commands

```bash
# View logs
pm2 logs server

# Restart services
pm2 restart server

# Stop services
pm2 stop all

# Monitor resources
pm2 monit

# Update application
cd ~/campus-event-hub
git pull
pnpm install
pnpm build
pm2 restart server
```

### 2.4 Configuration Files

**Nginx Configuration**:

- Serves frontend static files
- Proxies API requests to backend
- Enables HTTPS with SSL certificate
- Serves uploaded files

**PM2 Configuration**:

- Runs backend server (2 instances, cluster mode)
- Runs notification service (1 instance)
- Auto-restart on failure
- Logs to `/logs/` directory

**Environment Variables**:

- Database path
- JWT secret key
- SMTP email settings
- Application URLs
- File upload limits

---

## 3. EVALUATION PLAN (Success Metrics)

### 3.1 Deployment Validation Checklist

**Infrastructure Validation**:

- [ ] Server accessible and configured
- [ ] Dependencies installed
- [ ] Firewall rules applied
- [ ] SSL certificate active

**Application Validation**:

- [ ] Build artifacts deployed
- [ ] Database initialized
- [ ] Services running via PM2
- [ ] Health endpoint responding

**Functional Validation**:

- [ ] Frontend loads correctly
- [ ] API responds to requests
- [ ] Login system works
- [ ] File upload functional
- [ ] Email notifications working

**Performance Validation**:

- [ ] API response time <500ms
- [ ] Frontend load time <3s
- [ ] No memory leaks
- [ ] CPU usage <50%

### 3.2 Testing Procedures

**Health Check Script**:

```
#!/bin/bash
# Check all services are running

# 1. Nginx status
systemctl is-active nginx

# 2. PM2 processes
pm2 list

# 3. API health
curl https://your-domain.com/api/v1/health

# 4. Frontend
curl https://your-domain.com

# 5. Database file exists
test -f /var/www/campus-event-hub/packages/server/data/app.db
```

**Load Testing**:

```
# Test with 100 concurrent users for 30 seconds
./load_test.sh https://your-domain.com/api/v1 100 30
```

**API Testing**:

```
# Run Postman test collection
newman run campus-event-hub-collection.json \
  --environment production.json
```

---

## 4. TARGET (Deployment Specifications)

### 4.1 Azure Virtual Machine Configuration

**Cloud Provider**: Microsoft Azure (Azure for Students)

**Subscription & Resource**:

| Setting | Value |
|---|---|
| Subscription | Azure for Students |
| Resource Group | `rg-campus-event-hub` |
| VM Name | `vm-campus-event-hub` |
| Region | East Asia |
| Availability Zone | Zone 1 (Self-selected) |
| Cost | ~$0.1320 USD/hr |

**Instance Configuration**:

| Spec | Value |
|---|---|

| | |
|---|---|
| Size | Standard D2s v3 |
| vCPUs | 2 |
| Memory | 8 GiB |
| Security Type | Standard |
| Image | Ubuntu Server 24.04 LTS - Gen2 |
| Architecture | x64 |
| Hibernation | Disabled |
| Azure Spot | No |

**Authentication**:

| Setting | Value |
|---|---|
| Type | Password |
| Username | `pplkelompok1` |

**Storage (Disks)**:

| Setting | Value |
|---|---|
| OS Disk Size | Image default |
| OS Disk Type | Standard SSD LRS |
| Managed Disks | Yes |
| Delete with VM | Enabled |
| Ephemeral Disk | No |

**Networking**:

| Setting | Value |
|---|---|
| Virtual Network | `vm-campus-event-hub-vnet` (new) |
| Subnet | `default` (10.0.0.0/24) (new) |
| Public IP | `57.158.27.31` |
| Public Inbound Ports | SSH (22), HTTP (80), HTTPS (443) |
| Accelerated Networking | On |
| Load Balancer | No |
| Delete IP/NIC with VM | Enabled |

**Management**:

| Setting | Value |
|---|---|
| Microsoft Defender | Basic (free) |

| | |
|---|---|
| Managed Identity | Off |
| Entra ID Login | Off |
| Auto-shutdown | Off |
| Backup | Disabled |
| Hotpatch | Off |
| Patch Orchestration | Image Default |

**Monitoring**:

| Setting | Value |
|---|---|
| Alerts | Off |
| Boot Diagnostics | On |
| OS Guest Diagnostics | Off |
| App Health Monitoring | Off |

**Advanced**:

| Setting | Value |
|---|---|
| Extensions | None |
| VM Applications | None |
| Cloud Init | No |
| User Data | No |
| Disk Controller | SCSI |
| Proximity Placement | None |
| Capacity Reservation | None |

### 4.2 Network Configuration

**Firewall Rules (NSG)**:

| Priority | Name | Port | Protocol | Source | Action |
|---|---|---|---|---|---|
| 300 | SSH | 22 | TCP | Any | Allow |
| 320 | HTTP | 80 | TCP | Any | Allow |
| 340 | HTTPS | 443 | TCP | Any | Allow |

**DNS Configuration**:

- Static public IP address assigned
- Configure custom domain pointing to Azure Public IP
- SSL certificate via Let's Encrypt

### 4.3 Database Configuration

- **Type**: SQLite 3 (file-based)
- **Location**: `/var/www/campus-event-hub/packages/server/data/app.db`
- **Backup**: Daily automated backups
- **Retention**: 30 days

### 4.4 Monitoring Setup

**Tools**:

- PM2 built-in monitoring
- Nginx access/error logs
- Application logs in `/logs/`
- Server monitoring (CPU, RAM, disk)

**Alert Thresholds**:

- CPU usage >80%
- RAM usage >90%
- Disk space >85%
- API response time >1000ms
- Service downtime >2 minutes

---

## 5. MEASUREMENT SUCCESS PLAN (KPIs & Metrics)

### 5.1 Performance Benchmarks

| Metric | Target | Warning | Critical |
|---|---|---|---|
| **API Response Time** | <200ms | >500ms | >1000ms |
| **Frontend Load Time** | <1.5s | >3s | >5s |
| **Database Query Time** | <50ms | >100ms | >200ms |
| **Server CPU Usage** | <50% | >70% | >85% |
| **Server RAM Usage** | <60% | >80% | >90% |
| **System Uptime** | 99.9% | <99.5% | <99% |

### 5.2 Availability Metrics

**Service Level Objectives (SLO)**:

- Monthly uptime: **99.9%** (max 43.8 minutes downtime/month)
- API availability: **99.95%** (max 21.6 minutes downtime/month)
- Successful requests: **>99.5%**
- Mean time to recovery: **<15 minutes**

**Calculation**:

```
Uptime % = (Total Time - Downtime) / Total Time × 100

Example:
Month = 43,200 minutes
Downtime = 30 minutes
Uptime = (43,200 - 30) / 43,200 × 100 = 99.93% ✓
```

### 5.3 Capacity Metrics

**Expected Load**:

- Concurrent users: 100-200 users
- Peak concurrent: 500 users
- Daily active users: 1,000-2,000
- Events created: 50-100 per day
- File uploads: 200-500 per day

**Stress Test Targets**:

- Sustain 200 concurrent users with <500ms response
- Handle 500 requests/second without errors
- Process 1,000 file uploads without memory issues
- 72-hour continuous operation stability

### 5.4 Success Measurement Results

**Week 1 Post-Deployment Validation**:

| Metric | Target | Actual | Status |
|---|---|---|---|
| Deployment Time | <6 hours | *TBD* | ⏳ |
| API Health Check | 100% pass | *TBD* | ⏳ |
| Frontend Load | <3s | *TBD* | ⏳ |
| API Tests | 100% pass | *TBD* | ⏳ |
| SSL Certificate | Valid | *TBD* | ⏳ |
| Email System | Working | *TBD* | ⏳ |

**Monthly Monitoring**:

| Month | Uptime | Avg Response | Peak Users | Incidents |
|---|---|---|---|---|
| 1 | *TBD* | *TBD* | *TBD* | *TBD* |
| 2 | *TBD* | *TBD* | *TBD* | *TBD* |
| 3 | *TBD* | *TBD* | *TBD* | *TBD* |

### 5.5 Business Metrics

**User Satisfaction**:

- System speed rating: ≥4.5/5
- Feature availability: ≥95%
- Error encounter rate: <5%

**Operational Efficiency**:

- Deployment time: <6 hours
- Issue resolution: <2 hours
- Backup/restore: <30 minutes

**Cost Efficiency**:

- Azure VM (Standard D2s v3): ~~$0.1320/hr (~~$95/month if running 24/7)
- Azure for Students credits apply
- Bandwidth: <500 GB/month

- Storage growth: <10 GB/month

---

## 6. BACKUP & DISASTER RECOVERY

### 6.1 Backup Strategy

**Automated Daily Backups**:

```bash
#!/bin/bash
# Daily backup at 2 AM
BACKUP_DIR="/var/backups/campus-event-hub"
DATE=$(date +%Y-%m-%d)

# Backup database
cp /var/www/campus-event-hub/packages/server/data/app.db \
   $BACKUP_DIR/app-db-$DATE.db

# Backup uploaded files
tar -czf $BACKUP_DIR/uploads-$DATE.tar.gz \
   /var/www/campus-event-hub/packages/server/uploads/

# Keep last 30 days only
find $BACKUP_DIR -mtime +30 -delete
```

### 6.2 Rollback Procedure

```bash
# 1. Stop services
pm2 stop all

# 2. Restore previous version
tar -xzf backups/campus-event-hub-v0.9.0.tar.gz

# 3. Restore database
cp /var/backups/campus-event-hub/app-db-YYYY-MM-DD.db \
   packages/server/data/app.db

# 4. Restart
pm2 restart all

# 5. Verify
./scripts/health-check.sh
```

---

## 7. DEPLOYMENT TIMELINE

| Phase | Duration | Tasks |
|---|---|---|
| Preparation | 2 hours | Build, create package |
| Infrastructure | 1 hour | Server setup, dependencies |
| Deployment | 1 hour | Deploy app, configure |
| Testing | 1 hour | Health checks, tests |

| | | |
|---|---|---|
| Monitoring | 30 min | Setup PM2, alerts |
| Documentation | 30 min | Update docs |
| **Total** | **6 hours** | Complete deployment |

---

## CONCLUSION

This implementation plan provides a structured approach for deploying Campus Event Hub to production. The plan includes:

1. **Software deliverables** with build artifacts and deployment package
2. **Deployment procedures** with step-by-step instructions and configurations
3. **Success validation** with comprehensive testing and metrics
4. **Target specifications** with server requirements and monitoring
5. **Performance metrics** with specific numerical targets and thresholds

**Key Success Metrics**:

- Deployment completion: <6 hours
- System uptime: 99.9%
- API response time: <200ms
- User capacity: 200 concurrent users
- Monthly cost: $20-50

**Next Steps**:

1. Connect to Azure VM: `ssh pplkelompok1@57.158.27.31`
2. Execute deployment following steps in Section 2.2
3. Run validation tests from Section 3
4. Access application at: http://57.158.27.31
5. Monitor metrics from Section 5
6. Maintain with backup strategy from Section 6