



# **CORTEX Interaction Portal**

## Deployment Guide

WE ARE  
**CORTEX**

# Contents

|  |    |
|--|----|
| 1 Introduction .....                                     | 4  |
| 1.1 AppGyver Overview .....                              | 4  |
| 1.2 High-Level Architecture .....                        | 5  |
| 2 Setup and Deployment .....                             | 10 |
| 2.1 Pre-Requisites .....                                 | 10 |
| 2.2 Deployment Package Preparation .....                 | 10 |
| 2.3 AppGyver Account Setup .....                         | 11 |
| 2.4 App Variables Setup .....                            | 12 |
| 2.5 Theme Variables .....                                | 17 |
| 2.6 Import CORTEX Flows .....                            | 18 |
| 2.7 Config Flow Configuration .....                      | 19 |
| 2.8 Publish CORTEX Flows .....                           | 23 |
| 2.9 Setup OAuth Published for Flows .....                | 23 |
| 2.10 Initialise Shared Data (Reliable Collections) ..... | 24 |
| 2.11 Website Deployment .....                            | 30 |
| 2.12 Web Application Deployment .....                    | 31 |
| 2.13 Web App Redirect Rule .....                         | 32 |
| 2.14 CORS Configuration .....                            | 33 |
| 2.15 Housekeeping Tasks .....                            | 34 |
| 3 Testing .....  | 36 |
| 3.1 Testing User Access Management Flows .....           | 36 |
| 3.2 Testing the Web Application .....                    | 39 |
| 3.3 Testing UI-Driven Process (Service Requests) .....   | 39 |
| 3.4 Testing Process-Driven UI (Process Dashboard) .....  | 40 |
| 4 Appendix A: Renaming Web App Title .....               | 42 |
| 5 Appendix B: Issues with Self Signed Certificates ..... | 43 |
| 6 Appendix C: Load Balancer Configurations .....         | 44 |
| 6.1 Single-Portal Setup .....                            | 44 |
| 6.2 Multi-Portal Setup .....                             | 46 |

# About this Document

This document provides a general guide to the deployment of the CORTEX Interaction Portal, including the steps to configure and deploy alongside a CORTEX Innovation instance.

## Audience

This document is intended for developers who intend to install the CORTEX Interaction Portal alongside a CORTEX Innovation instance.

## Related Material

| Document                                  |
|---|
| CORTEX Interaction Portal Developer Guide |
| CORTEX Interaction Portal Merging Guide   |
| CORTEX Interaction Portal User Guide      |

## Referenced Files

| Document  | Description   |
|---|---|
| Cortex.Interaction.Portal.zip.gpg                 | AppGyver Solution package (to import into the AppGyver development environment) |
| User.Access.Management.studiopkg                  | CORTEX Innovation flows: User Access Management                                 |
| Interaction.Portal.Core.studiopkg                 | CORTEX Innovation flows: Portal and Management (for AppGyver integration)       |
| Process.Management.Extension.studiopkg            | CORTEX Innovation flows: Add-On for Process-Driven Flows components             |
| Cortex.Interaction.Portal.postman_collection.json | Postman collection for testing CORTEX Innovation flows                          |


## Abbreviations used in this Document

|             |                              |
|-------------|------------------------------|
| <b>UI</b>   | User Interface               |
| <b>UX</b>   | User Experience              |
| <b>IIS</b>  | Internet Information Service |
| <b>RBAC</b> | Role Based Access Control    |

# 1 Introduction

## 1.1 AppGyver Overview

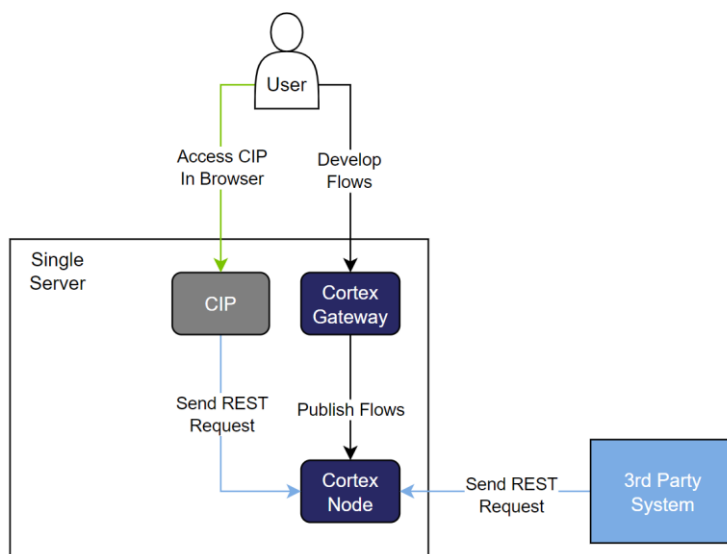
The CORTEX Interaction Portal is built using SAP AppGyver: a low-code web (and mobile) application building tool which offers pre-built *components* and the ability to create your own component templates. Also included is integrated logic to issue API requests, navigate pages, show data, and many more options.

 *AppGyver is offered in several different pricing tiers. At the time of writing the Community Edition includes all the core functionality required.*

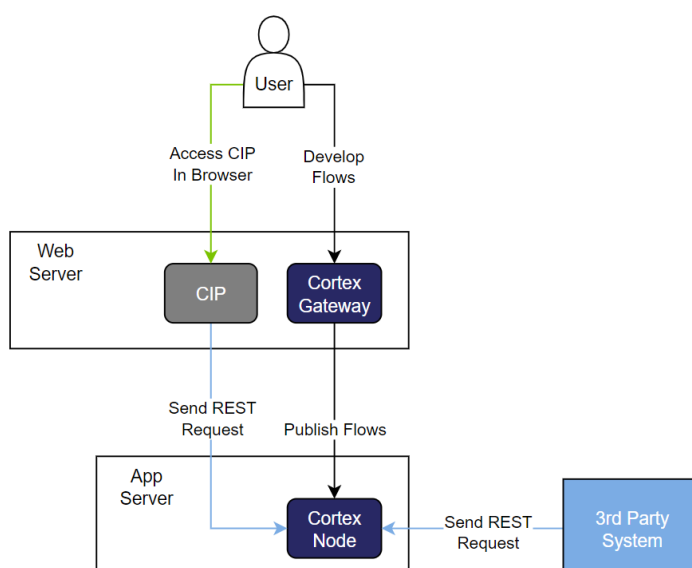
## 1.2 High-Level Architecture

### Single Node

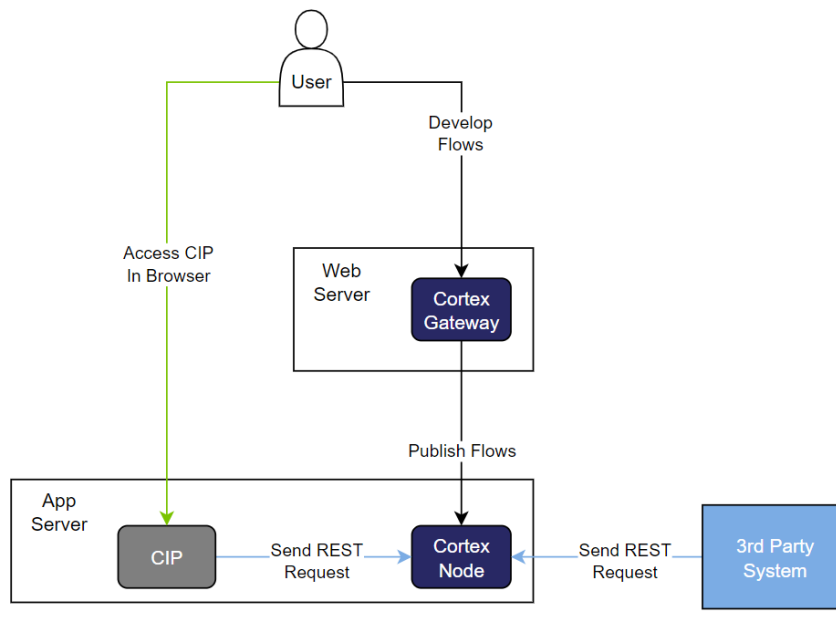
The simplest possible architecture for a CORTEX Interaction Portal is to install it alongside CORTEX Gateway and a single CORTEX node on one server. It may be accessed by a user alongside CORTEX Gateway and interacts with the flows published to the CORTEX node via its API Gateway Service.



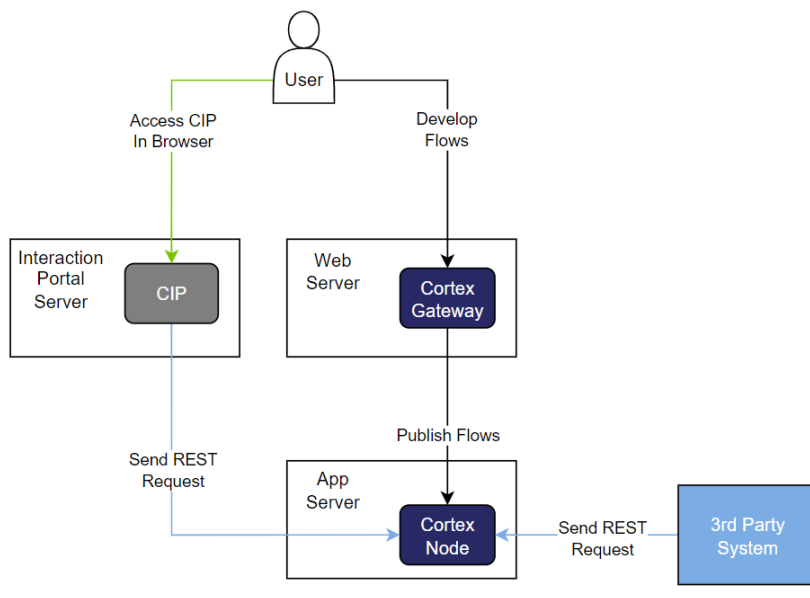
While this may serve for a simple proof-of-concept or development environment, often CORTEX Gateway is installed on its own dedicated web server. The CORTEX Interaction Portal may be installed alongside it here.



Alternatively, it may be installed alongside the CORTEX Node on the Application Server.



An Interaction Portal may be installed on its own dedicated server as well.

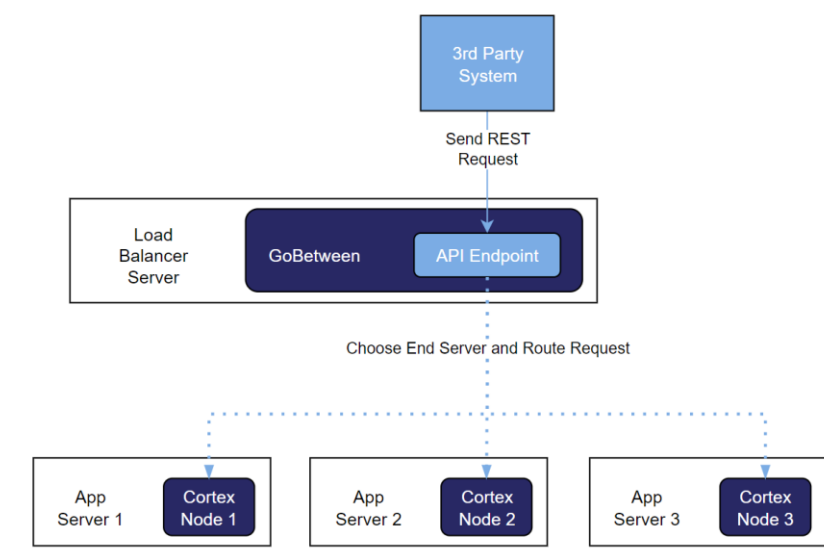


## Multi Node

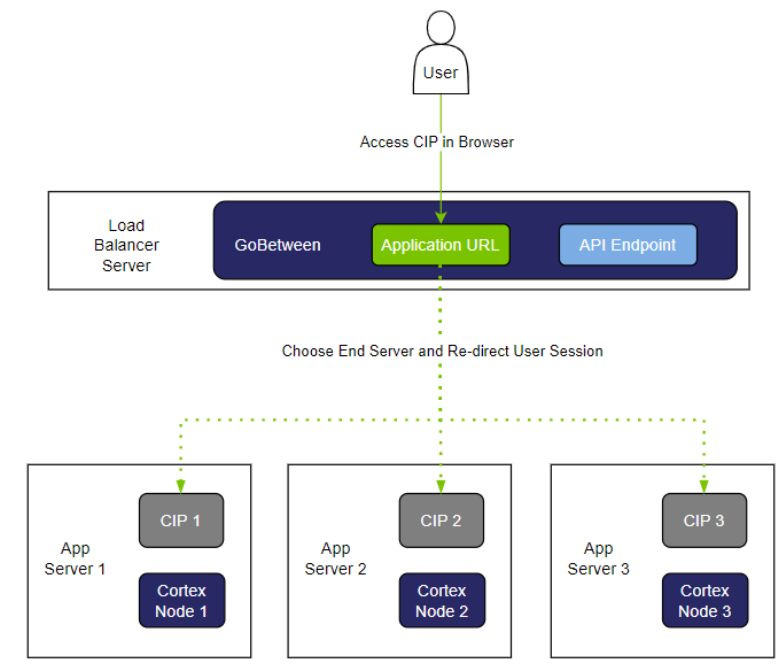
The below is the default set-up for a three node CORTEX environment without an Interaction Portal.

*For simplicity, the web server hosting CORTEX Gateway has been omitted.*

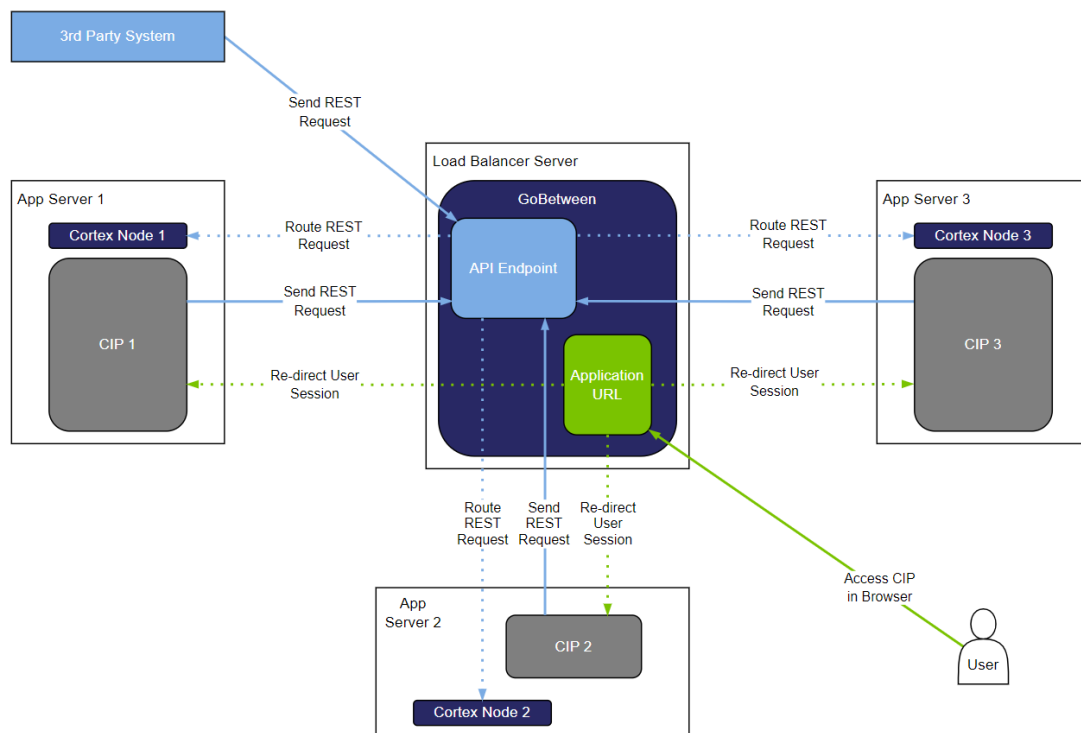
During a standard CORTEX installation including a Load Balancer server, the GoBetween service is installed there in order to route each inbound REST requests to one of the three nodes.



Similarly, if a CORTEX Interaction Portal web application is deployed to each of the three application servers, then it is possible to configure the load balancer to also route a user's session to one of them, as below:



Combining these diagrams, we may see the full interactions between the CORTEX Interaction Portals, the CORTEX Nodes, and the Load Balancer.





Note that in a multi-node setup, the end users would access the Web Application via the Load Balancer URL which would then route to an available node accordingly.

The Web Application could also be hosted on a single server and configured to send requests to the Load Balancer which would then forward them to an available node, however, this server would not have the high-availability in the same manner as the application nodes.

In terms of the functional aspect of the AppGyver Web Application, the CORTEX Interaction Portal will be sending requests to each CORTEX node's API Gateway via the Load Balancer, and performing actions based on the response. This includes the below core tasks, as well as any bespoke actions:

1. Authenticate User
2. Validate User Session (auth token)
3. Get / Set System Settings
4. Get Service Requests
5. Manage Service Requests and Groups (CRUD)
6. Get Dashboard Data - Process Counters
7. Get Process Execution Data
8. Get Process Task Input Data
9. Set Process Task Response Data

## 2 Setup and Deployment

Detailed steps for deploying the CORTEX Interaction Portal to a host server will be outlined in this section.

### 2.1 Pre-Requisites

| Component   | Requirements  |
|---|---|
| CORTEX Version                                    | This version of the CORTEX Interaction Portal requires CORTEX 2024.9 (or later).  |
| Application Nodes -> LDAP Server                  | Port 9389 must be permitted between any Application servers and the LDAP server which will be used for logging in to the Interaction Portal.  |
| Application Nodes (/ Load Balancer) – Ports       | Port 8722 must be permitted (inbound).<br>This is used to run the flows, authenticate via OAuth, and to interact with the data storage (Reliable Collections)   |
| Application Nodes – AD PowerShell Module          | Active Directory PowerShell Module must be installed.<br>This is included with the Remote Server Administration Tools feature in Windows Server.<br>Features > Remote Server Administration Tools > Role Administration Tools > AD DS and AD LDS Tools > Active Directory Module for Windows PowerShell |
| Web App Server – IIS Management PowerShell Module | IIS Management PowerShell Module must be installed to run the deployment script.<br>This is installed as part of the CORTEX Web App Server installation, so may already be present if installing the Interaction Portal on the same server as CORTEX Gateway.   |
| Web App Server                                    | Port 443 must be permitted (inbound).   |

### 2.2 Deployment Package Preparation

In order to install the CORTEX Interaction Portal, the relevant installation package must be acquired and made ready.

1. Download Cortex Innovation 2024.9 – Interaction Portal.zip
2. Extract the .zip to a suitable folder.

## 2.3 AppGyver Account Setup

The online AppGyver platform serves as a development environment for AppGyver apps and importing the pre-built CORTEX Interaction Portal app will allow us to make small changes to it so that it works as expected when deployed.

*🔗 This platform is also where further pages and functionality may be developed.*

To do this, complete the following steps:

1. Navigate to AppGyver online platform at <https://www.appgyver.com/community>.
2. Click the 'Join Our Community' or 'Login / Sign Up' button.
3. If an account is available, login to it before proceeding to Section 2.4.
4. If no account is available, click the 'Register' button and complete the steps to create one.

## 2.4 App Variables Setup

Once logged into an existing AppGyver platform account, complete the following steps to configure the application:

1. Select the 'Create New' button under 'My Apps'.
2. In the dialog which appears, select 'Import from file'.
3. Select Cortex.Interaction.Portal.zip.gpg, provided in the package downloaded in Section 2.2.
  - a. This is contained within the CORTEX Interaction Portal subfolder
  - b. For example, C:\Install\Cortex Innovation 2024.9 - Interaction Portal\Cortex Interaction Portal
4. Give the app a relevant name (such as CORTEX Interaction Portal) and click 'Create'.

Once the app has been imported, the Config Object must be set up correctly. The elements of this are global parameters that govern how the app behaves.

This single configuration is required, then when the app is built and deployed their values will be included in the deployment, providing all the necessary information to interact seamlessly with CORTEX flows.

To set up the App Variables, complete the following steps:

1. Open the app to any page.
2. Select the 'View / Variables' toggle in the top right of the screen to switch to the variables view.

VIEW  VARIABLES

3. App Variables should be shown by default, but page variables and parameters may be viewed using the panel on the left.
4. Select 'App Variables'

APP VARIABLES

PAGE VARIABLES

PAGE PARAMETERS

DATA VARIABLES

5. Selecting the 'ConfigVariables' object will allow the default value of each element to be configured in the panel on the right.

ConfigVariables  
App variable

Variable name ⓘ

ConfigVariables

Variable value type ⚙️

Object ▼

Add new property

Type property name +

Initial value

CortexUrl

https://<server>.<domain>.com

UamTenant

default

CortexPort

8722

CortexTenant

default

FlowAuthBase64

QmFzaWNBdXRoVXNlcjBREE5ODgz

6. The following elements should be configured:

a. Note that any elements not mentioned in the table below should be kept with their existing values.

| Variable Name     | Variable Usage  | Default Value Description   | Default Value Example         |
|-------------------|---|---|-------------------------------|
| CortexEnvironment | CORTEX Environment  | For most configurations this will be "default".   | default                       |
| CortexPort        | Port to communicate with the CORTEX API endpoint.   | For single node installations where REST requests are sent directly to a CORTEX Node's API Gateway, this is 8722.<br><br>For multi-node installations where REST requests are sent to a Load Balancer, this is 443.   | 8722                          |
| CortexTenant      | CORTEX Tenant   | For most configurations this will be "default".   | default                       |
| CortexUrl         | URL to communicate with the CORTEX API endpoint.  | Base CORTEX App URL.  | https://cortexApp1.domain.com |
| ApplicationName   | Name of the application, used to separate out any configuration and session related information.<br><br>This is also referenced in the deployment script later.           | This can be left blank, but if co-hosting multiple instances of CORTEX Interaction Portal, or 3 <sup>rd</sup> party applications which will use the User Access Management Module, it should be provided with a value | myApp1                        |
| FlowAuthBase64    | Base64 encoded username and password to communicate with the CORTEX API endpoint.<br><br><b>Only required if UseOAuth is false</b> – otherwise this should be left empty. | This is generated in the form of username:password<br><br>Note that there are tools online to encode text in Base64. You can also test in Postman using Basic Authentication,   | dXNlcm5hbWU6cGFzc3dvcmQ=      |

| Variable Name     | Variable Usage  | Default Value Description   | Default Value Example |
|-------------------|---|---|-----------------------|
|                   |   | and check the auto-generated headers.   |                       |
| MobileBreakpoint  | Used to determine the size of the screen before the app switches from a side bar to a hamburger menu. | For most configurations this will be 769. It is not suggested to modify this value.   | 769                   |
| TableBreakpoint   | Used to determine the size of the screen before the table layouts become smaller                      | For most configurations this will be 1224. It is not suggested to modify this value.  | 1224                  |
| UamEnvironment    | CORTEX Environment containing the published User Access Management flows.                             | For most configurations this will be "default"  | default               |
| UamPackageName    | Name of the package containing the published User Access Management flows.                            | When creating and publishing the package of flows, if the name is changed, this variable will need to be updated.   | UserAccessManagement  |
| UamPackageVersion | The version of the package containing the published User Access Management flows.                     | This should be left blank unless targeting a specific version of a published package (e.g. for testing).  |                       |
| UamTenant         | CORTEX Tenant containing the published User Access Management flows.                                  | For most configurations this will be "default".   | Default               |
| UseOAuth          | Flag indicating whether to use OAuth. If this is TRUE, the FlowAuthBase64 parameter can be left blank | If the Active Directory used for the end users is the same as the one used in CORTEX Gateway, it is suggested to use OAuth as it is a more secure manner of authentication to run the CORTEX flows. | True                  |

7. Save the app.



## 2.5 Theme Variables

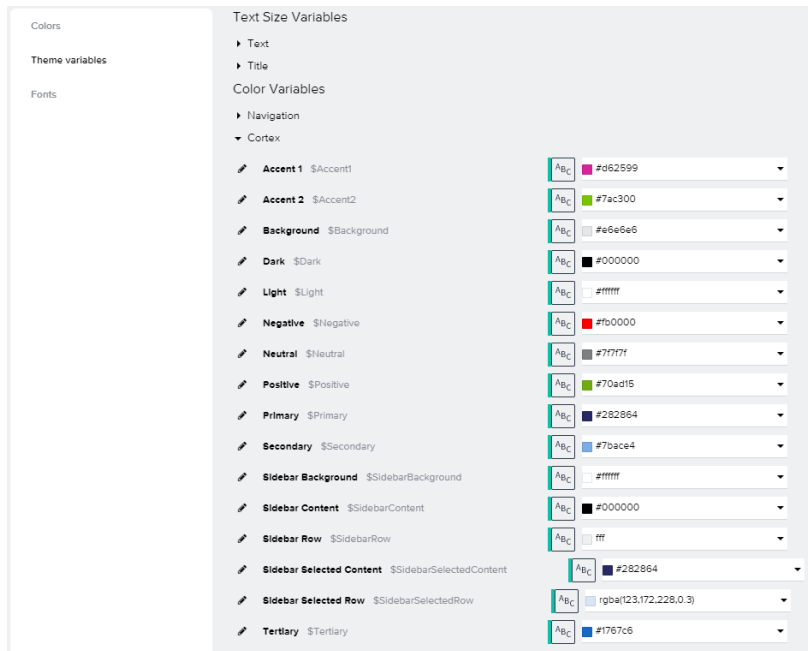
*This section is optional and will only be necessary if a custom theme for the Interaction Portal is required.*

Throughout the core pages of the CORTEX Interaction Portal, the styles used reference colours and text sizes that are stored as theme variables. For example, the 'Primary Font' variable is used below:



These variables control the look and feel of the Interaction Portal and are set to We Are CORTEX colours by default. To change the theme of the application, complete the following steps:

1. In the AppGyver online platform, with the application imported ready for editing, select 'Themes' from the bar at the top.
2. Select 'Theme Variables' from the panel to the left.
3. From the menu that is displayed, many different parameters that control the look and feel of the Interaction Portal can be configured.
4. The colour scheme of the core pages is stored under 'Color Variables' > 'Cortex'. Update these where necessary to quickly change the colour scheme of the Interaction Portal.



## 2.6 Import CORTEX Flows

The flows must be imported into the CORTEX Innovation environment(s).

1. Log into the CORTEX Gateway that was installed with the CORTEX application server.
2. In CORTEX Gateway, click 'Admin', then 'Studio Import'.
3. Select **User Access Management.Flows.studiopkg**, provided in the package downloaded in Section 2.2, inside the User Access Management directory, and import it.
  - a. If there are flow dependencies outside of the User Access Management group, include these unless they already exist on the system.
4. Select **Interaction.Portal.Core.Flows.studiopkg**, inside the CORTEX Interaction Portal directory, and import it.
5. Select **Process.Management.Extension.Flows.studiopkg**, inside the Process Management Extension directory, and import it.

*The import hierarchy should not need to change unless the flows need to be stored in a different location.*

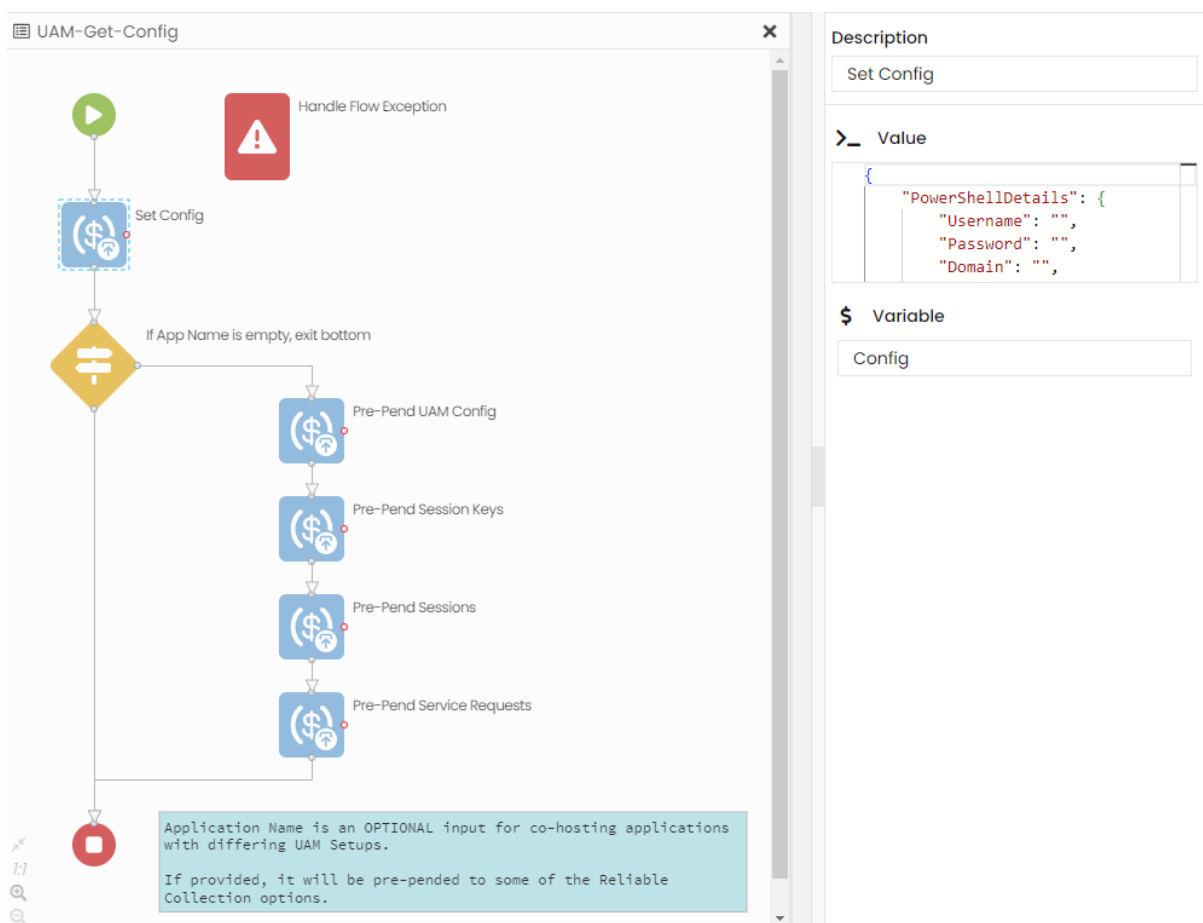
6. Set up the access to these flows using Studio Authorisation.

*Once the flows are imported, they should be available from the 'Dev' charms menu – note that you may need to refresh CORTEX Gateway after importing.*

## 2.7 Config Flow Configuration

### User Access Management Config Flow

1. Log into the CORTEX Gateway that was installed with the CORTEX application server.
2. In the Flows charms menu, navigate to Cortex Library > User Access Management > UAM-Get-Config
  - a. You may also search for the flow directly.
3. This flow contains a Set Variable block called 'Set Config' which must be edited accordingly:



4. Update the following parameters only, leaving any that are not listed as they are already configured.

| Parameter                  | Details  | Example              |
|----------------------------|--|----------------------|
| PowerShellDetails.Username | Username of an account to use PowerShell, e.g., Service Account. | "ctx_ServiceAccount" |

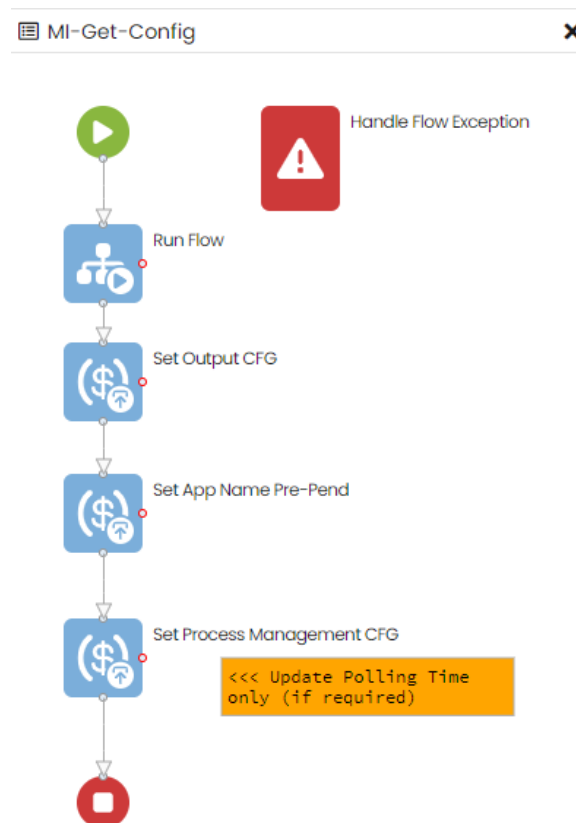
| Parameter                         | Details  | Example  |
|-----------------------------------|--|--|
|                                   | This user should be an administrator across the CORTEX servers.  |  |
| PowerShellDetails.Password        | Corresponding encrypted password   | "#_124211015226168!130105247000243225146179242013178~146135159100034!214216128191025238010012072111212#" |
| PowerShellDetails.Domain          | Domain of the account  | "myDomain"   |
| PowerShellDetails.Host            | Hostname of the machine  | "cortexAppl.myDomain.com"<br>"localhost"   |
| PowerShellDetails.Port            | PowerShell Port  | 5985   |
| PowerShellDetails.SSL             | Boolean indicating whether SSL should be used  | False  |
| RecursiveAccessControl            | Boolean indicating whether child user groups should inherit access control granted to parents.<br><br>e.g. User 1 is in 'Group A', Group A is a member of 'CIP Users', CIP Users has access to the system. If this is true, the CIP would allow User 1 access.<br><br>Default is False | True   |
| OptionalAdConfig.DomainController | The name of the domain controller server.<br><br>If empty, the domain that the node is attached to will be used.   | "server-name"  |

| Parameter                          | Details  | Example                            |
|------------------------------------|--|------------------------------------|
| OptionalAdConfig.BaseAdGroupSearch | The base path within the domain from which users can be selected.<br><br>If empty, the entire domain will be used. | "CN=Builtin,DC=CortexUsers,DC=com" |
| DataStorage Object                 | The object references for Shared Data Collections. These are suggested to leave as the default.                    | (Leave as default)                 |

5. The flow should then be saved and committed.

## Optional: Process Management Config Flow

1. Optional: Set Process Management Config – For most installations, the default values will not need to be changed.
  - a. In the Flows charms menu, navigate to Cortex Library > Manual Intervention > MI-Get-Config
    - i. You may also search for the flow directly.
  - b. Edit the final Set Variable block, as directed by the note on the workspace



- c. Edit the parameters inside this block if required.

- d. Save and Commit the flow if changes were made

## 2.8 Publish CORTEX Flows

The flows then need to be added to a package and published to the server.

1. In CORTEX Gateway, click 'Admin', then 'Packages'.
2. Select 'Add Package Definition' and enter a Package Name
  - a. The default name is 'UserAccessManagement', however any value can be entered.
  - b. If changing from default, the UamPackageName element of the ConfigVariables App Variable in AppGyver will need to be updated accordingly.
3. From the Flows panel below, select all the imported flows, along with any additional flows you might need with it.
  - a. Note that these flows have a dependency on the Reliable Collections and PowerShell flows inside the Generic Flow Library group.
  - b. These should be auto selected when creating the package for the first time; otherwise, ensure they also included.
4. The core flows will be the User Access Management, CORTEX Interaction Portal and Process Management groups.
5. Click Save, then when saved, click Publish.




## 2.9 Setup OAuth Published for Flows

If using OAuth (as set in the AppGyver Config Object variable), the configuration must be setup within the published Package.

1. In CORTEX Gateway, click 'Admin', then 'Packages'.
2. Select the package and version which you have just published.
3. Select the 'Authorisation' tab below the top grid.
  - a. If an error is shown while trying to load this, go to Settings > LDAP Connection and ensure this is configured / re-configure.
4. Select the relevant AD Groups which should have access to run these flows (i.e. the groups who should have access to the CORTEX Interaction Portal).
  - a. These should match the AD Groups who will be given Admin or User access to the portal).
5. Click Save.

 *Note that the CORTEX OAuth endpoint uses the LDAP Server configured during installation.*

## 2.10 Initialise Shared Data (Reliable Collections)

-  *Note that in a High Availability cluster, this only needs to be done once on one of the servers within the cluster.*
-  *This will initialise data for all components of the CORTEX Interaction Portal. If the Process Management feature is not being installed, the collection will be initialised but can be ignored.*
-  *Note that this needs to be run on an application node, i.e. a server hosting Service Fabric and CORTEX, in an HA scenario.*

The steps to initialise the shared data collections and configuration depend on the installation requirements:

- New Development System (with examples)
  - In this scenario, the demo files should be imported to create the example Service Requests and Processes
  - See '**Import Data**' section, referencing the provided example JSON files.
- New Blank System (no example data)
  - In this scenario, the demo files should not be imported. This means that no Service Requests or Process Configuration will be created.
  - See '**Initialise Data**' section.
- Migrated System (e.g. new environment, Dev > Test, Test > Prod)
  - In this scenario, data needs to be exported from an existing system first.
  - This data will then be imported to the new system.
  - See '**Export Data from Existing System**' and '**Import Data**' sections.

Based on the above, follow the relevant sections below.

### Export Data from Existing System

To run the script, complete the following steps on an existing CORTEX Application node.

1. Copy the package obtained in Section 2.2 to a server where the CIP is to be installed.
2. If it has not been unzipped, unzip it to a suitable location.
3. Open Windows PowerShell ISE **as an administrator**
4. In the PowerShell window, run the following to navigate to the 'Cortex Interaction Portal' directory where the file was unzipped:

```
cd "<file path>\Cortex Interaction Portal"
```



For example:

```
cd "C:\Install\Cortex Innovation 2024.9 - Interaction Portal\Cortex Interaction Portal"
```

5. In the script window, paste the following:



```
.\Export.Shared.Data.RC.ps1`
-URL "https://server.domain.com"`
-Port "8722"`
-Username "BasicAuthUser"`
-Password "<password>"`
-Tenant "default"`
-Environment "default"`
-ExportDirectory "C:\Temp"`
-ApplicationName "<AppName>"
```

6. Update any parameters as necessary.

| Parameter Name | Description   | Example                   |
|----------------|---|---------------------------|
| URL            | The URL for CORTEX / Reliable Collections. For HA solutions, this should be one of the servers in the cluster | https://server.domain.com |
| Port           | The port for the Service Fabric Gateway endpoint. Usually installed on port 8722.                             | 8722                      |
| Username       | Basic Authentication User for requests to CORTEX  | BasicAuthUser             |
| Password       | Basic Authentication Password for requests to CORTEX  |                           |
| Tenant         | The tenant for the shared data, currently only default is supported   | default                   |
| Environment    | The environment for the shared data, currently only default is supported                                      | default                   |

| Parameter Name  | Description  | Example |
|-----------------|--|---------|
| ExportDirectory | The folder used to generate the JSON exports.  | C:\Temp |
| ApplicationName | The optional Application Name used when deploying the application, which should match the Config Variables in AppGyver for the instance you are exporting. |         |

## Import Data

-  This will import JSON data, either from a previous Export operation, or the demo data included with the CORTEX Interaction Portal package.
-  The demo data is contained within the [Cortex Interaction Portal | Demo Config directory](#)

To run the script, complete the following steps.

- Copy the package obtained in Section 2.2 to a server where the CIP is to be installed.
- If it has not been unzipped, unzip it to a suitable location.
- Open Windows PowerShell ISE **as an administrator**
- In the PowerShell window, run the following to navigate to the directory where the file was unzipped:

```
cd "<file path>\Cortex Interaction Portal"
```

For example:

```
cd "C:\Install\Cortex Innovation 2024.9 - Interaction Portal\Cortex Interaction Portal"
```

- In the script window, paste the following:
  - Note that if importing the Demo config, -SettingsImportPath should be replaced with both -adminAdGroups and -userAdGroups as shown in the Initialise Data example.

```
.\Initialise.Shared.Data.RC.ps1 `
-URL "https://server.domain.com" `
-Port "8722" `
-Username "BasicAuthUser" `
-Password "<password>" `
-Tenant "default" `
```

```
-Environment "default" `
-ServiceRequestsImportPath "C:\Temp\ServiceRequests.json"
-ProcessDefinitionsImportPath "C:\Temp\RbacConfig.json"
-ProcessKeysImportPath "C:\Temp\RbacKeys.json"
-SettingsImportPath "C:\Temp\AppSettings.json"
-ApplicationName "<AppName>"
```


6. Update any parameters as necessary.

| Parameter Name               | Description   | Example                      |
|------------------------------|---|------------------------------|
| URL                          | The URL for CORTEX / Reliable Collections. For HA solutions, this should be one of the servers in the cluster | https://server.domain.com    |
| Port                         | The port for the Service Fabric Gateway endpoint. Usually installed on port 8722.                             | 8722                         |
| Username                     | Basic Authentication User for requests to CORTEX  | BasicAuthUser                |
| Password                     | Basic Authentication Password for requests to CORTEX  |                              |
| Tenant                       | The tenant for the shared data, currently only default is supported   | default                      |
| Environment                  | The environment for the shared data, currently only default is supported                                      | default                      |
| ServiceRequestsImportPath    | Path to the exported JSON for Service Requests (or Demo data JSON)  | C:\Temp\ServiceRequests.json |
| ProcessDefinitionsImportPath | Path to the exported JSON for Process   | C:\Temp\RbacConfig.json      |

| Parameter Name        | Description  | Example               |
|-----------------------|--|-----------------------|
|                       | Definitions (or Demo data JSON)  |                       |
| ProcessKeysImportPath | Path to the exported JSON for Process Definition Keys (GUIDs) (or Demo data JSON)  | C:\Temp\RbacKeys.json |
| SettingsImportPath    | Path to the exported JSON for App Settings.<br><br>Note that this can be replaced with the <b>adminAdGroups</b> and <b>userAdGroups</b> parameters from the below 'Initialise Data' script if import of settings is not required | C:\Temp\Settings.json |
| ApplicationName       | The optional Application Name you would like to provide, which should match the Config Variables in AppGyver for the new instance you are creating.  |                       |

7. Run the script and ensure there are no errors

## Initialise Data

 *These steps should be followed to create a fresh system without the demo data or imported data from another system.*

To run the script, complete the following steps.

1. Copy the package obtained in Section 2.2 to a server where the CIP is to be installed.
2. If it has not been unzipped, unzip it to a suitable location.
3. Open Windows PowerShell ISE **as an administrator**
4. In the PowerShell window, run the following to navigate to the directory where the file was unzipped:

```
cd "<file path>\Cortex Interaction Portal"
```

For example:

```
cd "C:\Install\Cortex Innovation 2024.9 - Interaction Portal\Cortex Interaction Portal"
```

5. In the script window, paste the following:

```
.\Initialise.Shared.Data.RC.ps1 `
-URL "https://server.domain.com" `
-Port "8722" `
-Username "BasicAuthUser" `
-Password "<password>" `
-Tenant "default" `
-Environment "default" `
-adminAdGroups @("Domain Admins") `
-userAdGroups @("Domain Users") `
-ApplicationName "<AppName>"
```

6. Update any parameters as necessary.

| Parameter Name | Description   | Example                            |
|----------------|---|------------------------------------|
| URL            | The URL for CORTEX / Reliable Collections. For HA solutions, this should be one of the servers in the cluster | https://server.domain.com          |
| Port           | The port for the Service Fabric Gateway endpoint. Usually installed on port 8722.                             | 8722                               |
| Username       | Basic Authentication User for requests to CORTEX  | BasicAuthUser                      |
| Password       | Basic Authentication Password for requests to CORTEX  |                                    |
| Tenant         | The tenant for the shared data, currently only default is supported   | default                            |
| Environment    | The environment for the shared data, currently only default is supported                                      | default                            |
| adminAdGroups  | A comma separated list of the user groups in Active   | @("Domain Admins", "Local Admins") |

| Parameter Name  | Description   | Example                          |
|-----------------|---|----------------------------------|
|                 | Directory to give admin roles   |                                  |
| userAdGroups    | A comma separated list of the user groups in Active Directory to give user roles  | @("Domain Users", "Local Users") |
| ApplicationName | The optional Application Name you would like to provide, which should match the Config Variables in AppGyver for the new instance you are creating. |                                  |

7. Run the script and ensure there are no errors

## 2.11 Website Deployment

The deployment script included in the package obtained in Section 2.2 will perform the following required actions:

1. Set up the website in IIS ready for the application to be deployed to it.

To run the script, complete the following steps.

1. Copy the package obtained in Section 2.2 to a server where the CIP is to be installed.
2. If it has not been unzipped, unzip it to a suitable location.
3. Open Windows PowerShell ISE **as an administrator**
4. In the PowerShell window, run the following to navigate to the directory where the file was unzipped:

```
cd "<file path>\Cortex Interaction Portal"
```

For example:

```
cd "C:\Install\Cortex Innovation 2024.9 - Interaction Portal\Cortex Interaction Portal"
```

5. In the script window, paste the following:

```
.\Deploy.Cortex.Interaction.Portal.ps1 `
-PortalIISSiteName "Cortex" `
-PortalIISSApplicationName "CortexInteractionPortal" `
-PortalIISSAppPool "CortexInteractionPortal" `
```

```
-IISDirectory "C:\inetpub\wwwroot" `
-iisSiteHostName ""
```

- Update any parameters as necessary.

| Parameter Name           | Description  | Example                   |
|--------------------------|--|---------------------------|
| PortalIISSiteName        | The name of the web site to which the application will be deployed   | "Cortex"                  |
| PortalIISApplicationName | The name of the application when it is deployed.<br><br><i>Note that this has no relation to the ApplicationName used in AppGyver and the previous scripts</i> | "CortexInteractionPortal" |
| PortalIISAppPool         | The name of the application pool under which the application will run  | "CortexInteractionPortal" |
| IISDirectory             | The IIS root folder  | "C:\inetpub\wwwroot"      |
| iisSiteHostName          | The hostname to bind to the website, not always required within IIS  | ""                        |

- The PowerShell ISE window should look as follows:

```
1 .\Deploy.Cortex.Interaction.Portal.ps1 `
2 -PortalIISSiteName "Cortex" `
3 -PortalIISApplicationName "CortexInteractionPortal" `
4 -PortalIISAppPool "CortexInteractionPortal" `
5 -IISDirectory "C:\inetpub\wwwroot" `
6 -iisSiteHostName ""
```

- Run the script using the play button above the script window.
- Once complete, repeat this on any other servers (if hosting the application on multiple servers).

## 2.12 Web Application Deployment

- Login to AppGyver and go to the App.
- Select a page (for example Home Page) and select 'Launch' from the toolbar at the top.

3. From the 'Launch' options, select 'Open build service' under the 'Build your app' section.
4. If you previously created a web configuration, click on it, otherwise press the 'Create Configuration' button, then configure the build:
  - a. For 'Choose a target platform', select 'Web'
  - b. Provide a name for the configuration, such as 'Web configuration'
  - c. Press 'Create'
  - d. Click on the created web configuration.
5. Enter the build details:
  - a. Enter a version number, for example 0.0.1
  - b. Select the Runtime Version to 'Latest'.
  - c. Click 'Build'

Once the operation has completed after some time, an email will be sent to the email address for this AppGyver account. This will contain a link from which the .zip file may be downloaded. This can then be extracted and placed into an IIS Site on the App Server(s) that was created by the script executed in Section 2.11.

1. Unzip the file that was downloaded.
2. Navigate to the folder where the web application was installed in File Explorer
  - a. This will be of the form:

`<IISDirectory>\<PortalIISSiteName>\<PortalIISApplicationName>`

e.g. C:\inetpub\wwwroot\Cortex\CortexInteractionPortal

3. Copy all the unzipped files across
  - a. If the copy operation fails, you may need to first copy them to a directory local to your user (such as a folder in Downloads)
4. If required, repeat this file copy operation on any other servers hosting the application

Once this has been completed, the Web Application is accessible from your server directly, for example: <https://www.myserver.myDomain.com/CortexInteractionPortal>.

## 2.13 Web App Redirect Rule

If CORTEX Gateway is installed on the same server (and within the same website in IIS) as the CORTEX Interaction Portal, there are some redirect rules which will prevent a user from accessing the Interaction Portal.

1. Navigate to the IIS Directory, for example, C:\inetpub\wwwroot\Cortex
  - a. You should see subdirectories for both the Interaction Portal and Gateway here
2. Open the web.config file in this directory.



- a. If this file is not present within the top level (e.g. 'Cortex') IIS directory, this change will not be necessary and you can proceed to the next section.
3. If there is a 'Redirect Cortex to gateway' rule present, you will need to add the below line to it, replacing the '**CortexInteractionPortal**' with the application name if it is different:

```
<add input="{REQUEST_URI}" pattern=".*\CortexInteractionPortal.*" negate="true" />
```

4. Save and exit the file.

## 2.14 CORS Configuration

CORS, or Cross-Origin Resources Sharing, is a security protocol in websites which prevents requests from an unknown origin. To ensure the Web application can call CORTEX correctly, it must be listed as an allowed origin.

An origin is identified based on the scheme (e.g., HTTPS), domain, and port. The CORTEX API Gateway should share the same scheme and domain, but with a different port – this means that by default, the requests will not be permitted.

1. Navigate to  
C:\ProgramData\SF\<Customer>.<Node>\Fabric\work\ImageCache\Store\Cortex.Innovation.Core\ApiGatewayPkg.Code.<Version>

*<Customer> and <Node> will depend on how the node was configured in Service Fabric during the installation of CORTEX.*

2. Open the appsettings.json file.
3. Under the Cors section, edit it as follows:

| Parameter               | Details   | Example   |
|-------------------------|---|---|
| Enabled                 | Sets CORS as enabled within the rules specified below.  | true  |
| AllowedOrigins          | Array of Strings containing all the allowed origins.<br><br>Note that the origin of the app server(s) hosting the Web App is required, and the origin of the AppGyver Preview App site is optional. | [<br>"https://*.<domain>.com",<br>"https://*.appgyver.com"<br>] |
| AllowCredentials        | Whether or not credentials are permitted in the request   | true  |
| AllowWildcardSubdomains | Whether or not wildcards (*) are permitted for subdomains. Based on the   | true  |

| Parameter | Details  | Example |
|-----------|--|---------|
|           | example in AllowedOrigins, this should be true |         |

Once this is done, navigate to the Service Fabric Explorer page and restart any Service Fabric nodes for the changes to take effect.

1. Navigate to the Service Fabric Explorer page.
  - a. For example, <https://cortexApp1.myDomain.com:9080/Explorer/>
2. Expand Nodes in the left-hand panel, and in turn, select the node and restart it. If you need to continue operations on the server, they should be restarted 1 at a time.
  - a. Click the node.
  - b. In the right-hand panel select the 'Actions' dropdown and click Restart
  - c. This can take 5-10 minutes per-node to complete.

## 2.15 Housekeeping Tasks

Included as part of the flow packages are some housekeeping flows which need to be scheduled.

Before scheduling, you should go to each of the flows mentioned in the below table and update the variable 'ListOfApplications'. This needs to be a list which contains each 'ApplicationName' (from the AppGyver Config Variable and the Initialise / Import Data script) which has been deployed. Any applications deployed in future should then be added to this in the same manner.

For example, if you deployed an application without a dedicated App Name:

```
new List<dynamic>()
{
    "",
}
```

Or if you co-host multiple applications:

```
new List<dynamic>()
{
    "Config Management",
    "Reporting",
    "Portal",
}
```

Save and Commit these flows once done.

To setup the schedule in CORTEX, select the published package version from Settings > 'Packages', select the Schedules tab at the bottom and click the Add button.

You must give the schedule a name, select the flows detailed in the table below, and enter a schedule in CRON syntax. Each flow will be setup as a separate schedule.

As an example, the CRON syntax to start a flow at 8am every day is 0 0 8 ? \* \*.

| Flow Name           | Description   | Schedule |
|---------------------|---|----------|
| UAM-Session-Cleanup | A flow to delete Session json files that are older than 24 hours. | Daily    |
| MI-Process-Cleanup  | A flow to delete Process files that are older than 1 week.        | Daily    |

## 3 Testing

### 3.1 Testing User Access Management Flows

Once the configuration is complete, the flows can be tested from CORTEX Gateway and from Postman.

**Note that the full OAuth login process cannot be tested from Gateway, as it uses the OAuth Token in the request to provide the user information.**

**This means that for OAuth requests, Username and Password would not normally be provided to the Authenticate flow,**

1. In CORTEX Gateway, open the User Access Management > Authentication > UAM-Authenticate-User flow.
2. In the right-hand panel, enter a username and password in the input variables.
  - a. Username should be in the format "domain\\username"
3. Set a breakpoint on the final 'End Flow' block and run the flow.
  - a. If any exceptions occur, these should be diagnosed and resolved.
  - b. If it goes through successfully, check the output variables (accessLevel and authToken)

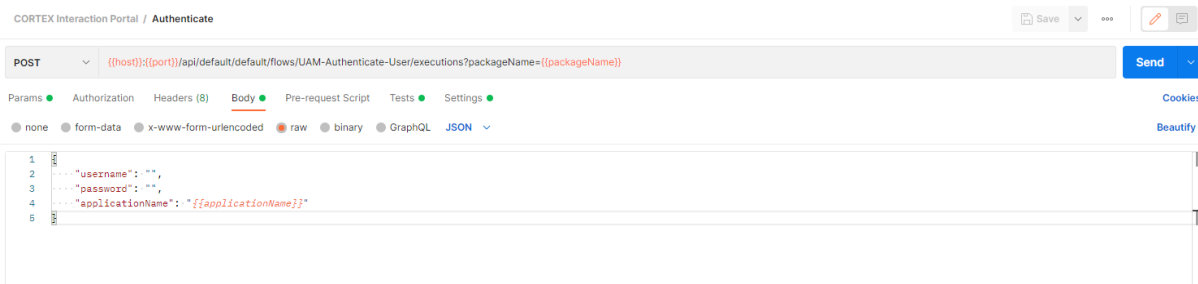
Once this flow has been validated, the published flows can be tested using the postman collection **Cortex.Interaction.Portals.postman\_collection.json**, provided as part of the package obtained in Section 2.2.

1. Import the collection to Postman, which if required can be downloaded and installed from <https://www.postman.com/>.
2. Update the Collection variables.

| CORTEX Interaction Portal  |                    |                           |                           | Watch | 0 |
|--|--------------------|---------------------------|---------------------------|-------|---|
| Authorization  | Pre-request Script | Tests                     | Variables                 |       |   |
| These variables are specific to this collection and its requests. <a href="#">Learn more about collection variables.</a> |                    |                           |                           |       |   |
|  | VARIABLE           | INITIAL VALUE             | CURRENT VALUE             |       |   |
| <input checked="" type="checkbox"/>  | host               | https://server.domain.com | https://server.domain.com |       |   |
| <input checked="" type="checkbox"/>  | port               | 8722                      | 8722                      |       |   |
| <input checked="" type="checkbox"/>  | packageName        | UserAccessManagement      | UserAccessManagement      |       |   |
| <input checked="" type="checkbox"/>  | applicationName    | cip2                      | cip2                      |       |   |
| <input checked="" type="checkbox"/>  | username           |                           |                           |       |   |
| <input checked="" type="checkbox"/>  | password           |                           |                           |       |   |
| <input checked="" type="checkbox"/>  | authToken          |                           |                           |       |   |
| <input type="checkbox"/>   | packageVersion     |                           |                           |       |   |
|  | Add a new variable |                           |                           |       |   |

| Variable        | Description  | Example Value                   |
|-----------------|--|---------------------------------|
| host            | The host for communication with the relevant CORTEX API Gateway  | https://cortexApp1.myDomain.com |
| port            | The port for communication with the relevant CORTEX API Gateway  | 8722                            |
| packageName     | The package containing the user access management flows.   | UserAccessManagement            |
| applicationName | Name of the Application to run the requests against (can be empty if not configured as a specific application).                  |                                 |
| username        | Username to authenticate via OAuth   | <Username>                      |
| password        | Password to authenticate via OAuth   | <Password>                      |
| authToken       | Authentication token returned from the 'Authenticate' request. This will be automatically saved from the 'Authenticate' requests |                                 |

3. Select save to persist the variable values.
4. Click on the Authorization tab, scroll down and select 'Get New Access Token'
  - a. If successful, there should be an option to use this token which will enable OAuth for all the requests in the collection.
5. Test the 'Authenticate' request. This is what is called when entering a username and password to access the Interaction Portal.
  - a. Select the request and run it. For OAuth, the credentials inside the body are not required.



6. Click 'Send'
  - a. Examine the response. If successful it will take the following structure:

```
{
  "accessLevel": "<e.g. Admin>",
  "userProperties": {
    "name": "<username>",
    "accountName": "<username>",
    "upn": "<username>@myDomain.com",
    "roles": [
      "<user groups to which the user belongs>"
    ]
  },
  "status": {
    "Status": "OK"
  },
  "authToken": "84f01914-6j74-4055-gf38-e6de9ns9ba41"
}
```

7. The authToken returned at the bottom of the response may be used for further requests. This will automatically be saved in the collection for the requests to use.

## 3.2 Testing the Web Application

- 📖 *If the CORTEX Application server does not have inbound internet connectivity, you will not be able to test from the AppGyver preview mode and this step can be skipped.*
- 📖 *If you have not configured CORS to enable the AppGyver Preview website, this will not work.*

To test the app, complete the following steps:

1. Login to AppGyver and go to the app.
2. Select a page (for example Home Page) and select Launch from the toolbar at the top.
3. In the left-hand panel, select Preview, then select Open App Preview Portal
  - a. Once the page loads, you should open Developer Tools to diagnose any potential issues with the requests to CORTEX.
4. You can then attempt to login to the Web Application using domain credentials and validate the request from the Network tab in Developer Tools.
5. Along with the below tests, any configuration aspects within the Admin Settings pages can now be completed (such as configuring access levels for Service Requests and Processes, if not already done).


## 3.3 Testing UI-Driven Process (Service Requests)

- 📖 *Note that the example Service Request is a very simple example used to validate that the Service Requests are working correctly and show the general flow of data within the UI.*

1. Navigate to the UI hosted on the server and open Developer Tools to diagnose any potential issues.
  - a. For example:  
<https://www.cortexApp1.myDomain.com/CortexInteractionPortal>
2. Login using domain credentials with the relevant permissions.
3. Click Service Requests – by default there should be 1 Service Request configured if you have imported the demo data.
4. Select 'Add Numbers' in the left-hand panel, and then 'Start' in the right-hand panel.
5. In the page that opens, enter 2 numbers, and click the button.
6. Ensure that a response is received from CORTEX and shown in the page.

- 📖 *For a more complex example and more rigorous test, start the example 'Server Provisioning' Service Request and follow it through to completion.*

## 3.4 Testing Process-Driven UI (Process Dashboard)

 *For this process, the role-based access control will need to be set up – you can check rbac.json if required*

 *Note that it is possible to assign different levels of permission, for example user access for the 1<sup>st</sup> and 2<sup>nd</sup> UI and manager access for the 3<sup>rd</sup> (approval) UI.*

1. From CORTEX Gateway or Postman, start 'Example-Onboarding-Flow'.
  - a. In CORTEX Gateway, the flow can be found under User Access Management/Example Flows.
  - b. In Postman, the execution will be synchronous. Therefore, no response will be received until the flow completes its execution, which will require manual interventions using the Process Dashboard.
2. Ensure there are no exceptions after it starts running
3. Navigate to the UI hosted on the server and open Developer Tools to diagnose any potential issues.
4. Login using domain credentials with the relevant permissions.
5. Click 'Process Dashboard'
  - a. Data should be received from CORTEX and used to render the table view.
6. Select either the 'Onboarding Example' Process filter or the 'Pending' Status filter
  - a. A list of relevant tasks should be displayed, most likely with 1 item.
  - b. Providing the RBAC configuration is set up correctly, you should also be able to see this execution on the top-level Dashboard page.
7. Click the document button to view a list of logs for this process execution.

### Actions



- a. Close the modal once it appears.
8. Click the 'Open Task' (eye) button in the Action column.

### Actions



- a. You should be taken to a UI where you can select the persons department – complete this.
9. Refresh the task list – after a short time it should be showing a new task again.
10. Click Open Task again – you should be prompted to assign the person some assets.
  - a. Complete this UI by selecting some assets and clicking the button.



11. Refresh again – it should now be pending manager approval.
  - a. Complete this, either approving or denying the request

## 4 Appendix A: Renaming Web App Title

- 📖 Note that as of the latest version of AppGyver, the Web App Title is set based on the name given when you import the application.
- 📖 Therefore, these steps should not be required unless you require a name change after deployment.

By default, the Web Application title is set to the name given to the application when you import into the AppGyver IDE. This can be updated by modifying content in some of the post-build files. The below script can automate this action by providing the directory of the unzipped files and the required website name.

- 📖 This can be done either before or after copying the files to the IIS directory on the server.
- 📖 For a multi-node environment, it is suggested to perform these steps before the copy to save running it multiple times.

```
$websiteName = "Cortex Interaction Portal"
$websiteDirectory = "<website directory>"

# Replace <Title> in HTML Files
$htmlPages = Get-ChildItem -Path "$websiteDirectory\*.html" -Recurse | ForEach-Object {
    # Read the file and use replace()
    (Get-Content $_).Replace("<title>AppGyver</title>","<title>$websiteName</title>") | Set-Content $_
}

# Replace title JSON in JavaScript File
$javascriptFile = Get-ChildItem -Path
"$websiteDirectory\_next\static\chunks\pages\*app*.js" | ForEach-Object {
    # Read the file and use replace()
    (Get-Content $_).Replace('""title""',null,"AppGyver"),('""title""',null,"$websiteName") | Set-Content $_
}
```

## 5 Appendix B: Issues with Self Signed Certificates

Even if CORS is configured correctly as in Section 2.14, an 'Is CORS enabled' error message may be encountered still while attempting to call a CORTEX flow if a Self-Signed Certificate without valid Subject Alternate Names is used.

To resolve this, complete the following steps:

1. Go to `https://<server>.<domain>.com:8722`
2. An error dialog should be displayed, click 'Advanced' then 'Continue Anyway'.
3. Wait for page to navigate to a 'failed to load' page.
4. This ensures that your browser will ignore the self-signed certificate in the API Gateway endpoint, so the browser can run flows.

## 6 Appendix C: Load Balancer Configurations

In an HA Cluster, there are many different options for configuration and architecture. 2 of these are detailed below.

### 6.1 Single-Portal Setup

The 'Single-Portal' setup refers to running 1 Web Application containing the CORTEX Interaction Portal. This could be alongside the Load Balancer, or on a separate server.

Either way, this means that the server containing the Interaction Portal will be issuing requests to the Load Balancer (whether local or remote), most likely on port 8722, which will then forward these requests on to the relevant node.

These end nodes will run the CORTEX flows, as well as store the Shared Data (Reliable Collections) used by the application – although the data is only every retrieved via flows.

### GoBetween Configuration

In the JSON configuration for the GoBetween Load Balancer, there should only be one entry in the 'servers' attribute (although this will reference multiple servers in the servers > discovery > static\_list array). This configuration will route requests from a certain inbound port on the Load Balancer server, to the relevant Service Fabric ports on the end servers.

The easiest configuration for this is to have the inbound port set to the same as the individual nodes, most likely using port 8722.

The below example can be used for reference when setting up the configuration.

Example Configuration:

```
{
  "logging": {
    "level": "debug",
    "output": "stdout"
  },
  "api": {
    "enabled": false,
    "bind": ":8888"
  },
  "servers": {
    "CortexNodes": {
```

```

"max_connections": 0,
"client_idle_timeout": "5m",
"backend_idle_timeout": "5m",
"backend_connection_timeout": "10s",
"protocol": "tls",
"balance": "roundrobin",
"bind": "0.0.0.0:8722",
"backends_tls": {
    "cert_path": ".\\client.crt",
    "key_path": ".\\client.key",
    "min_version": "tls1.2",
    "ciphers": [
        "TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384",
        "TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256"
    ],
    "prefer_server_ciphers": false,
    "session_tickets": false,
    "ignore_verify": false
},
"tls": {
    "cert_path": ".\\client.crt",
    "key_path": ".\\client.key",
    "min_version": "tls1.2",
    "ciphers": [
        "TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384",
        "TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256"
    ],
    "prefer_server_ciphers": false,
    "session_tickets": false
},
"discovery": {
    "kind": "static",
    "failpolicy": "keeplast",
    "interval": "0",
    "timeout": "0",
    "static_list": [
        "node1.domain.com:8722",
        "node2.domain.com:8722",
    ]
}

```

```
        "node3.domain.com:8722"
      ]
    },
    "healthcheck": {
      "kind": "exec",
      "interval": "3s",
      "timeout": "2s",
      "exec_command": ".\\ApiGatewayServiceHealthcheck.bat",
      "exec_expected_positive_output": "1",
      "exec_expected_negative_output": "0"
    }
  }
}
```

## 6.2 Multi-Portal Setup

The 'Multi-Portal' setup refers to running a Web Application containing the CORTEX Interaction Portal on each of the individual nodes, which can then be accessed from the Load Balancer and routed accordingly.

This means that in addition to the routing traffic over the Service Fabric port (e.g. 8722), the Load Balancer will also forward requests to access the portal (e.g. using port 443).

The Portal itself will then send requests back to the load balancer, to be forwarded on to an individual node.

Due to the shared data within Reliable Collections, it will not matter whether the server used to service the web application is different to the server used to run the flows as a request is made.

## GoBetween Configuration

In the JSON configuration for the GoBetween Load Balancer, there should only be two entries in the 'servers' attribute (although each will reference multiple servers in the servers > discovery > static\_list arrays).

The first configuration route requests from a certain inbound port on the Load Balancer server, to the relevant Service Fabric ports on the end servers. The easiest configuration for

this is to have the inbound port set to the same as the individual nodes, most likely using port 8722.

The second configuration will route requests to the CORTEX Interaction Portal, for example using port 443.

The below example can be used for reference when setting up the configuration.

Example Configuration:

```
{
  "logging": {
    "level": "debug",
    "output": "stdout"
  },
  "api": {
    "enabled": false,
    "bind": ":8888"
  },
  "servers": {
    "CortexNodes": {
      "max_connections": 0,
      "client_idle_timeout": "5m",
      "backend_idle_timeout": "5m",
      "backend_connection_timeout": "10s",
      "protocol": "tls",
      "balance": "roundrobin",
      "bind": "0.0.0.0:8722",
      "backends_tls": {
        "cert_path": ".\\client.crt",
        "key_path": ".\\client.key",
        "min_version": "tls1.2",
        "ciphers": [
          "TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384",
          "TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256"
        ],
        "prefer_server_ciphers": false,
        "session_tickets": false,
        "ignore_verify": false
      }
    }
  }
}
```

```

    },
    "tls": {
      "cert_path": ".\\client.crt",
      "key_path": ".\\client.key",
      "min_version": "tls1.2",
      "ciphers": [
        "TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384",
        "TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256"
      ],
      "prefer_server_ciphers": false,
      "session_tickets": false
    },
    "discovery": {
      "kind": "static",
      "failpolicy": "keeplast",
      "interval": "0",
      "timeout": "0",
      "static_list": [
        "node1.domain.com:8722",
        "node2.domain.com:8722",
        "node3.domain.com:8722"
      ]
    },
    "healthcheck": {
      "kind": "exec",
      "interval": "3s",
      "timeout": "2s",
      "exec_command": ".\\ApiGatewayServiceHealthcheck.bat",
      "exec_expected_positive_output": "1",
      "exec_expected_negative_output": "0"
    }
  },
  "CortexInteractionPortal": {
    "max_connections": 0,
    "client_idle_timeout": "5m",
    "backend_idle_timeout": "5m",
    "backend_connection_timeout": "10s",
    "protocol": "tcp",

```



```

    "balance": "roundrobin",
    "bind": "loadBalancer.domain.com:443",
    "discovery": {
      "kind": "static",
      "static_list": [
        "node1.domain.com:443",
        "node2.domain.com:443",
        "node3.domain.com:443"
      ]
    },
    "healthcheck": {
      "fails": 1,
      "passes": 1,
      "interval": "10s",
      "timeout": "9s",
      "kind": "ping",
      "ping_timeout_duration": "500ms"
    }
  }
}

```