Deployment Guide - Sports Bar TV Controller

Overview

This guide covers deploying the Sports Bar TV Controller application with Drizzle ORM and n8n workflow automation integration.

System Requirements

Server Requirements

- **OS**: Ubuntu 20.04+ or similar Linux distribution
- **Node.js**: v18.x or higher
- **Memory**: 2GB RAM minimum (4GB recommended)
- **Storage**: 20GB available space
- **Network**: Access to Atlas processor (192.168.5.101:5321)

Hardware Integrations

- **Atlas Processor**: AtlasIED Atmosphere DSP
- IP: 192.168.5.101
- TCP Port: 5321
- Credentials: admin/6809233DjD$$$
- **Wolfpack Matrix**: Video routing matrix
- **TVs**: Connected via HDMI matrix outputs
- **Audio Zones**: Connected via Atlas processor

Installation Steps

1. Clone the Repository

```
cd /home/ubuntu
git clone https://github.com/dfultonthebar/Sports-Bar-TV-Controller.git
cd Sports-Bar-TV-Controller
```

2. Install Dependencies

```
npm install
```

3. Configure Environment Variables

Create `.env.local` file:

```
cp .env.example .env.local
nano .env.local
```

Required Environment Variables:

```
# Database Configuration
DATABASE_URL=file:./prisma/data/sports_bar.db

# Application URL
NEXT_PUBLIC_APP_URL=http://192.168.5.xxx:3001
PORT=3001

# n8n Integration
N8N_WEBHOOK_TOKEN=your-secure-random-token-here

# Atlas Processor
ATLAS_IP=192.168.5.101
ATLAS_PORT=5321
ATLAS_USERNAME=admin
ATLAS_PASSWORD=6809233DjD$$$

# Authentication (Optional)
NEXTAUTH_SECRET=your-nextauth-secret
NEXTAUTH_URL=http://192.168.5.xxx:3001

# OpenAI API (for AI features)
OPENAI_API_KEY=your-openai-key-here
```

Generate Secure Tokens:

```
# Generate N8N_WEBHOOK_TOKEN
node -e "console.log(require('crypto').randomBytes(32).toString('hex'))"

# Generate NEXTAUTH_SECRET
openssl rand -base64 32
```

4. Database Setup

```
# Generate Drizzle schema
npm run db:generate

# Push schema to database
npm run db:push

# Verify database
npm run db:studio
```

5. Build the Application

```
npm run build
```

6. Test the Application

```
# Development mode
npm run dev

# Production mode
npm start
```

Access the application at: `http://your-server-ip:3001`

Production Deployment

Option 1: PM2 (Recommended)

Install PM2 globally:

```
npm install -g pm2
```

Create PM2 ecosystem file `ecosystem.config.js`:

```
module.exports = {
  apps: [{
    name: 'sports-bar-controller',
    script: 'npm',
    args: 'start',
    cwd: '/home/ubuntu/Sports-Bar-TV-Controller',
    instances: 1,
    autorestart: true,
    watch: false,
    max_memory_restart: '1G',
    env: {
      NODE_ENV: 'production',
      PORT: 3001
    },
    error_file: '/home/ubuntu/logs/sports-bar-error.log',
    out_file: '/home/ubuntu/logs/sports-bar-out.log',
    log_file: '/home/ubuntu/logs/sports-bar-combined.log',
    time: true
  }]
}
```

Start with PM2:

```
# Start the application
pm2 start ecosystem.config.js

# Save PM2 configuration
pm2 save

# Setup PM2 to start on boot
pm2 startup

# Monitor
pm2 monit

# View logs
pm2 logs sports-bar-controller

# Restart
pm2 restart sports-bar-controller
```

Option 2: Systemd Service

Create systemd service file `/etc/systemd/system/sports-bar-controller.service`:

```
[Unit]
Description=Sports Bar TV Controller
After=network.target

[Service]
Type=simple
User=ubuntu
WorkingDirectory=/home/ubuntu/Sports-Bar-TV-Controller
Environment=NODE_ENV=production
Environment=PORT=3001
ExecStart=/usr/bin/npm start
Restart=on-failure
RestartSec=10

[Install]
WantedBy=multi-user.target
```

Enable and start:

```
sudo systemctl daemon-reload
sudo systemctl enable sports-bar-controller
sudo systemctl start sports-bar-controller
sudo systemctl status sports-bar-controller

# View logs
sudo journalctl -u sports-bar-controller -f
```

Option 3: Docker (Optional)

Create `Dockerfile`:

```
FROM node:18-alpine

WORKDIR /app

COPY package*.json ./
RUN npm ci --production

COPY . .
RUN npm run build

EXPOSE 3001

ENV NODE_ENV=production
ENV PORT=3001

CMD ["npm", "start"]
```

Build and run:

```
docker build -t sports-bar-controller .
docker run -d \
  --name sports-bar-controller \
  --restart unless-stopped \
  -p 3001:3001 \
  -v $(pwd)/prisma/data:/app/prisma/data \
  -v $(pwd)/.env.local:/app/.env.local \
  sports-bar-controller
```

Network Configuration

Firewall Rules

```
# Allow application port
sudo ufw allow 3001/tcp

# Allow SSH (if not already allowed)
sudo ufw allow 22/tcp

# Enable firewall
sudo ufw enable
```

Port Forwarding (if needed)

Configure your router to forward external requests to the server:
- External Port: 3001 → Internal: 192.168.5.xxx:3001

Static IP (Recommended)

Configure a static IP for the server to ensure consistent access.

n8n Integration Setup

Install n8n (Self-Hosted)

```
# Using Docker
docker run -d \
  --name n8n \
  --restart unless-stopped \
  -p 5678:5678 \
  -v ~/.n8n:/home/node/.n8n \
  n8nio/n8n

# Access n8n at http://your-server:5678
```

Configure n8n Webhooks

1. Create new workflow in n8n
2. Add “HTTP Request” node
3. Configure:
   - Method: POST
   - URL: `http://your-server:3001/api/n8n/webhook`
   - Headers:
     - `Content-Type`: `application/json`

- `Authorization`: `Bearer your-webhook-token`
- Body: See N8N_INTEGRATION.md (./docs/N8N_INTEGRATION.md)

Updates and Maintenance

Pulling Updates

```
cd /home/ubuntu/Sports-Bar-TV-Controller
git pull origin main
npm install
npm run build

# Restart application
pm2 restart sports-bar-controller
# OR
sudo systemctl restart sports-bar-controller
```

Database Migrations

```
# Generate new migrations
npm run db:generate

# Apply migrations
npm run db:push

# Backup database first!
cp prisma/data/sports_bar.db prisma/data/sports_bar.db.backup
```

Backup Strategy

Database Backup

```
#!/bin/bash
# backup-database.sh

DATE=$(date +%Y%m%d_%H%M%S)
BACKUP_DIR="/home/ubuntu/backups"
DB_PATH="/home/ubuntu/Sports-Bar-TV-Controller/prisma/data/sports_bar.db"

mkdir -p $BACKUP_DIR
cp $DB_PATH $BACKUP_DIR/sports_bar_$DATE.db

# Keep only last 30 backups
cd $BACKUP_DIR
ls -t sports_bar_*.db | tail -n +31 | xargs rm -f

echo "Backup completed: sports_bar_$DATE.db"
```

Add to crontab for daily backups:

```
# Edit crontab
crontab -e

# Add daily backup at 3 AM
0 3 * * * /home/ubuntu/backup-database.sh >> /home/ubuntu/logs/backup.log 2>&1
```

Application Backup

```
#!/bin/bash
# backup-application.sh

DATE=$(date +%Y%m%d_%H%M%S)
BACKUP_DIR="/home/ubuntu/backups"
APP_DIR="/home/ubuntu/Sports-Bar-TV-Controller"

tar -czf $BACKUP_DIR/app_$DATE.tar.gz \
  --exclude='node_modules' \
  --exclude='.next' \
  --exclude='.git' \
  $APP_DIR

echo "Application backup completed: app_$DATE.tar.gz"
```

Monitoring

Health Checks

Test application health:

```
# Application health
curl http://localhost:3001/api/health

# n8n webhook health
curl http://localhost:3001/api/n8n/webhook

# Atlas processor connection
curl http://localhost:3001/api/atlas/query-hardware
```

Log Management

View application logs:

```
# PM2 logs
pm2 logs sports-bar-controller --lines 100

# Systemd logs
sudo journalctl -u sports-bar-controller -n 100 -f

# Application log files (if configured)
tail -f /home/ubuntu/logs/sports-bar-combined.log
```

Performance Monitoring

```
# CPU and Memory usage
pm2 monit

# System resources
htop

# Disk usage
df -h
```

Troubleshooting

Application Won’t Start

1. Check Node.js version:
   ```bash
   node --version  # Should be v18.x or higher
   ```
2. Check port availability:
   ```bash
   sudo lsof -i :3001
   ```
3. Check environment variables:
   ```bash
   cat .env.local
   ```
4. Check logs:
   ```bash
   pm2 logs sports-bar-controller
   ```

Database Connection Issues

1. Verify database file exists:
   ```bash
   ls -la prisma/data/sports_bar.db
   ```
2. Check permissions:
   ```bash
   chmod 644 prisma/data/sports_bar.db
   ```
3. Regenerate database:
   ```bash
   rm prisma/data/sports_bar.db
   npm run db:push
   ```

Atlas Processor Connection Issues

1. Test network connectivity:
   ```bash
   ping 192.168.5.101
   telnet 192.168.5.101 5321
   ```
2. Verify credentials in `.env.local`
3. Check Atlas processor logs

n8n Webhook Failures

1. Verify webhook token:
   ```bash
   echo $N8N_WEBHOOK_TOKEN
   ```
2. Test webhook manually:
   ```bash
   curl -X POST http://localhost:3001/api/n8n/webhook \
     -H "Content-Type: application/json" \
   ```

```
    -H "Authorization: Bearer your-token" \
    -d '{"action":"health_check","data":{}}'
```

3. Check webhook logs:
   ```sql
   SELECT * FROM N8nWebhookLog ORDER BY createdAt DESC LIMIT 10;
   ```

Security Checklist

- [ ] Strong webhook tokens configured
- [ ] Environment variables not committed to git
- [ ] Firewall rules configured
- [ ] Database backups automated
- [ ] HTTPS configured (if exposed publicly)
- [ ] Regular security updates applied
- [ ] Log rotation configured
- [ ] Access logs monitored

Support

For issues or questions:
1. Check application logs
2. Review N8N_INTEGRATION.md (./docs/N8N_INTEGRATION.md)
3. Review DRIZZLE_MIGRATION_GUIDE.md (./DRIZZLE_MIGRATION_GUIDE.md)
4. Check GitHub Issues: https://github.com/dfultonthebar/Sports-Bar-TV-Controller/issues

Additional Resources

- **Next.js Documentation**: https://nextjs.org/docs
- **Drizzle ORM Documentation**: https://orm.drizzle.team
- **n8n Documentation**: https://docs.n8n.io
- **PM2 Documentation**: https://pm2.keymetrics.io
- **Atlas Processor Manual**: See `/docs/ATS006993-B-AZM4-AZM8-3rd-Party-Control.pdf`