WEB PORTAL ACTIVATION SYSTEM

Complete Implementation Guide

OVERVIEW

The Web Portal Activation system allows field installers to activate cameras using **any web browser** - no mobile app required. The RPi creates a WiFi hotspot with a captive portal for easy, secure activation.

HOW IT WORKS

The Complete Flow

```
┌──────────────────────────────────────────────────────────────┐
│                    ACTIVATION FLOW                           │
└──────────────────────────────────────────────────────────────┘

OFFICE (Super Admin):
1. Pre-register camera in dashboard
2. System generates provisioning token: PT_A7X9K2M5P8Q3R7T2
3. Print QR code sticker
4. Apply sticker to RPi case
5. Ship RPi to installation site

ON-SITE (Field Installer):
1. Mount and connect RPi
2. Power on RPi
   └─> RPi detects no config.json
   └─> Automatically creates WiFi hotspot
   └─> SSID: "Camera-Setup-ShawnRPi"
   └─> Password: "Activate2025"

3. Installer connects phone/laptop to WiFi
   └─> Browser automatically opens portal
   └─> Or manually go to: http://192.168.4.1

4. Portal shows device information
   └─> Hostname, MAC, Serial (for verification)
```

```
5. Installer activates camera:
   Option A: Scan QR code with phone camera
   Option B: Manually type provisioning token

6. Click "Activate Camera"
   └─> Portal calls provisioning server
   └─> Server validates token
   └─> Server generates camera credentials
   └─> Config.json deployed to RPi

7. Activation complete!
   └─> WiFi hotspot shuts down (15 seconds)
   └─> Camera connects to main network
   └─> Camera agent starts
   └─> Shows as "Online" in dashboard

TOTAL TIME: 2-3 minutes
```

INSTALLATION ON RPI

Method 1: Automated Script (Recommended)

```bash
# 1. Download installation script
cd ~/ai-od-counter-multitenant/rpi-camera-client
wget https://your-server.com/install-provisioning-portal.sh

# Or if you have the file already:
chmod +x install-provisioning-portal.sh

# 2. Run installation
sudo ./install-provisioning-portal.sh

# Installation will:
# ✓ Install Flask, NetworkManager, dnsmasq
# ✓ Create provisioning portal script
# ✓ Set up systemd service
# ✓ Configure firewall
# ✓ Enable auto-start on boot (if no config exists)
```

Method 2: Manual Installation

```bash
# 1. Install dependencies
sudo apt update
sudo apt install -y python3-pip network-manager dnsmasq hostapd

# 2. Install Python packages
sudo pip3 install --break-system-packages flask requests

# 3. Create provisioning portal
sudo nano /opt/camera-agent/provisioning_portal.py
# [Paste the provisioning_portal.py content]
sudo chmod +x /opt/camera-agent/provisioning_portal.py

# 4. Create systemd service
sudo nano /etc/systemd/system/provisioning-portal.service
# [Paste the service file content]

# 5. Enable service
sudo systemctl daemon-reload
sudo systemctl enable provisioning-portal.service
```

TESTING THE PORTAL

Test Before Deployment

```bash
```

```
# 1. Remove existing config (if any) for testing
sudo mv /opt/camera-agent/config.json /opt/camera-agent/config.json.backup

# 2. Start provisioning portal manually
sudo systemctl start provisioning-portal

# 3. Check logs
sudo journalctl -u provisioning-portal -f

# Expected output:
# ============================================================
# PROVISIONING PORTAL READY
# ============================================================
# WiFi Network: Camera-Setup-ShawnRPi
# Password: Activate2025
# Portal URL: http://192.168.4.1
#
# Instructions:
# 1. Connect your phone/laptop to the WiFi network above
# 2. Browser should automatically open to activation portal
# 3. If not, navigate to: http://192.168.4.1
# ============================================================

# 4. On your phone/laptop:
#    - Connect to "Camera-Setup-ShawnRPi" WiFi
#    - Password: Activate2025
#    - Browser should auto-open to portal
#    - If not, go to: http://192.168.4.1

# 5. Test activation:
#    - Enter test token: TEST_TOKEN_12345
#    - (You'll need a real token from your dashboard for actual activation)

# 6. Stop portal after testing
sudo systemctl stop provisioning-portal

# 7. Restore config if needed
sudo mv /opt/camera-agent/config.json.backup /opt/camera-agent/config.json
```

SUPER ADMIN DASHBOARD INTEGRATION

Generate Provisioning Token (Dashboard)

```javascript
```

```
// Cloud Function: generateProvisioningToken
exports.generateProvisioningToken = functions.https.onCall(async (data, context) => {
  // Verify Super Admin role
  if (context.auth.token.role !== 'super_admin') {
    throw new functions.https.HttpsError('permission-denied', 'Super Admin only');
  }

  const { cameraName, siteId, orgId, expiryDays = 7 } = data;

  // Generate cryptographically secure token
  const crypto = require('crypto');
  const tokenId = crypto.randomBytes(10).toString('hex').toUpperCase();
  const provisioningToken = `PT_${tokenId}`;

  // Calculate expiry
  const expiresAt = new Date();
  expiresAt.setDate(expiresAt.getDate() + expiryDays);

  // Store in Firestore
  await db.collection('provisioningTokens').doc(provisioningToken).set({
    token: provisioningToken,
    cameraName,
    siteId,
    orgId,
    status: 'pending',
    createdAt: admin.firestore.FieldValue.serverTimestamp(),
    createdBy: context.auth.uid,
    expiresAt: expiresAt.toISOString(),
    restrictions: {
      maxUses: 1,
      usedCount: 0
    }
  });

  // Generate QR code data
  const qrData = JSON.stringify({
    action: 'provision_camera',
    token: provisioningToken,
    server: 'https://provision.yourcompany.com',
    version: '1.0'
  });

  return {
```

```
    provisioningToken,
    qrData,
    expiresAt: expiresAt.toISOString(),
    cameraName
  };
});
```

Provisioning API Endpoint

javascript

```
// Cloud Function: provisionCamera
exports.provisionCamera = functions.https.onRequest(async (req, res) => {
  // CORS headers
  res.set('Access-Control-Allow-Origin', '*');
  res.set('Access-Control-Allow-Methods', 'POST');

  if (req.method === 'OPTIONS') {
    res.status(204).send('');
    return;
  }

  if (req.method !== 'POST') {
    res.status(405).send('Method not allowed');
    return;
  }

  const { provisioningToken, deviceInfo } = req.body;

  try {
    // 1. Validate token
    const tokenDoc = await db.collection('provisioningTokens').doc(provisioningToken).get();

    if (!tokenDoc.exists) {
      res.status(404).json({ error: 'Invalid provisioning token' });
      return;
    }

    const token = tokenDoc.data();

    // Check status
    if (token.status !== 'pending') {
      res.status(400).json({ error: 'Token already used' });
      return;
    }

    // Check expiry
    if (new Date() > new Date(token.expiresAt)) {
      res.status(400).json({ error: 'Token expired' });
      return;
    }

    // Check usage limit
    if (token.restrictions.usedCount >= token.restrictions.maxUses) {
```

```
    res.status(400).json({ error: 'Token usage limit exceeded' });
    return;
  }

  // 2. Generate camera credentials
  const cameraId = `CAM_${crypto.randomBytes(6).toString('hex').toUpperCase()}`;
  const apiKey = await generateSecureApiKey(cameraId);

  // 3. Create camera record in Firestore
  await db.collection('organizations').doc(token.orgId)
    .collection('sites').doc(token.siteId)
    .collection('cameras').doc(cameraId)
    .set({
      cameraId,
      name: token.cameraName,
      serialNumber: deviceInfo.serialNumber,
      macAddress: deviceInfo.macAddress,
      hostname: deviceInfo.hostname,
      status: 'online',
      registeredAt: admin.firestore.FieldValue.serverTimestamp(),
      registeredBy: token.createdBy,
      provisioningToken: provisioningToken,
      activatedAt: admin.firestore.FieldValue.serverTimestamp()
    });

  // 4. Generate config.json
  const config = {
    cameraId,
    siteId: token.siteId,
    orgId: token.orgId,
    apiKey,
    firebaseConfig: {
      apiKey: functions.config().firebase.api_key,
      authDomain: functions.config().firebase.auth_domain,
      projectId: admin.instanceId().app.options.projectId,
      storageBucket: functions.config().firebase.storage_bucket
    },
    serviceAccount: await generateServiceAccount(cameraId),
    detectionConfig: {
      modelPath: '/opt/camera-agent/models/yolov8n.tflite',
      objectClasses: ['person', 'vehicle', 'forklift'],
      confidenceThreshold: 0.75,
      detectionZones: [],
      fps: 15,
```

```
      resolution: [1920, 1080]
    },
    transmissionConfig: {
      aggregationInterval: 300,
      maxRetries: 3,
      timeout: 10000
    }
  };

  // 5. Mark token as used
  await tokenDoc.ref.update({
    status: 'used',
    usedAt: admin.firestore.FieldValue.serverTimestamp(),
    assignedCameraId: cameraId,
    'restrictions.usedCount': admin.firestore.FieldValue.increment(1)
  });

  // 6. Log audit event
  await db.collection('auditLogs').add({
    eventType: 'CAMERA_PROVISIONED_VIA_PORTAL',
    provisioningToken,
    cameraId,
    deviceInfo,
    timestamp: admin.firestore.FieldValue.serverTimestamp()
  });

  // 7. Return config to RPi
  res.status(200).json({
    success: true,
    cameraId,
    config
  });

 } catch (error) {
  console.error('Provisioning error:', error);
  res.status(500).json({ error: 'Internal server error' });
 }
});
```

FIELD INSTALLER GUIDE

Simple Instructions for Non-Technical Staff

CAMERA ACTIVATION - FIELD GUIDE

WHAT YOU NEED:

- Raspberry Pi camera (with QR code sticker)
- Smartphone or laptop with WiFi
- Power cable and network cable

INSTALLATION STEPS:

Step 1: Physical Installation

- ☐ Mount camera at designated location
- ☐ Connect ethernet cable to network
- ☐ Connect power cable
- ☐ Camera will power on (red LED lights up)
- ⏱ Wait 60 seconds for camera to boot

Step 2: Connect to Camera WiFi

- ☐ On your phone: Settings → WiFi
- ☐ Look for network: "Camera-Setup-XXXX"
- ☐ Password: Activate2025
- ☐ Connect to this network
- ⏱ Browser should open automatically

Step 3: Activate Camera

- ☐ Portal shows camera information
- ☐ Option A: Tap "Scan QR Code"
  - • Point phone camera at QR sticker on case
  - • QR code scans automatically
- ☐ Option B: Type provisioning code
  - • Find code on QR sticker label
  - • Example: PT_A7X9K2M5P8Q3R7T2
- ☐ Click "Activate Camera" button
- ⏱ Wait for confirmation (10-30 seconds)

Step 4: Verify Activation

- ☐ Portal shows: "✓ Camera activated successfully!"
- ☐ Portal shows Camera ID (write this down)

☐ WiFi network will disappear in 15 seconds
☐ You can close browser and disconnect WiFi

Step 5: Final Verification
☐ Reconnect to regular WiFi
☐ Take photo of installed camera
☐ Report Camera ID to supervisor
☐ Mark installation as complete

TROUBLESHOOTING:

Can't see "Camera-Setup" WiFi?
→ Wait another minute, camera might still be booting
→ Power cycle camera (unplug, wait 10s, plug back in)

Browser doesn't open automatically?
→ Manually open browser and go to: 192.168.4.1

"Invalid token" error?
→ Double-check token spelling
→ Verify token hasn't expired (check with supervisor)

Activation fails?
→ Check camera has internet via ethernet cable
→ Try again (button will re-enable)
→ Contact IT support if still failing

COMPLETION:
Camera ID: _____ (from activation screen)
Installed By: _____
Date/Time: _____
Photos Taken: ☐ Yes ☐ No

SECURITY FEATURES

Token Security

- ✅ Cryptographically random (160-bit entropy)
- ✅ One-time use (invalidates after activation)
- ✅ Time-limited (7-day default expiry)

- ✅ Revocable by Super Admin at any time
- ✅ Complete audit trail
- ✅ HTTPS-only communication

Network Security

- ✅ Hotspot only active when unconfigured
- ✅ Auto-shutdown after activation
- ✅ WPA2 password protection
- ✅ 30-minute timeout if no activation
- ✅ TLS 1.3 for provisioning API calls

Physical Security

- ✅ QR sticker inside case (not externally visible)
- ✅ Token printed on destructible sticker
- ✅ Cannot reuse token if device lost/stolen

ADVANTAGES OVER MOBILE APP

Feature	Mobile App	Web Portal
Installation	Requires app download	No installation
Compatibility	iOS/Android only	Any device with WiFi
Updates	App store approval needed	Instant updates
Corporate Devices	May be blocked	Always works
Offline Use	Limited	Works offline
Training	App-specific training	Universal (browser)
Support	Platform-specific bugs	Standard web tech

DEPLOYMENT CHECKLIST

Pre-Deployment (Per RPi)

- [ ] Install provisioning portal software
- [ ] Test WiFi hotspot creation
- [ ] Verify portal accessible at 192.168.4.1
- [ ] Print QR code sticker
- [ ] Apply sticker inside RPi case
- [ ] Test full activation flow
- [ ] Document Camera ID/token pairing
- [ ] Pack and ship to site

On-Site Installation

- [ ] Mount camera
- [ ] Connect power and network
- [ ] Wait for hotspot to appear
- [ ] Connect and activate via portal
- [ ] Verify "Online" status in dashboard
- [ ] Configure detection zones (Super Admin)
- [ ] Test counting functionality
- [ ] Mark installation complete

MONITORING & LOGS

Check Provisioning Status

```bash
```

```
# View provisioning portal logs
sudo journalctl -u provisioning-portal -f

# Check if hotspot is active
nmcli connection show | grep Hotspot

# View portal status
curl http://192.168.4.1/status

# Check camera agent status
sudo systemctl status camera-agent
```

Dashboard Monitoring

Super Admin can track in real-time:

- Pending tokens (not yet used)
- Recently activated cameras
- Failed activation attempts
- Token expiration warnings

TROUBLESHOOTING

Portal Not Accessible

bash

```
# Check if service is running
sudo systemctl status provisioning-portal

# Check hotspot status
nmcli connection show Hotspot

# Restart provisioning portal
sudo systemctl restart provisioning-portal

# View detailed logs
sudo journalctl -u provisioning-portal -n 100 --no-pager
```

Hotspot Not Creating

```bash
# Check NetworkManager status
sudo systemctl status NetworkManager

# Restart NetworkManager
sudo systemctl restart NetworkManager

# Manual hotspot creation (for testing)
sudo nmcli device wifi hotspot ssid Camera-Setup password Activate2025
```

Activation Fails

```bash
# Check internet connectivity
ping -c 4 8.8.8.8

# Test provisioning server
curl -X POST https://provision.yourcompany.com/api/v1/provision \
  -H "Content-Type: application/json" \
  -d '{"test": true}'

# Check firewall
sudo ufw status

# Verify config wasn't created
ls -l /opt/camera-agent/config.json
```

SUMMARY

What This System Provides:

✅ **Zero-Install Activation** - Works with any browser ✅ **2-Minute Setup** - Fastest activation method ✅ **No Technical Skills** - Anyone can activate ✅ **Secure** - One-time tokens, encrypted transmission ✅ **Reliable** - Auto-shutdown, timeout protection ✅ **Universal** - Works on any WiFi-capable device ✅ **Auditable** - Complete activation trail

Perfect For:

- Large deployments (50+ cameras)
- Non-technical installers
- Corporate environments (app restrictions)
- Remote locations (no cell service)
- Quick replacements (swap and activate)

Deployment Time:

- First camera (with testing): 30 minutes
- Subsequent cameras: 2-3 minutes each