You can utilize the Amazon S3 storage service for storing your website files even though the website is hosted on-premise, on your own server.

> **File name case**
>
> Unlike standard Windows file systems, the Amazon S3 storage is case-sensitive. To ensure consistent behavior, Kentico automatically converts all file and folder names to lower case when processing files on Amazon S3.

Configuring Kentico to use Amazon S3

1. Make sure that you have your Amazon S3 account set up and that you have created at least one bucket. Amazon Management Console.
2. Add the following key into the *appSettings* section of your project's **web.config**.

```
<add key="CMSExternalStorageName" value="amazon" />
```

> **Important**: Setting the **CMSExternalStorageName** key maps the entire file system of your application to the Amazon S3 storage. Do NOT set this key if you only wish to map specific folders (for example media file folders).
>
> To map a specific folder, define an Amazon S3 storage provider as described in the Storing files in different buckets section.

3. Add the following key to specify the bucket that you want to use to store files. Replace the value with the name of the bucket.

```
<add key="CMSAmazonBucketName"  value="YourBucketName" />
```

4. Specify the ID of your Amazon access key by inserting the following key:

```
<add key="CMSAmazonAccessKeyID" value="YourKey" />
```

5. Specify the Amazon access key by adding the following key:

```
<add key="CMSAmazonAccessKey" value="YourSecret" />
```

After you save your web.config, Kentico starts storing files in the specified Amazon S3 bucket.

> **Additional website settings**
>
> When configuring this type of storage, keep in mind that the website itself must be configured to store files in the file system rather than in the database only. In **Settings -> System -> Files** enable the **Store files in file system** option.
>
> It is also recommended to enable **Redirect files to disk** in **Settings -> System -> Performance**. This means that files will be requested from the Amazon S3 account rather than from the database (if possible).

Also, see the Media library notes at the end of this page for additional information about specifics of media libraries when using Amazon S3 storage.

You can configure the following optional settings:

Key	Description	Sample Value

CMSAmazonTempPath	Path to a local directory that the system uses to store temporary files. **Default value**: <installation root>\CMS\App_Data\AmazonTemp	`<add key="CMSAmazonTempPath" value="C:\Windows\Temp" />`
CMSAmazonCachePath	Path to a local directory where the provider stores cached files. **Default value**: <installation root>\CMS\App_Data\AmazonCache	`<add key="CMSAmazonCachePath" value="C:\Cache" />`
CMSAmazonEndPoint	Allows you to change the default URL of the Amazon S3 Website Endpoint. For example, you can change the endpoint if you want to use CloudFront CDN. **Default value**: http://<yourbucketname>.s3.amazonaws.com	`<add key="CMSAmazonEndPoint" value="http://someendpoint.s3.amazonaws.com" />`

CMSAmazonRestApiEndPoint	Allows you to change the default URL of the Amazon S3 REST API Endpoint. The system uses the REST API to determine the region of buckets. **Default value**: https://s3.amazonaws.com	`<add key="CMSAmazonRestApiEndPoint" value="http://s3.amazonaws.com" />`
CMSAmazonPublicAccess	Specifies whether files uploaded to Amazon S3 through Kentico are accessible for public users. **Default value**: *true* if you specify an endpoint *false* If no endpoint is specified	`<add key="CMSAmazonPublicAccess" value="true" />`

Storing files in different buckets

By default, the system stores files in a single bucket. However, the built-in Amazon S3 storage provider allows you to map sections of the file system to different buckets.

1. Open the Kentico web project in Visual Studio (using the **WebSite.sln** or **WebApp.sln** file).
2. Create a custom module class.
   - Either add the class into a custom project within the Kentico solution (recommended) or directly into the Kentico web project (into a custom folder under the **CMSApp** project for *web application* installations, into the **App_Code** folder for *website* installations).

   > For basic execution of initialization code, you only need to register a "code-only" module through the API. You do NOT need to create a new module within the **Modules** application in the Kentico administration interface.

3. Override the module's **OnInit** method and perform the following:
   - Create a new instance of the Amazon S3 storage provider.
   - Specify the target bucket using the **CustomRootPath** property of the provider.
   - (Optional) You can specify whether you want the bucket to be publicly accessible using the **PublicExternalFolderObject** property of the provider. True means the bucket is publicly accessible.
   - Map a directory to the provider. This is the directory that you want to store in the bucket.

```
using CMS;
using CMS.Base;
using CMS.DataEngine;
using CMS.IO;

// Registers the custom module into the system
[assembly: RegisterModule(typeof(CustomInitializationModule))]

public class CustomInitializationModule : Module
{
        // Module class constructor, the system registers the module under the
name "CustomInit"
        public CustomInitializationModule()
                : base("CustomInit")
        {
        }

        // Contains initialization code that is executed when the application
starts
        protected override void OnInit()
        {
                base.OnInit();

                // Creates a new StorageProvider instance for Amazon S3
                var mediaProvider = StorageProvider.CreateAmazonStorageProvider();

                // Specifies the target bucket
                mediaProvider.CustomRootPath = "mymediabucket";

                // Makes the bucket publicly accessible
                mediaProvider.PublicExternalFolderObject = true;

                // Maps a directory to the provider
                StorageHelper.MapStoragePath("~/MySite/Media/", mediaProvider);
        }
}
```

4. Save the file. If your project is installed in the web application format, rebuild the solution.

Configuring Kentico to use Amazon CloudFront CDN

A Content Delivery Network (CDN) speeds up distribution of content to the end users through a network of data centers. See Amazon CloudFront Product Details to learn more.

To start using the Amazon CloudFront service with Kentico:

1. Create a **CloudFront Distribution**. You can use the Amazon Management Console. Select your Amazon S3 storage bucket as the **Origin Domain Name**.
2. Edit the **web.config** file of your Kentico project.
3. Add the **CMSAmazonPublicAccess** and **CMSAmazonEndPoint** keys into the web.config file:

```
<appSettings>
        <add key="CMSAmazonPublicAccess" value="true" />
        <add key="CMSAmazonEndPoint" value="EndpointURL" />
</appSettings>
```

4. Set the value of the **CMSAmazonEndPoint** key to the **Domain Name** URL of your created CDN.

> ⚠ If your site runs under **HTTPS**, we recommend always specifying the endpoint URL with the **HTTPS** protocol as well.
>
> For example: *https://domain.cloudfront.net*
>
> Not doing so may result in Mixed Content warnings being logged in your web browser's console when retrieving files from the CDN.

Your project now starts using the created CDN service.

Media library notes

Using Amazon S3 storage for your project has some effects on media libraries.

Storing too many files in one media library folder

Storing a large number of media files in a single folder can significantly affect the performance of your project when editing the files in the Media library application. See Media library limitations when storing files in an external storage for details.

Using the same bucket names when CMSAmazonPublicAccess is set to true

If all following conditions are true:

- you use Amazon S3 as external storage
- you set the *CMSAmazonPublicAccess* key to *true*
- you want to use content staging

then the buckets on all storages must have the **same name**.

The reason is that when using the *CMSAmazonPublicAccess* key, direct file links use the name of the bucket in the URL. Therefore, if your staging server is connected to a different storage, the buckets where the media library files are stored must have the same name in all storages.