Overview

Table ***dbo.t_configuration*** contains most of the environmental settings. On change made to this table, most of the services/APIs require rows in ***dbo.t_caches*** table to have their ***publishid*** value updated to new random GUID for hot reload. Some services/APIs require restart upon change of specific keys.

Following are the groups of values based on the Full example script at the end of this page.

1. Main URLs

These values hold the URLs used by services for internal communication inside the closed network. Unless containers domain names or routing are changed, these values will stay the same.

There is one exception in form of ***Client_AuthBaseUri*** which contains EXTERNAL URL leading to the **Authorization controller** of the **cb_auth** container.

For ease of use, shared parts of root URLs are declared as variables at start of the Full example script.

2. No REDIS / With REDIS

These two groups are two side of the same coin. Depending on whether Realtime with REDIS are used or not, one of these groups is left out.

Server_RedisCacheConfig

Connection string consists of several parts separated by comma:

- server
  - positional required parameter, [server] format
  - in case of REDIS instance running on different than default port, it needs to be specified after colon in "*server:port*" format
- ssl
  - optional parameter, ssl=[True/False] format
  - indicates whether encrypted communication should be used
- password
  - optional parameter, password=[value] format
- abortConnect
  - optional parameter, abortConnect=[True/False] format
  - specifies whether connection should be aborted on failure (prevents reconnection)

Examples:

- `redis,ssl=False,abortConnect=False`
- `redis-name.redis.cache.windows.net:6380,password=*****,ssl=True,abortConnect=False`

Server_RedisCacheServerName

Name of connection. By default REDIS server URI without port can be specified.

Examples:

- `redis`
- `redis-name.redis.cache.windows.net`

3. Security

Server_WebApiOTPSecret and Server_WebApiTokenSecret

These two keys hold random strings that are used for securing the OTP token and access/refresh token generation/encryption. The Server_WebApiOTPSecret must have a length of be 32 characters, and the Server_WebApiTokenSecret must have a length of 64 characters. In the Full example script random string values have been used.

Server_CryptextKeys

The value of this setting is a JSON object, serialized as a string, that holds the key and initialization vector (IV) properties for the symmetric encryption algorithm. The key must be 32 bytes and the IV must be 16 bytes.

```
{
  "Key": "[[SYMMETRIC ENCRYPTION KEY, AS A BASE64 STRING]]",
  "IV": "[[INITIALIZATION VECTOR, AS A BASE64 STRING]]"
}
```

Server_WebApiExternalLoginProviders

Value of this configuration key is a JSON array of objects representing sets of configurations. Most of the time there will be only one set of providers defined.

The **AppCode** value of the set is it's ID and based on value specified in **appName** of env.Override.js

Following is the template for all 5 currently supported providers. Template variables (text enclosed in [[ ]]) need to be replaced with proper values (including the [[ ]]).

- [[CLIENT_ID]] -- Client ID of the app registration.
- [[CLIENT_SECRET]] -- Client Secret generated for the app registration.
- [[WEB_APP_URL]] -- Base URL under which frontend application is accessible. (i.e. web.collaboard.app)
- [[ADFS_URL]] -- ADFS server URL.
- [[SAML_AUTH_URL]] -- SAML Identity Provider authenticate URL.
- [[X509_CERTIFICATE]] -- SAML Identity Provider public X509 certificate
- [[WEB_API_URL]] -- Base URL under which web api is accessible. (i.e. api.collaboard.app)

```
[
    {
        "AppCode": "webapp",
        "LoginProviders": [
            {
                "Provider": "ADFS",
                "ClientId": "[[CLIENT_ID]]",
                "ClientSecret": "[[CLIENT_SECRET]]",
                "RedirectUri": "https://[[WEB_APP_URL]]/authenticate",
                "AuthorizeUri": "https://[[ADFS_URL]]/adfs/oauth2/authorize",
                "TokenUri": "https://[[ADFS_URL]]/adfs/oauth2/token",
                "Scope": "openid email profile",
                "TFAEnabled": false
            },
            {
                "Provider": "SAML",
                "ClientId": "[[CLIENT_ID]]",
                "ClientSecret": "[[X509_CERTIFICATE]]",
                "RedirectUri": "https://[[WEB_API_URL]]/server/auth/api/Authorization/HandleExternalLoginCallback?app=webapp&provider=SAML&redirectUri=https%3A%2F%2F[[WEB_APP_URL
                "AuthorizeUri": "[[SAML_AUTH_URL]]",
                "TokenUri": null,
                "Scope": null,
                "TFAEnabled": false
            },
            {
                "Provider": "Google",
                "ClientId": "[[CLIENT_ID]]",
                "ClientSecret": "[[CLIENT_SECRET]]",
                "AuthorizeUri": "https://accounts.google.com/o/oauth2/v2/auth",
                "TokenUri": "https://www.googleapis.com/oauth2/v4/token",
                "RedirectUri": "https://[[WEB_APP_URL]]/authenticate",
                "Scope": "openid email profile",
                "TFAEnabled": false
            },
            {
                "Provider": "Microsoft",
                "ClientId": "[[CLIENT_ID]]",
                "ClientSecret": "[[CLIENT_SECRET]]",
                "AuthorizeUri": "https://login.microsoftonline.com/common/oauth2/v2.0/authorize",
                "TokenUri": "https://login.microsoftonline.com/common/oauth2/v2.0/token",
                "RedirectUri": "https://[[WEB_APP_URL]]/authenticate",
                "Scope": "openid email profile",
                "TFAEnabled": false
            },
            {
                "Provider": "Apple",
                "ClientId": "[[CLIENT_ID]]",
                "ClientSecret": "[[CLIENT_SECRET]]",
                "AuthorizeUri": "https://appleid.apple.com/auth/authorize",
                "TokenUri": "https://appleid.apple.com/auth/token",
                "RedirectUri": "https://[[WEB_API_URL]]/server/auth/api/Authorization/HandleExternalLoginCallback?app=webapp&provider=Apple&redirectUri=https%3A%2F%2F[[WEB_APP_UR
                "Scope": "name email",
                "TFAEnabled": false
            }
        ]
    }
]
```

4. Email

Email configuration. Values are set from Planning the installation step.

5. MFT

This section contains MFT related values.

Client_MFTServiceBaseUri and Server_MftUri

Those two keys are to be set to the same value. They contain closed network address of MFT API container.

Server_RemoteStorageType

Indicates what kind of storage is used. Currently only two values are supported, 0 for MFT and 1 for Azure blob storage.

Server_BaseOnPremisePath

Network path leading to where Collaboard data are being persisted. It needs to be aligned with respective mount point (see mount points here).

TempPath

Network path leading to where temp files are being shared passed along among services. It needs to be aligned with respective mount point (see mount points here).

6. Licensing

Here you will provide path to the mounted licensing file, for example '/var/overrides/license.lic'.

7. ZOOM

If you are going to use ZOOM with your installation, you need to provide ApiKey and ApiSecret for it.

Full template

```
USE [IBV.Database]
GO

DECLARE @externalUrl NVARCHAR(50) = '< EXTERNAL URL >'
DECLARE @internalUrl NVARCHAR(50) =  'http://api:8080'
DECLARE @internalMftUrl NVARCHAR(50) = 'http://mftapi:8080'
DECLARE @internalAuthUrl NVARCHAR(50) = 'http://auth:8080'
DECLARE @internalLicensingUrl NVARCHAR(50) = 'http://licensing:8080'

---- With REDIS + Realtime
DECLARE @realtimeUrl NVARCHAR(50) =  'http://realtime:8080'

DECLARE @DefaultAppDomain NVARCHAR(50) = 'CollaboardServices'
DECLARE @GenericAppDomain NVARCHAR(50) = '*'


DECLARE @Configuration TABLE ([key] NVARCHAR(35), [value] NVARCHAR(MAX), [description] NVARCHAR(250), [AppDomain] NVARCHAR(50))

INSERT INTO @Configuration
VALUES ('Client_HubServiceBaseUri', @internalUrl + '/api/collaborationhub', 'Base uri for Hub Service',  @DefaultAppDomain)
,('Client_AuthBaseUri', @externalUrl + '/server/auth/api/Authorization', 'The URI for the authorization systems',  @GenericAppDomain)
,('Server_AuthBaseUri', @internalAuthUrl + '/api/authorization', 'The URI for the authorization systems',  @GenericAppDomain)
,('Client_CfgServiceBaseUri', @internalUrl + '/api/clientconfiguration', 'URI for the client configuration',  @DefaultAppDomain)
,('Server_CanvasShotterURL', @externalUrl + '/collaboard/{0}/{1}/{2}','', @DefaultAppDomain)

---- No REDIS
--,('Client_HubUrl', @internalUrl + '/signalr', 'SignalR path',  @GenericAppDomain)
--,('Server_RedisCacheEnabled', 'false', 'Indicates if Redis cache is enabled or not', @DefaultAppDomain)

---- With REDIS + Realtime
,('Client_HubUrl', @realtimeUrl + '/signalr', 'SignalR path',  @GenericAppDomain)
,('Server_RedisCacheEnabled', 'true', 'Indicates if Redis cache is enabled or not', @DefaultAppDomain)
,('Server_RedisCacheConfig', '< PROVIDE VALUE >', 'Redis Cache Configuration', @DefaultAppDomain)
,('Server_RedisCacheServerName', '< PROVIDE VALUE >', 'Server Name fro Redis Cache', @DefaultAppDomain)

---- Security
,('Server_WebApiOTPSecret', '< PROVIDE VALUE >', 'Server Web API OTP generation security key',  @GenericAppDomain)
,('Server_WebApiOTPDurationMins', '< PROVIDE VALUE >', 'Server Web API generated OTP code duration in minutes', @GenericAppDomain)
,('Server_WebApiTokenSecret', '< PROVIDE VALUE >', 'Server Web API authentication token encryption secret key',  @GenericAppDomain)
,('Server_WebApiRefrTokenDurationMins', '< PROVIDE VALUE >', 'Server Web API refresh token duration in minutes', @GenericAppDomain)
,('Server_WebApiAuthTokenDurationMins', '< PROVIDE VALUE >', 'Server Web API authentication token duration in minutes', @GenericAppDomain)
,('Server_WebApiExternalLoginProviders', '< PROVIDE VALUE >', 'External authentication providers configuration',  @GenericAppDomain)
,('Server_CryptextKeys', '< PROVIDE VALUE >', 'Key and IV for symmetric encrypting and decrypting', @GenericAppDomain)

--- Email
,('Server_MailProviderConnectionString', '< PROVIDE VALUE >', 'Mail provider connection string', @DefaultAppDomain)
,('Server_MailTemplates', '[{
  "Theme": "default",
  "Templates": [
   { "Code": "ResetUserPassword", "Language": "en-US", "Sender": "noreply@collaboard.app", "Recipient": "{Recipient}", "Subject": "We got a request to reset your Collaboard passw
   { "Code": "VerifyUserAccount", "Language": "en-US", "Sender": "noreply@collaboard.app", "Recipient": "{Recipient}", "Subject": "Please verify your account", "Body": "<!DOCTYPE
   { "Code": "SendUserOTP", "Language": "en-US", "Sender": "noreply@collaboard.app", "Recipient": "{Recipient}", "Subject": "Here is your one-time password", "Body": "<!DOCTYPE h
   { "Code": "InviteProjectParticipant", "Language": "en-US", "Sender": "noreply@collaboard.app", "Recipient": "{Recipient}", "Subject": "You have been invited to a Collaboard pr
   { "Code": "ImportLicensedUser", "Language": "en-US", "Sender": "noreply@collaboard.app", "Recipient": "{Recipient}", "Subject": "Collaboard license activated", "Body": "<!DOCT
  ]
 }]', 'Mail templates for email sending', @DefaultAppDomain)

---- MFT Storage
,('Server_RemoteStorageType', '0', '0 = OnPrem, 1 = Azure',  @GenericAppDomain)
,('Client_MFTServiceBaseUri', @internalMftUrl, 'Uri for the MFT Service',  @GenericAppDomain)
,('Server_MftUri', @internalMftUrl, 'Uri for the MFT Service',  @GenericAppDomain)
,('Server_BaseOnPremisePath', '/var/mft/_cbfiles_', 'Base path to store project files',  @DefaultAppDomain)
,('TempPath', '/var/mft/_temp_', 'Temporary path for storing files',  @GenericAppDomain)
```

```sql
---- Licensing
,('Client_LicensingServiceBaseUri', @internalLicensingUrl + '/api/licensing', 'Uri for the licensing services',  @GenericAppDomain)
,('Server_LicensingLicenseProvider', 'Provider=FileLicenseProvider;LicenseFile=< PATH TO MOUNTED LICENSE.LIC FILE >;', 'Licensing - License provider connection string', @DefaultA
,('Server_LicensingPlans', '[{"Name":"Free","ProductId":null,"NumberOfProjects":3,"NumberOfParticipants":5,"GuestParticipantsAllowed":false},{"Name":"Personal","ProductId":1,"Num

---- ZOOM
,('Server_Zoom_ApiKey', '< PROVIDE VALUE >', 'API Key for connecting to the Zoom server', @DefaultAppDomain)
,('Server_Zoom_ApiSecret', '< PROVIDE VALUE >', 'API Secret for connecting to the Zoom server', @DefaultAppDomain)



DECLARE @Key NVARCHAR(35)
DECLARE @Value NVARCHAR(MAX)
DECLARE @Description NVARCHAR(250)
DECLARE @AppDomain NVARCHAR(50)

WHILE EXISTS(SELECT TOP (1) 1 FROM @Configuration)
BEGIN
 SELECT TOP (1) @Key = [key], @Value = [value], @Description = [description], @AppDomain = [AppDomain] FROM @Configuration

 IF EXISTS(SELECT TOP (1) 1 FROM [dbo].[t_configuration] WHERE [key] = @Key)
 BEGIN
  UPDATE
   [dbo].[t_configuration]
  SET
   [value] = @Value,
   [description] = @Description,
   [AppDomain] = @AppDomain
  WHERE
   [key] = @Key
 END
 ELSE
 BEGIN
  INSERT INTO
   [dbo].[t_configuration] ([key], [value], [description], [AppDomain])
  VALUES
   (@Key, @Value, @Description, @AppDomain)
 END

 DELETE FROM
  @Configuration
 WHERE
  [key] = @Key
END

-- Delete all caches, insert new values so we are sure it all works just fine now...
DELETE from dbo.t_caches

INSERT INTO dbo.t_caches
(
    cacheid,
    publishid,
    pollingtimeinseconds,
    cachename,
    cachedescription
)
VALUES
( 0, NEWID(), 300, N'cache', N'cache for cache' ),
( 1, NEWID(), 300, N'configuration', N'cache for configuration' )
```