# Run-time Requirements for React.js / TypeScript / MUI / Webpack Web Application on Windows Server

## 1. Windows Server Version

* Windows Server 2016, 2019, or 2022 (Latest recommended)
* Ensure all necessary updates are installed

## 2. Required Software

### A. Node.js & NPM

* Install Node.js LTS
* Verify installation:

```sh

node -v

npm -v

```

* Used for managing dependencies and building the React app

### B. IIS (Internet Information Services) Configuration

* Enable IIS and required features:

- URL Rewrite Module (required for SPA routing)

- Application Request Routing

- Windows Authentication (internal VPN access)

- Static Content (for serving React build files)

* \*\*React Application Hosting\*\*:

- IIS will serve the React application directly to browsers

- React build output location: `%SystemDrive%\inetpub\wwwroot\react-app`

- Application files to serve:

```

/build/

├── index.html # Entry point

├── static/ # Build output

│ ├── css/ # Compiled CSS

│ ├── js/ # Compiled JavaScript

│ └── media/ # Assets

└── favicon.ico # Site icon

```

- Configure IIS to:

1. Serve static files from the build directory

2. Route all requests to index.html for SPA support

3. Set appropriate MIME types for .js, .css, and other static files

4. Enable compression for text-based files

* \*\*Web.config Requirements\*\*:

```xml

<configuration>

<system.webServer>

<!-- Enable static file serving -->

<staticContent>

<mimeMap fileExtension=".js" mimeType="application/javascript" />

<mimeMap fileExtension=".css" mimeType="text/css" />

<mimeMap fileExtension=".json" mimeType="application/json" />

</staticContent>

<!-- SPA routing support -->

<rewrite>

<rules>

<rule name="SPA Routes" stopProcessing="true">

<match url=".\*" />

<conditions logicalGrouping="MatchAll">

<add input="{REQUEST\_FILENAME}" matchType="IsFile" negate="true" />

<add input="{REQUEST\_FILENAME}" matchType="IsDirectory" negate="true" />

</conditions>

<action type="Rewrite" url="index.html" />

</rule>

</rules>

</rewrite>

</system.webServer>

</configuration>

```

## 3. URL and Routing Requirements

### A. URL Structure Preservation

* Must maintain existing URL patterns exactly for production:

```

Production: https://{tenant}-adminsvc.cfssinternal.com/ClientPortal/Login

Test: https://{tenant}-adminsvc-test.cfssinternal.com/ClientPortal/Login

```

* During pre-production testing phase, use port 8443:

```

Pre-production Testing: https://{tenant}-adminsvc.cfssinternal.com:8443/ClientPortal/Login

Pre-production Test: https://{tenant}-adminsvc-test.cfssinternal.com:8443/ClientPortal/Login

```

* No URL changes permitted during final production cutover
* All existing deep links must continue to function

### B. IIS Configuration Requirements

1. \*\*Site Structure\*\*:

- Maintain existing IIS sites and bindings

- Configure URL rewrite rules for routing traffic

- \*\*Host Header Bindings Required\*\*:

```

Production Site Bindings:

- Type: https

- IP: All Unassigned

- Port: 443

- Host name: \*-adminsvc.cfssinternal.com

- Certificate: Use existing wildcard cert for \*.cfssinternal.com

Test Site Bindings:

- Type: https

- IP: All Unassigned

- Port: 443

- Host name: \*-adminsvc-test.cfssinternal.com

- Certificate: Use existing wildcard cert for \*.cfssinternal.com

Pre-production Testing Bindings:

- Type: https

- IP: All Unassigned

- Port: 8443

- Host name: \*-adminsvc.cfssinternal.com

- Host name: \*-adminsvc-test.cfssinternal.com

- Certificate: Use existing wildcard cert for \*.cfssinternal.com

```

2. \*\*SSL Certificate Configuration\*\*:

a. \*\*Existing Certificates\*\*:

- Reuse existing wildcard certificate for \*.cfssinternal.com

- Certificate must be valid for:

```

\*.cfssinternal.com

\*-adminsvc.cfssinternal.com

\*-adminsvc-test.cfssinternal.com

```

- No new certificate installation required

b. \*\*IIS Binding Configuration\*\*:

```powershell

# View existing certificate bindings

netsh http show sslcert

# For new port 8443 binding, use existing certificate:

$existing = Get-ChildItem -Path Cert:\LocalMachine\My |

Where-Object { $\_.Subject -like "\*cfssinternal.com\*" } |

Select-Object -First 1

# If needed, create new binding for port 8443

$appid = [System.Guid]::NewGuid().ToString("B")

$certHash = $existing.Thumbprint

# Add SSL binding for port 8443

netsh http add sslcert `

ipport=0.0.0.0:8443 `

certhash=$certHash `

appid=$appid `

certstorename=MY

```

c. \*\*Verify SSL Bindings\*\*:

```powershell

# Verify all required bindings

$requiredPorts = @(443, 8443)

$bindings = netsh http show sslcert

foreach ($port in $requiredPorts) {

if ($bindings -notlike "\*0.0.0.0:$port\*") {

Write-Error "Missing SSL binding for port $port"

}

}

```

3. \*\*Host Header-Based Routing\*\*:

- Single IIS website handles all tenant traffic

- Tenant identification through host headers

- Wildcard SSL certificate required for \*.cfssinternal.com

- Host header pattern: `{tenant-id}-adminsvc.cfssinternal.com`

- Example host headers to configure:

```

1stnorcalcu68668-adminsvc.cfssinternal.com

firstcoastccu68524-adminsvc.cfssinternal.com

firstcommunitycu68530-adminsvc.cfssinternal.com

onthegridfinancial383-adminsvc.cfssinternal.com

```

4. \*\*URL Rewrite Rules\*\*:

```xml

<!-- Testing Phase Configuration -->

<rule name="React App Testing Route">

<match url=".\*" />

<conditions>

<add input="{REMOTE\_ADDR}" pattern="^(TEST\_IP\_RANGES)$" />

</conditions>

<action type="Rewrite" url="/react-app/{R:0}" />

</rule>

<!-- Production Configuration (Post-Cutover) -->

<rule name="React App Production Route">

<match url=".\*" />

<action type="Rewrite" url="/react-app/{R:0}" />

</rule>

```

5. \*\*SSL Requirements\*\*:

- Maintain existing SSL certificates

- Ensure coverage for all tenant subdomains

- Valid for both production and test environments

## 4. Deployment Strategy

### A. Manual Deployment Instructions

1. \*\*Build Process\*\* (Development Team):

```bash

# In the project directory

npm install

npm run build

# Create deployment package

cd build

zip -r ../react-app-deployment.zip ./\*

```

- Provide the following files to infrastructure team:

1. `react-app-deployment.zip` (contains build output)

2. `web.config` (provided separately for easy access)

2. \*\*Deployment Process\*\* (Infrastructure Team):

a. \*\*Backup Existing Deployment\*\* (if exists):

```powershell

# Create timestamped backup

$timestamp = Get-Date -Format "yyyyMMdd\_HHmmss"

Rename-Item `

-Path "C:\inetpub\wwwroot\react-app" `

-NewName "react-app\_backup\_$timestamp" `

-ErrorAction SilentlyContinue

```

b. \*\*Prepare Deployment Directory\*\*:

```powershell

# Create fresh directory

New-Item -ItemType Directory -Path "C:\inetpub\wwwroot\react-app" -Force

```

c. \*\*Deploy Files\*\*:

```powershell

# Extract build files

Expand-Archive `

-Path "path\to\react-app-deployment.zip" `

-DestinationPath "C:\inetpub\wwwroot\react-app" `

-Force

# Copy web.config

Copy-Item `

-Path "path\to\web.config" `

-Destination "C:\inetpub\wwwroot\react-app\web.config" `

-Force

```

d. \*\*Set Permissions\*\*:

```powershell

# Set IIS\_IUSRS permissions

$acl = Get-Acl "C:\inetpub\wwwroot\react-app"

$accessRule = New-Object System.Security.AccessControl.FileSystemAccessRule(

"IIS\_IUSRS",

"ReadAndExecute",

"ContainerInherit,ObjectInherit",

"None",

"Allow"

)

$acl.SetAccessRule($accessRule)

Set-Acl "C:\inetpub\wwwroot\react-app" $acl

```

3. \*\*Verify Deployment\*\*:

a. \*\*Check File Structure\*\*:

```powershell

# Verify all required files are present

$requiredFiles = @(

"index.html",

"web.config",

"static\css",

"static\js"

)

$missingFiles = $requiredFiles | Where-Object {

-not (Test-Path "C:\inetpub\wwwroot\react-app\$\_")

}

if ($missingFiles) {

Write-Error "Missing required files: $($missingFiles -join ', ')"

}

```

b. \*\*Verify Permissions\*\*:

```powershell

# Check IIS\_IUSRS access

Get-Acl "C:\inetpub\wwwroot\react-app" |

Select-Object -ExpandProperty Access |

Where-Object IdentityReference -like "\*IIS\_IUSRS\*"

```

c. \*\*Test URLs\*\* (using internal network/VPN):

- Pre-production test URLs:

```

https://1stnorcalcu68668-adminsvc.cfssinternal.com:8443/ClientPortal/Login

https://1stnorcalcu68668-adminsvc-test.cfssinternal.com:8443/ClientPortal/Login

```

- Verify:

1. Page loads without errors

2. Static assets (CSS/JS) load correctly

3. Application routing works

4. Host header-based tenant identification functions

4. \*\*Rollback Procedure\*\* (if needed):

```powershell

# Remove current deployment

Remove-Item -Path "C:\inetpub\wwwroot\react-app" -Recurse -Force

# Restore from backup

$latestBackup = Get-ChildItem -Path "C:\inetpub\wwwroot" |

Where-Object Name -like "react-app\_backup\_\*" |

Sort-Object Name -Descending |

Select-Object -First 1

Rename-Item `

-Path $latestBackup.FullName `

-NewName "react-app"

```

### B. Testing Phase

1. Deploy React app alongside existing Outsystems application

2. Configure IIS bindings for port 8443

3. Configure IIS URL rewrite rules to route test traffic

4. Pre-production testing URLs:

```

1st NorCal:

https://1stnorcalcu68668-adminsvc.cfssinternal.com:8443/ClientPortal/Login

https://1stnorcalcu68668-adminsvc-test.cfssinternal.com:8443/ClientPortal/Login

First Coast Community:

https://firstcoastccu68524-adminsvc.cfssinternal.com:8443/ClientPortal/Login

https://firstcoastccu68524-adminsvc-test.cfssinternal.com:8443/ClientPortal/Login

First Community Credit Union:

https://firstcommunitycu68530-adminsvc.cfssinternal.com:8443/ClientPortal/Login

https://firstcommunitycu68530-adminsvc-test.cfssinternal.com:8443/ClientPortal/Login

OnTheGrid FCU:

https://onthegridfinancial383-adminsvc.cfssinternal.com:8443/ClientPortal/Login

https://onthegridfinancial383-adminsvc-test.cfssinternal.com:8443/ClientPortal/Login

```

5. Maintain existing application for production traffic

6. Test with specific IPs/users while others use legacy system

### C. Production Cutover

1. Schedule maintenance window

2. Update IIS rewrite rules to route all traffic to React app

3. Keep legacy system available as fallback

4. No URL changes required for end users

## 5. Multi-Tenant Configuration

### A. Tenant Identification

* Extract tenant information from hostname
* Current production URLs that must be supported:

```

1st NorCal:

https://1stnorcalcu68668-adminsvc.cfssinternal.com/ClientPortal/Login

First Coast Community:

https://firstcoastccu68524-adminsvc.cfssinternal.com/ClientPortal/Login

First Community Credit Union:

https://firstcommunitycu68530-adminsvc.cfssinternal.com/ClientPortal/Login

OnTheGrid FCU:

https://onthegridfinancial383-adminsvc.cfssinternal.com/ClientPortal/Login

```

* Current test environment URLs that must be supported:

```

1st NorCal:

https://1stnorcalcu68668-adminsvc-test.cfssinternal.com/ClientPortal/Login

First Coast Community:

https://firstcoastccu68524-adminsvc-test.cfssinternal.com/ClientPortal/Login

First Community Credit Union:

https://firstcommunitycu68530-adminsvc-test.cfssinternal.com/ClientPortal/Login

OnTheGrid FCU:

https://onthegridfinancial383-adminsvc-test.cfssinternal.com/ClientPortal/Login

```

### B. Environment-Specific Requirements

* Configure separate IIS sites for test and production
* Maintain consistent URL patterns across environments
* Environment-specific configuration via web.config transforms

## 6. Monitoring & Logging

* Standard IIS logs
* Windows Event Viewer
* Application-specific logging
* VPN access logs

## 7. Security Considerations

* Internal VPN-only access
* Windows Authentication
* Existing security protocols remain in place