ASMC Admin Panel - Installation & Setup Guide

This guide provides step-by-step instructions for installing, configuring, and setting up the ASMC Admin Panel for both development and production environments.

Table of Contents

Prerequisites

System Requirements

Minimum Requirements

- **Operating System**: Windows 10+, macOS 10.15+, or Ubuntu 18.04+
- **Node.js**: Version 18.0.0 or higher
- **npm**: Version 8.0.0 or higher
- **RAM**: 4GB minimum, 8GB recommended
- **Storage**: 2GB free space
- **Network**: Internet connection for package installation

Recommended Requirements

- **Operating System**: Windows 11, macOS 12+, or Ubuntu 20.04+
- **Node.js**: Version 18.17.0 or higher (LTS)
- **npm**: Version 9.0.0 or higher
- **RAM**: 8GB or higher
- **Storage**: 5GB free space (SSD recommended)
- **Network**: Stable internet connection

Software Dependencies

Required Software

1. **Node.js and npm**

   ```
   # Check versions
   node --version  # Should be 18.0.0+
   npm --version   # Should be 8.0.0+
   ```

2. **Git** (for version control)

   ```
   git --version
   ```

3. **ASMC Backend API** (running on port 7055)

Optional Software

- **Visual Studio Code** (recommended IDE)
- **Chrome DevTools** (for debugging)
- **Postman** (for API testing)

Development Setup

1. Clone the Repository

```
# Clone the repository
git clone <repository-url>
cd asmc-admin

# Verify you're in the correct directory
ls -la
```

2. Install Dependencies

```
# Install all dependencies
npm install

# Verify installation
npm list --depth=0
```

3. Environment Configuration

Create environment files for different stages:

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

# Create production environment file (optional)
cp .env.example .env.production
```

4. Start Development Server

```
# Start the development server
npm start

# The application will open at http://localhost:3000
```

5. Verify Installation

1. Open your browser and navigate to `http://localhost:3000`
2. You should see the ASMC Admin Panel login page
3. Check the browser console for any errors
4. Verify the API connection (check Network tab in DevTools)

Environment Configuration

Environment Variables

Create a `.env.local` file in the root directory with the following variables:

```
# API Configuration
REACT_APP_API_BASE_URL=http://localhost:7055/api
REACT_APP_API_TIMEOUT=30000

# Application Configuration
REACT_APP_APP_NAME=ASMC Admin Panel
REACT_APP_VERSION=1.0.0
REACT_APP_ENVIRONMENT=development

# Authentication
REACT_APP_JWT_STORAGE_KEY=asmc_admin_token
REACT_APP_REFRESH_TOKEN_KEY=asmc_admin_refresh
REACT_APP_TOKEN_EXPIRY_CHECK_INTERVAL=60000

# Feature Flags
REACT_APP_ENABLE_ANALYTICS=false
REACT_APP_ENABLE_LOGGING=true
REACT_APP_ENABLE_DEBUG_MODE=true
REACT_APP_ENABLE_REDUX_DEVTOOLS=true

# File Upload
REACT_APP_MAX_FILE_SIZE=10485760
REACT_APP_ALLOWED_FILE_TYPES=jpg,jpeg,png,pdf,doc,docx

# Pagination
REACT_APP_DEFAULT_PAGE_SIZE=10
REACT_APP_MAX_PAGE_SIZE=100

# Cache Configuration
REACT_APP_CACHE_DURATION=300000
REACT_APP_ENABLE_OFFLINE_MODE=false
```

Environment File Structure

```
asmc-admin/
├── .env.example          # Template file
├── .env.local           # Development environment (git ignored)
├── .env.development     # Development environment
├── .env.production      # Production environment
└── .env.test           # Test environment
```

Environment-Specific Configuration

Development Environment (`.env.local`)

```
REACT_APP_API_BASE_URL=http://localhost:7055/api
REACT_APP_ENVIRONMENT=development
REACT_APP_ENABLE_DEBUG_MODE=true
REACT_APP_ENABLE_REDUX_DEVTOOLS=true
```

Production Environment (`.env.production`)

```
REACT_APP_API_BASE_URL=https://api.asmc.com/api
REACT_APP_ENVIRONMENT=production
REACT_APP_ENABLE_DEBUG_MODE=false
REACT_APP_ENABLE_REDUX_DEVTOOLS=false
REACT_APP_ENABLE_ANALYTICS=true
```

Database Connection

Backend API Connection

The admin panel connects to the ASMC Backend API. Ensure the backend is running:

```
# Check if backend API is running
curl http://localhost:7055/api/health

# Expected response
{
  "status": "success",
  "message": "API is running",
  "timestamp": "2025-01-XX"
}
```

API Endpoints Verification

Verify these key endpoints are accessible:

```
# Authentication endpoint
curl -X POST http://localhost:7055/api/auth/login

# Members endpoint
curl http://localhost:7055/api/members

# Health check endpoint
curl http://localhost:7055/api/health
```

API Integration

Authentication Setup

1. **Login Configuration**

   ```
   // The admin panel will automatically use these endpoints:
   // POST /api/auth/login - User login
   // POST /api/auth/logout - User logout
   // POST /api/auth/refresh - Token refresh
   ```

2. **Token Management**

   - JWT tokens are automatically stored in localStorage
   - Automatic token refresh on expiration
   - Automatic logout on invalid tokens

API Base Configuration

The admin panel is configured to connect to these API endpoints:

Service	Endpoint	Description
Authentication	`/api/auth/*`	Login, logout, token refresh
Members	`/api/members/*`	Member management
Bookings	`/api/booking/*`	Booking management
Events	`/api/events/*`	Event management
Facilities	`/api/facility/*`	Facility management
Staff	`/api/staff/*`	Staff management
Common	`/api/common/*`	Common utilities
Documentation	`/api/documentation/*`	Documentation access

Production Setup

1. Build the Application

```
# Create production build
npm run build

# Verify build was created
ls -la build/
```

2. Deploy to Web Server

Option 1: Static File Hosting

```
# Copy build files to web server
cp -r build/* /var/www/html/

# Set proper permissions
chown -R www-data:www-data /var/www/html/
chmod -R 755 /var/www/html/
```

Option 2: Using Serve Package

```
# Install serve globally
npm install -g serve

# Serve the build
serve -s build -l 3000

# Or with custom port
serve -s build -l 8080
```

3. Nginx Configuration

Create `/etc/nginx/sites-available/asmc-admin`:

```
server {
    listen 80;
    server_name admin.asmc.com;
    root /var/www/html/build;
    index index.html;

    # Handle client-side routing
    location / {
        try_files $uri $uri/ /index.html;
    }

    # Cache static assets
    location /static/ {
        expires 1y;
        add_header Cache-Control "public, immutable";
    }

    # API proxy to backend
    location /api {
        proxy_pass http://localhost:7055;
        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;
    }

    # Security headers
    add_header X-Frame-Options "SAMEORIGIN" always;
    add_header X-Content-Type-Options "nosniff" always;
    add_header X-XSS-Protection "1; mode=block" always;
}
```

Enable the site:

```
# Enable the site
sudo ln -s /etc/nginx/sites-available/asmc-admin /etc/nginx/sites-enabled/

# Test configuration
sudo nginx -t

# Restart nginx
sudo systemctl restart nginx
```

4. SSL Configuration (Optional)

```
# Install Certbot
sudo apt install certbot python3-certbot-nginx
```

```
# Get SSL certificate
sudo certbot --nginx -d admin.asmc.com

# Auto-renewal
sudo crontab -e
# Add: 0 12 * * * /usr/bin/certbot renew --quiet
```

Docker Setup

1. Create Dockerfile

Create `Dockerfile` in the root directory:

```
# Multi-stage build
FROM node:18-alpine as build

# Set working directory
WORKDIR /app

# Copy package files
COPY package*.json ./

# Install dependencies
RUN npm ci --only=production

# Copy source code
COPY . .

# Build the application
RUN npm run build

# Production stage
FROM nginx:alpine

# Copy built files
COPY --from=build /app/build /usr/share/nginx/html

# Copy nginx configuration
COPY nginx.conf /etc/nginx/nginx.conf

# Expose port
EXPOSE 80

# Start nginx
CMD ["nginx", "-g", "daemon off;"]
```

2. Create nginx.conf

Create `nginx.conf`:

```
events {
    worker_connections 1024;
}
```

```
http {
    include /etc/nginx/mime.types;
    default_type application/octet-stream;

    server {
        listen 80;
        server_name localhost;
        root /usr/share/nginx/html;
        index index.html;

        location / {
            try_files $uri $uri/ /index.html;
        }

        location /static/ {
            expires 1y;
            add_header Cache-Control "public, immutable";
        }
    }
}
```

3. Build and Run Docker Container

```
# Build the Docker image
docker build -t asmc-admin .

# Run the container
docker run -d -p 3000:80 --name asmc-admin-container asmc-admin

# Verify container is running
docker ps

# Check logs
docker logs asmc-admin-container
```

4. Docker Compose Setup

Create `docker-compose.yml`:

```
version: '3.8'

services:
  asmc-admin:
    build: .
    ports:
      - "3000:80"
    environment:
      - REACT_APP_API_BASE_URL=http://localhost:7055/api
    depends_on:
      - asmc-api
    restart: unless-stopped
```

```
  asmc-api:
    image: asmc-api:latest
    ports:
      - "7055:7055"
    environment:
      - NODE_ENV=production
    restart: unless-stopped
```

Run with Docker Compose:

```
# Start services
docker-compose up -d

# Check status
docker-compose ps

# View logs
docker-compose logs -f asmc-admin
```

Troubleshooting

Common Installation Issues

1. Node.js Version Issues

Problem: "Node.js version not supported"

Solution:

```
# Check current version
node --version

# Install Node.js 18 LTS
# Using Node Version Manager (nvm)
curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.39.0/install.sh | bash
source ~/.bashrc
nvm install 18
nvm use 18

# Or download from https://nodejs.org/
```

2. npm Install Failures

Problem: "npm install" fails with permission errors

Solution:

```
# Clear npm cache
npm cache clean --force

# Remove node_modules and package-lock.json
rm -rf node_modules package-lock.json
```

```
# Reinstall with correct permissions
npm install

# Or use yarn instead
npm install -g yarn
yarn install
```

3. Port Already in Use

Problem: "Port 3000 is already in use"

Solution:

```
# Kill process using port 3000
lsof -ti:3000 | xargs kill -9

# Or use different port
PORT=3001 npm start

# Or change port in package.json
"scripts": {
  "start": "PORT=3001 react-scripts start"
}
```

4. API Connection Issues

Problem: "Cannot connect to API"

Solution:

```
# Check if backend API is running
curl http://localhost:7055/api/health

# Check environment variables
cat .env.local

# Verify API URL in browser DevTools Network tab
```

Development Issues

1. Hot Reload Not Working

Problem: Changes not reflecting in browser

Solution:

```
# Clear browser cache
Ctrl+Shift+R (hard refresh)

# Restart development server
npm start

# Check for syntax errors in console
```

2. Redux DevTools Not Working

Problem: Redux DevTools extension not connecting

Solution:

```
# Check if extension is installed
# Verify REACT_APP_ENABLE_REDUX_DEVTOOLS=true in .env.local

# Check store configuration
# Ensure devTools: true in store config
```

3. Build Failures

Problem: "npm run build" fails

Solution:

```
# Check for TypeScript errors
npm run build 2>&1 | grep -i error

# Check for unused imports
npm install -g eslint
npx eslint src/

# Clear build directory
rm -rf build/
npm run build
```

Production Issues

1. 404 Errors on Refresh

Problem: Page shows 404 when refreshed

Solution:

```
# Ensure nginx configuration includes:
location / {
    try_files $uri $uri/ /index.html;
}
```

2. API CORS Issues

Problem: CORS errors in production

Solution:

```
// Check backend CORS configuration
// Ensure admin panel domain is allowed

// Check environment variables
REACT_APP_API_BASE_URL=https://api.asmc.com/api
```

3. Performance Issues

Problem: Slow loading times

Solution:

```
# Analyze bundle size
npm install -g webpack-bundle-analyzer
npm run build
npx webpack-bundle-analyzer build/static/js/*.js

# Enable gzip compression in nginx
gzip on;
gzip_types text/css application/javascript application/json;
```

Common Issues

Issue 1: Module Not Found

Error: `Module not found: Can't resolve 'module-name'`

Solution:

```
# Install missing dependency
npm install module-name

# Or check import path
import Component from './components/Component';
```

Issue 2: Memory Issues

Error: `JavaScript heap out of memory`

Solution:

```
# Increase Node.js memory limit
NODE_OPTIONS="--max-old-space-size=4096" npm start

# Or in package.json
"scripts": {
  "start": "NODE_OPTIONS='--max-old-space-size=4096' react-scripts start"
}
```

Issue 3: Authentication Loop

Problem: Stuck in login loop

Solution:

```
# Clear localStorage
localStorage.clear();

# Check token expiration
# Verify refresh token logic

# Check API authentication endpoint
curl -X POST http://localhost:7055/api/auth/login
```

Support

For additional support:

1. **Check the logs**: Browser console and terminal output
2. **Verify prerequisites**: Node.js, npm versions
3. **Check environment**: API connectivity and configuration
4. **Review documentation**: Architecture and component guides
5. **Contact support**: Development team for complex issues

Version: 1.0.0
Last Updated: January 2025
Maintainer: ASMC Development Team