Installation Guide

Netflix-Style Video Player - Complete Installation Guide

This comprehensive guide will help you install and configure the Netflix-style Video Player on various platforms and environments.

Table of Contents

Prerequisites

System Requirements

Before installation, ensure your system meets the minimum requirements:

Component	Minimum	Recommended
OS	Ubuntu 18.04+, Windows 10+, macOS 10.14+	Ubuntu 22.04 LTS
CPU	2 cores, 2.0 GHz	4+ cores, 3.0+ GHz
RAM	2 GB	8+ GB
Storage	10 GB free	50+ GB SSD
Network	10 Mbps upload	100+ Mbps

Required Software

- **Node.js** 18.0.0 or higher
- **NPM** 8.0.0 or higher
- **FFmpeg** 4.4.0 or higher
- **Git** (for cloning repository)

⚡ Quick Start

1. Clone Repository

```
git clone https://github.com/kailash6962/video-player.git
cd video-player
```

2. Install Dependencies

```
npm install
```

3. Install FFmpeg

```
# Ubuntu/Debian
sudo apt update && sudo apt install ffmpeg -y

# macOS
brew install ffmpeg

# Windows (using Chocolatey)
choco install ffmpeg
```

4. Configure Environment

```
cp .env.example .env
# Edit .env with your settings
```

5. Start Application

```
# Development mode
npm start

# Production mode
npm run production
```

6. Access Application

Open your browser and navigate to: `http://localhost:5555`

Platform-Specific Installation

Ubuntu/Debian Linux

Step 1: Update System

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

Step 2: Install Node.js 18

```
# Add NodeSource repository
curl -fsSL https://deb.nodesource.com/setup_18.x | sudo -E bash -

# Install Node.js
sudo apt-get install -y nodejs

# Verify installation
node --version  # Should show v18.x.x
npm --version   # Should show 8.x.x or higher
```

Step 3: Install FFmpeg

```
sudo apt install ffmpeg -y

# Verify installation
ffmpeg -version
```

Step 4: Install Git

```
sudo apt install git -y
```

Step 5: Clone and Setup

```
# Clone repository
git clone https://github.com/kailash6962/video-player.git
cd video-player

# Install dependencies
npm install

# Create environment file
cp .env.example .env
```

Step 6: Configure Environment

```
nano .env
```

Edit the following variables:

```
NODE_ENV=production
PORT=5555
VIDEO_DIR=/path/to/your/videos
ADMIN_PIN=your-admin-pin
```

Step 7: Create Video Directory

```
sudo mkdir -p /opt/videos
sudo chown $USER:$USER /opt/videos
```

Step 8: Start Application

```
# Development mode
npm start

# Or for production with PM2
sudo npm install -g pm2
pm2 start ecosystem.config.js
```

Windows 10/11

Step 1: Install Node.js

1. Download Node.js 18+ from nodejs.org
2. Run the installer and follow the wizard
3. Open Command Prompt and verify:

```
node --version
npm --version
```

Step 2: Install FFmpeg

Option A: Using Chocolatey (Recommended)

```
# Install Chocolatey first if not installed
Set-ExecutionPolicy Bypass -Scope Process -Force;
[System.Net.ServicePointManager]::SecurityProtocol =
[System.Net.ServicePointManager]::SecurityProtocol -bor 3072; iex
((New-Object System.Net.WebClient).DownloadString('https://
community.chocolatey.org/install.ps1'))
```

```
# Install FFmpeg
choco install ffmpeg -y
```

Option B: Manual Installation 1. Download FFmpeg from ffmpeg.org 2. Extract to `C:\ffmpeg` 3. Add `C:\ffmpeg\bin` to your PATH environment variable

Step 3: Install Git

Download and install Git from git-scm.com

Step 4: Clone Repository

```
git clone https://github.com/kailash6962/video-player.git
cd video-player
```

Step 5: Install Dependencies

```
npm install
```

Step 6: Configure Environment

```
copy .env.example .env
notepad .env
```

Step 7: Start Application

```
npm start
```

macOS

Step 1: Install Homebrew

```
/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/
        Homebrew/install/HEAD/install.sh)"
```

Step 2: Install Node.js

```
brew install node@18
node --version  # Should show v18.x.x
```

Step 3: Install FFmpeg

```
brew install ffmpeg
ffmpeg -version  # Verify installation
```

Step 4: Clone Repository

```
git clone https://github.com/kailash6962/video-player.git
cd video-player
```

Step 5: Install Dependencies

```
npm install
```

Step 6: Configure Environment

```
cp .env.example .env
nano .env  # or use your preferred editor
```

Step 7: Start Application

```
npm start
```

⚙ Configuration

Environment Variables (.env)

Create and configure your `.env` file:

```
# Server Configuration
NODE_ENV=production
PORT=5555

# Video Directory
VIDEO_DIR=/path/to/your/videos

# Admin Configuration
ADMIN_PIN=1234

# Database Configuration (Optional)
DB_PATH=./databases/

# Logging (Optional)
LOG_LEVEL=info
LOG_FILE=./logs/app.log

# Security (Optional)
SESSION_SECRET=your-random-secret-key
CORS_ORIGIN=http://localhost:5555
```

```
# Performance (Optional)
MAX_CONCURRENT_STREAMS=10
THUMBNAIL_CACHE_SIZE=1000
SUBTITLE_CHUNK_SIZE=600
```

Video Directory Setup

```
# Create video directory structure
mkdir -p /path/to/videos/Movies
mkdir -p /path/to/videos/Series
mkdir -p /path/to/videos/Documentaries

# Set proper permissions (Linux/macOS)
chmod 755 /path/to/videos
chmod 644 /path/to/videos/**/*.mp4
```

Database Initialization

The application will automatically create SQLite databases on first run: - `databases/home.db` - Main database - `databases/users.db` - User data (if separate)

First Run

1. Start the Application

```
npm start
```

2. Initial Setup

1. Open browser to `http://localhost:5555`
2. You'll see the user selection screen
3. Click "Add New User" to create your first user
4. Set up a username, display name, and PIN (or skip PIN)

3. Add Video Content

1. Copy your video files to the configured video directory
2. Organize them in folders (Movies, Series, etc.)
3. Restart the application to scan new content

4. Admin Panel Access

1. Navigate to `http://localhost:5555/admin`
2. Enter your admin PIN (default: 0000)
3. Configure system settings as needed

Production Deployment

Using PM2 (Recommended)

Step 1: Install PM2

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

Step 2: Configure Ecosystem

The project includes `ecosystem.config.js`. Review and modify as needed:

```
module.exports = {
  apps: [{
    name: 'video-player',
    script: 'server.js',
    instances: 'max',
    exec_mode: 'cluster',
    env: {
      NODE_ENV: 'production',
      PORT: 5555
    },
    max_memory_restart: '1G',
    watch: false,
    ignore_watch: ['node_modules', 'videos', 'logs']
  }]
};
```

Step 3: Start with PM2

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

# Save PM2 configuration
pm2 save

# Setup PM2 startup script
pm2 startup
# Follow the generated command

# Monitor application
pm2 monit
```

Using Systemd (Linux)

Step 1: Create Service File

```
sudo nano /etc/systemd/system/video-player.service
```

```
[Unit]
Description=Netflix-Style Video Player
After=network.target

[Service]
Type=simple
User=videostreamer
WorkingDirectory=/opt/video-player
ExecStart=/usr/bin/node server.js
Restart=always
RestartSec=10
Environment=NODE_ENV=production
Environment=PORT=5555

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

Step 2: Enable and Start Service

```
sudo systemctl daemon-reload
sudo systemctl enable video-player
sudo systemctl start video-player
sudo systemctl status video-player
```

Nginx Reverse Proxy

Step 1: Install Nginx

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

Step 2: Configure Nginx

```
sudo nano /etc/nginx/sites-available/video-player
```

```
server {
    listen 80;
    server_name your-domain.com;

    client_max_body_size 100M;
```

```
    location / {
        proxy_pass http://localhost:5555;
        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;

        # Video streaming optimizations
        proxy_buffering off;
        proxy_request_buffering off;
        proxy_read_timeout 300s;
        proxy_connect_timeout 75s;
    }

    # Static file caching
    location ~* \.(jpg|jpeg|png|gif|ico|css|js)$ {
        expires 1y;
        add_header Cache-Control "public, immutable";
    }
}
```

Step 3: Enable Site

```
sudo ln -s /etc/nginx/sites-available/video-player /etc/nginx/
        sites-enabled/
sudo nginx -t
sudo systemctl restart nginx
```

Docker Installation

Using Docker Compose (Recommended)

Step 1: Create docker-compose.yml

```
version: '3.8'

services:
  video-player:
    build: .
```

```
    container_name: video-player-app
    ports:
      - "5555:5555"
    volumes:
      - ./videos:/app/videos:ro
      - ./databases:/app/databases
      - ./logs:/app/logs
    environment:
      - NODE_ENV=production
      - PORT=5555
      - VIDEO_DIR=/app/videos
      - ADMIN_PIN=1234
    restart: unless-stopped
    depends_on:
      - nginx

  nginx:
    image: nginx:alpine
    container_name: video-player-nginx
    ports:
      - "80:80"
      - "443:443"
    volumes:
      - ./nginx/nginx.conf:/etc/nginx/nginx.conf:ro
      - ./nginx/ssl:/etc/nginx/ssl:ro
    restart: unless-stopped
    depends_on:
      - video-player
```

Step 2: Create Dockerfile

```
FROM node:18-alpine

WORKDIR /app

# Install FFmpeg
RUN apk add --no-cache ffmpeg

# Copy package files
COPY package*.json ./

# Install dependencies
RUN npm ci --only=production
```

```
# Copy application code
COPY . .

# Create directories
RUN mkdir -p logs databases

# Expose port
EXPOSE 5555

# Health check
HEALTHCHECK --interval=30s --timeout=3s --start-period=5s --
        retries=3 \
  CMD curl -f http://localhost:5555/api/health || exit 1

# Start application
CMD ["node", "server.js"]
```

Step 3: Build and Run

```
# Build and start
docker-compose up -d

# View logs
docker-compose logs -f

# Stop
docker-compose down
```

Using Docker Run

```
# Build image
docker build -t video-player .

# Run container
docker run -d \
  --name video-player \
  -p 5555:5555 \
  -v $(pwd)/videos:/app/videos:ro \
  -v $(pwd)/databases:/app/databases \
  -e NODE_ENV=production \
  -e ADMIN_PIN=1234 \
  --restart unless-stopped \
  video-player
```

Troubleshooting

Common Issues and Solutions

Issue: “FFmpeg not found”

Solution:

```
# Check FFmpeg installation
which ffmpeg
ffmpeg -version

# Install if missing (Ubuntu)
sudo apt install ffmpeg -y

# Add to PATH if needed (Windows)
# Add FFmpeg bin directory to system PATH
```

Issue: “Port 5555 already in use”

Solution:

```
# Find process using port
sudo lsof -i :5555  # Linux/macOS
netstat -ano | findstr :5555  # Windows

# Kill process or change port in .env
PORT=3000
```

Issue: “Permission denied accessing videos”

Solution:

```
# Fix permissions (Linux/macOS)
sudo chown -R $USER:$USER /path/to/videos
chmod -R 755 /path/to/videos
```

Issue: “Database locked” error

Solution:

```
# Stop application
pm2 stop video-player

# Check for locked database files
ls -la databases/
```

```
# Remove lock files if present
rm databases/*.db-wal databases/*.db-shm

# Restart application
pm2 start video-player
```

Issue: “Out of memory” errors

Solution:

```
# Increase Node.js memory limit
node --max-old-space-size=4096 server.js

# Or in PM2 ecosystem.config.js
node_args: ['--max-old-space-size=4096']
```

Issue: “Video won’t play”

Solution: 1. Check video format compatibility 2. Verify FFmpeg codecs:

```
ffmpeg -codecs | grep -E "(h264|hevc|vp9)"
```

1. Test video file:

```
ffmpeg -i /path/to/video.mp4 -t 10 -f null -
```

Issue: “Slow video loading”

Solutions: 1. Check network bandwidth 2. Verify storage performance 3. Enable hardware acceleration (if available) 4. Reduce concurrent streams in config

Log Analysis

```
# Application logs
tail -f logs/app.log

# PM2 logs
pm2 logs video-player

# System logs
sudo journalctl -u video-player -f

# Nginx logs
sudo tail -f /var/log/nginx/error.log
```

Verification

Post-Installation Checklist

System Verification

```
# Check Node.js version
node --version  # Should be 18+

# Check NPM version
npm --version   # Should be 8+

# Check FFmpeg
ffmpeg -version # Should be 4.4+

# Check application files
ls -la server.js package.json

# Check permissions
ls -la videos/ databases/
```

Application Testing

```
# Test server start
npm start

# Test API endpoints
curl http://localhost:5555/api/videos/home
curl -I http://localhost:5555/api/video/home/sample.mp4

# Test admin panel
curl http://localhost:5555/admin
```

Browser Testing

1. Open `http://localhost:5555`
2. Create a test user
3. Upload a sample video
4. Test video playback
5. Test subtitle loading
6. Test theme switching
7. Test mobile responsiveness

Performance Testing

```
# Test concurrent connections
ab -n 100 -c 10 http://localhost:5555/

# Monitor resource usage
htop
iotop
```

Health Check Script

Create `scripts/health-check.sh`:

```
#!/bin/bash

echo "  Video Player Health Check"
echo "============================"

# Check if server is running
if curl -f -s http://localhost:5555/ > /dev/null; then
    echo "  Server is responding"
else
    echo "  Server is not responding"
    exit 1
fi

# Check database
if [ -f "databases/home.db" ]; then
    echo "  Database exists"
else
    echo "  Database not found"
    exit 1
fi

# Check video directory
if [ -d "$VIDEO_DIR" ]; then
    echo "  Video directory accessible"
else
    echo "  Video directory not found"
    exit 1
fi

# Check FFmpeg
if command -v ffmpeg &> /dev/null; then
    echo "  FFmpeg is installed"
```

```
else
    echo "  FFmpeg not found"
    exit 1
fi

echo "  All checks passed!"
```

Next Steps

After successful installation:

1. **Read the User Manual** - Learn about all features
2. **Configure Admin Settings** - Set up user registration, limits
3. **Add Video Content** - Organize your media library
4. **Set up Backups** - Protect your data and configuration
5. **Monitor Performance** - Set up logging and monitoring
6. **Security Hardening** - Follow security best practices

Getting Help

Documentation

- **README.md** - Project overview and features
- **ARCHITECTURE.md** - Technical architecture details
- **API Documentation** - Postman collection in `documents/postman/`

Support Channels

- **GitHub Issues** - Bug reports and feature requests
- **Community Forum** - User discussions and help
- **Email Support** - Direct technical support

Useful Commands

```
# View application status
pm2 status

# Restart application
pm2 restart video-player

# View real-time logs
pm2 logs video-player --lines 100

# Monitor system resources
htop
```

```
# Test network connectivity
ping your-server.com
```

Congratulations! Your Netflix-style Video Player is now installed and ready to use!

For the complete feature overview, check out the **User Manual** and start enjoying your personal streaming platform!