

**Collaboard Install**

INSTALLING collaboard ON DOCKER

|  |  |
| --- | --- |
| Title: | Installing Collaboard on docker |
| Abstract: | This document details the specification for the installation of Collaboard in a container environment |
| Status\* | Approved |
| Document Owner: | IBV |
| Document Author: | Dennis Vroegop |
| \*Draft, In progress, Reviewed, Authorized, Approved, Completed | |

|  |  |  |
| --- | --- | --- |
| **Document Control** | | |
| Document Distribution: | Internal: | All dev Teams |
|  | External: |  |
| Document Authorization: | Authorizer(s): | Gian Paolo Santopaolo |

|  |  |  |  |
| --- | --- | --- | --- |
| **Change History** | | | |
| Date | Ver | Changed By | Details |
| 05/11/2020 | 1.0 | Dennis Vroegop | Initial Draft |
| 10/11/2020 | 1.1 | Václav Šedivec | Added container schema |
| 19/11/2020 | 1.2 | Dennis Vroegop | Added installation steps |
| 15/12/2020 | 1.3 | Dennis Vroegop | Updated to latest container version |
| 16/12/2020 | 1.3 | GP Santopaolo | Document review and approval |
| 25/01/2021 | 1.4 | Václav Šedivec | Updated with appsettings override info. |
| 28/01/2021 | 1.5 | Václav Šedivec | Updated with env.js override info. |

Index

[Installing Collaboard on Docker with Docker-Compose 3](#_Toc62742088)

[Prerequisites 3](#_Toc62742089)

[Quickstart 3](#_Toc62742090)

[Overview 4](#_Toc62742091)

[Installation 5](#_Toc62742092)

[.env file 6](#_Toc62742093)

[appsettings.Override.json 6](#_Toc62742094)

[env.Override.js 6](#_Toc62742095)

[1. Script 7](#_Toc62742096)

[2. Manual 7](#_Toc62742097)

[3. Attachments 9](#_Toc62742098)

# Installing Collaboard on Docker with Docker-Compose

## Prerequisites

1. **Docker Desktop**
   * [Docker Desktop for Windows - Docker Hub](https://hub.docker.com/editions/community/docker-ce-desktop-windows/)
   * [Docker Desktop for Mac - Docker Hub](https://hub.docker.com/editions/community/docker-ce-desktop-mac/)
2. **Azure CLI**
   * [Install the Azure CLI for Windows | Microsoft Docs](https://docs.microsoft.com/en-us/cli/azure/install-azure-cli-windows?view=azure-cli-latest&tabs=azure-cli)
   * [Install the Azure CLI for macOS | Microsoft Docs](https://docs.microsoft.com/en-us/cli/azure/install-azure-cli-macos?view=azure-cli-latest)
3. **Access to containers registry.**
   * Contact DevOps/Azure team to request access to your account.
4. **Access to documentation repository.**
   * Contains docker-compose, scripts, and localhost certificate.
   * Contact DevOps/Azure team to request access to your account.

## Quickstart

1. Install all prerequisites.
2. Ensure you have the required access to resources.
3. Start Docker Desktop.
4. Git clone the Distribution repository
   * **HTTPS:** git clone https://ibvsolutions@dev.azure.com/ibvsolutions/CollaboardServices/\_git/Distribution
   * **SSH:** git clone git@ssh.dev.azure.com:v3/ibvsolutions/CollaboardServices/Distribution
5. Adjust variables in .env file.
6. Prepare appsettings.Override.json file if necessary.
7. Run **update-and-run.sh** or **update-and-run.ps1** script.
   * If you have yet to log in for the first time, it will ask for credentials.
   * You will be asked whether to update DB. The safe choice is yes (just hit enter), unless you know you don't need it.
8. Make sure all containers are up and running (*docker container ls -a*)
9. Access website on <https://localhost:8443/>
   * You might get error code 400 if you try using HTTP. In that case, refresh the page, and it will redirect to HTTPS.
   * When you browse the page, if you haven't installed the self-signed certificate on your machine, you might be informed that the site is not secure. You shall proceed

## Overview

These are the containers we currently use in our landscape:



|  |  |  |
| --- | --- | --- |
| Name | Type | Remarks |
| Nginx | Proxy server | Handles all traffic from the outside world to the internal containers running on the Docker network "CBNetwork". This is the only open part of the system. |
| API | HTTP Server | Main API for the system. Reached from the outside through the proxy via https://<server>/server/api |
| API/SignalR | HTTP Server | The realtime component. Reached from the outside through the proxy via https://<server>/server/signalr |
| Licensing | HTTP Server | Takes care of the licensing handling. Reached from the outside through the proxy via https://<server>/server/licensing |
| Auth | HTTP Server | Handles all authentication. Reached from the outside through the proxy via https://<server>/server/auth |
| MFTApi | HTTP Server | Handles all file and bloc storage and retrieval. Reached from the outside through the proxy via https://<server>/mft |
| Frontend | HTTP Server | The main frontend server, serving the React files. Reached from the outside through the proxy via https://<server>/ |
| DB | Database | Database server. Not accessible from the outside. Internal open port 1443. Running SQL Server 2017 |
| Copy | Background service | Handles the asynchronous copying of tiles and projects. |
| Image Resizer | Background service | Handles the resizing off the images for a responsive system. |
| File Converter | Background service | Handles the asynchronous conversion of Office files to images for thumbnail display. |
| MFT Merge | Background service | Handles the merging of all bits retrieved for the blob storage |
| MFT Storage Service | Background service | Handles the asynchronous tasks for handling all blob storage |
| MFT Cleaner | Background service | Handles the asynchronous tasks for cleaning up the residual files for the blob storage |
| Worker 1 | Background service | Assorted background processes |
| Worker 2 | Background service | Assorted background processes |
| Canvas Shotter | Background service | Handles the asynchronous generation of thumbnails for the projects |

# Installation

There are two ways to install the containers. One is manual, second is by using one of the prepared script. For both .env file is used:

### .env file

The .env file contains set of variables that can be changed before execution:

* **ENV\_VERSION** -- basically defines which version should be downloaded, default is **latest** (=production).
* **SQL\_DATA** -- path to folder where SQL persists database data.
* **SQL\_LOGS** -- path to folder where SQL persists database logs.
* **SQL\_SCRIPTS** -- path to folder where SQL configuration script is. It is required in case that script filename is defined.
* **SQL\_SCRIPT\_FILE** -- filename of SQL configuration script.
* **CB\_OVERRIDES** -- path to folder with appsettings override file.
* **CB\_OVERRIDE\_FILE** -- filename of appsettings override file.
* **CB\_OVERRIDE\_ENV\_JS** -- filename of frontend env.js override file.
* **CB\_OVERRIDE\_NGINX** – filename of proxy nginx.conf override file.
* **CB\_FILES** -- path to folder where application persists project files.
* **CB\_TEMP** -- path to folder where application stores temporary data.
* **SSL\_CERTS** -- path to folder with certificate files (\*.crt + \*.key).
* **SA\_PASSWORD** -- password for sa user of the database.

*!! IMPORTANT !!*  
*If you change the ENV\_VERSION after having it already installed, the new environment will be installed side by side. In order for both of them stay functional you must change the path to database data as well. Also only one of them can be running at a time. There would have to be some more adjustments if we want the script include that.*

### appsettings.Override.json

This file is **optional** and it is loaded on startup on top of the default appsettings.Containers.json (overloading only specified variables). It holds connection strings, CORS settings and logging settings.

Example of override: [3.1. appsettings.Override.json example](#_3.1._appsettings.Override.json_exam)

Such file can be reused for multiple containers since these settings are often identical.

### env.Override.js

This file is **optional** and it is loaded on startup. It completely rewrites the original env.js. It holds various URLs (server api, mft api, ...) and other environment dependent values. It also contains feature flags.

Example of override: [3.2. env.Override.js example](#_3.2._env.Override.js_example)

## 1. Script

In order to have the system running/updated as easily as possible, there are two scripts prepared. Both contain the same steps, it depends on operating system which one it can use.

* **update-and-run.ps1**
* **update-and-run.sh**

Scripts require following files/folders be present in the same directory:

* **.env** – contains adjustable variables
* **cb-proxy.crt**, **cb-proxy.key** – their location can be changed in .env file, they represent SSL certificate used for https communication
* **docker-compose.yml** – compose file declaring the system structure and configuration

First check the .env file values, whether you want to change some paths, version etc.  
If you want to change the path where SQL server stores the database data to for example /usr/tmp/sql\_data, then you need to change the line

SQL\_DATA=c:/\_SQL\_/\_DATA\_ to SQL\_DATA=/user/tmp/sql\_data

After script execution, there might be prompt for login. After login there will be prompt asking whether to update database as well. Always update database unless you know there were no changes. But if skipped, it saves a lot of time.

Re-running the script will update to latest version, if available. Database is always updated unless skipped.

## 2. Manual

*Following are basically manual steps that are part of the previous scripts. Replace the C:\\_SQL\_ and similar values with their equivalent from .env file. Also adjust the .env file accordingly before doing these steps.*

*The --env-file=.env parts can be removed, .env is included by default but it might come in handy in case of multiple files for various envs.*

**2.1. Prepare database**

The data in the database server is stored in the share C:\_SQL\_. So this folder should contain all necessary files for SQL to work properly.  
To get those files, one need to run the following command once:

docker run -it --env-file=.env -v c:/\_SQL\_/\_DATA\_:/var/opt/mssql/data -v c:/\_SQL\_/\_LOGS\_:/var/opt/mssql/logs ibvcollaboard.azurecr.io/cb\_database\_init:latest

This will pull the correct image and run the container. Inside the container are all the commands needed to create the databases. After this is done, the system will be ready and we don't need this container anymore.

**2.2. Install/Update containers**

When you have the main docker-compose.yml file, all you need to do to get a up and running system is issue the command:

docker-compose --env-file=.env up -d

This will initially draw all images from the repository, this might take a long time! Once this is done, the system will start and Collaboard can be used by going to a browser and opening the page [https://localhost:8443](https://localhost:8443/)

There needs to be SSL certificate pair present on relative path to the docker-compose.yml. Path can be changed in the .env file.

* certs\localhost\cb-proxy.crt
* certs\localhost\cb-proxy.key

Subsequent updates can be done using these commands:

docker run -it --name cb\_database\_init --env-file=.env -v c:/\_SQL\_/\_DATA\_:/var/opt/mssql/data -v c:/\_SQL\_/\_LOGS\_:/var/opt/mssql/logs ibvcollaboard.azurecr.io/cb\_database\_init:latest  
docker-compose --env-file=.env pull  
docker-compose --env-file=.env up -d

## 3. Attachments

### 3.1. appsettings.Override.json example

{

"ConnectionStrings": {

"CollaborationHubDB": "Server=db,1433;Initial Catalog=IBV.Database;Persist Security Info=False;User ID=sa;Password=Welcome@2;Integrated Security=false;Connection Timeout=30;",

"IdentityDB": "Server=db,1433;Initial Catalog=IBV.Database;Persist Security Info=False;User ID=sa;Password=Welcome@2;Integrated Security=false;Connection Timeout=30;",

"ConfigurationDB": "Server=db,1433;Initial Catalog=IBV.Database;Persist Security Info=False;User ID=sa;Password=Welcome@2;Integrated Security=false;Connection Timeout=30;",

"LicensingDB": "Server=db,1433;Initial Catalog=IBV.Database;Persist Security Info=False;User ID=sa;Password=Welcome@2;Integrated Security=false;Connection Timeout=30;",

"MFT\_DB": "Server=db,1433;Initial Catalog=IBV.Database;Persist Security Info=False;User ID=sa;Password=Welcome@2;Integrated Security=false;Connection Timeout=30;"

},

"Cors": {

"AllowedOrigins": [

"https://localhost:8082",

"chrome-search://local-ntp"

]

},

"Logging": {

"LogLevel": {

"Default": "Trace",

"Microsoft": "Warning",

"Microsoft.Hosting.Lifetime": "Warning"

}

}

}

### 3.2. env.Override.js example

window.runtimeConfig = {

apiUrl: "https://localhost:8443/server",

apiTimeout: 20000,

signalRLogging: false,

signalRPath: "/signalr",

appName: "webapp-dev",

env: "local",

isOnPremise: true,

onPremiseMftApiUrl: "https://localhost:8443/mft",

helpPageUrl: "https://help.collaboard.app/",

stripePublishableKey: "pk\_test\_1BLtPBMDSb3WAaEt9NPT9cov00wmdcZEHB",

youtubeApiKey: "AIzaSyDWN\_M1HYRy9cDQfhP2mDqWPuw1oMqnsaE",

customer: "default",

authProviders: ["Google", "Microsoft", "ADFS", "Apple"],

features: {

signalR: true,

inkSplitting: false,

ink: true,

embed: false,

voting: true,

history: true,

documents: true,

videos: true,

videoCrop: true,

useCompositeTogetherWithInk: false,

imageCrop: false,

downloadModified: false,

copyPaste: true,

comments: false,

clickableURLs: true,

zoomWarning: true,

autoAlignment: true,

disableConsoleLogs: false,

connectionToInk: false,

areaSelection: true,

selectObjectInGroup: true,

connectObjectInGroup: true,

embedFonts: false,

shareModal: true,

batchUserImport: false,

teamPage: true,

plansPage: true,

licensing: true,

stripeAntiFraud: false,

deviceLicenses: false,

generateProjectThumbnail: true,

inkSizeIndependentOfZoom: false,

templates: true,

inkCacheLoD: true,

testUtils: true,

drawOnlyVisibleSegments: true,

textInShape: true,

activatePenOnTouch: false,

presentationMode: true,

canvasMouseTracker: false,

inkV2: true,

panOnSpace: true,

selectionMode: true,

newSidebars: true,

objectToolbarV2: true,

drawingToolbar: true,

},

maxSignalRMessageSizeInKB: 64,

bigInkLowerBoundary: 500,

bigInkUpperBoundary: 1500,

useCacheForConnectionAnchor: true,

checkForNewVersionInterval: 0, // minutes

autoAlignZoomRatio: 50, // the size zoom ratio for which auto alignment is enabled

autoAlignSnapRadius: 3, // in this radius the objects will snap to each other // smoothness of the auto align

monitoringSignalRQueueGradientThreshold: 1,

timerForEditableObjectBorder: 1500,

maxAnimatedObjectsNumber: 15,

};