Specification for hosting .NET core samples online

## Aim

As of now, we share samples to the customer in zipped format and that is error prone in terms of system configuration, software requirements to download and run the given samples.

To make the process of sharing of sample to the customer, we have decided to make a web application and host in a common server that targets more than one Syncfusion versioned assemblies, thereby making the sample sharing hassle free.

This way user(developer) can add any number of samples to the hosted web application and share the link to the customer, making the process much simpler.

## Current analysis

This application could be created by having separate project files for each version and add samples to the respective projects that targets the required framework

For eg., say we need to share a sample with Datepicker component in two versions of Syncfusion.EJ.AspNet.Core assembly (16.1.0.37, 16.2.0.41). For this we’ll have two projects that runs with the respective versions.

Developer should add sample in both the projects and share the hosted link that points to the sample.

## Usage

The application will be available in a common server that will be accessible via remote access.

The developer could add or delete files to the corresponding projects via remote access and rerun the application. The generated link can be shared to the customer as such.

## Things to consider

**Server maintenance**

Since we need to keep our application updated to the current version, we need to keep the targeted versions of the application to a limited number.

Also, for each release, we should update the framework versions of the application and that must be tested in a staging machine before making it to be available online.

To keep track of the changes in application, we could use SVN server. This will help us have version control of our app.

**Sample** **maintenance**

We also must maintain the number of days the samples should be kept on the server to keep our application light weight.

To keep control of the additional assemblies and assets being added to the application,

1. **In terms of application configuration**: we could keep a separate ‘JSON’ file that contains additional project settings such as adding control specific assembly files to the project etc.,
2. We should restrict editing of certain files like ‘launchsetting.json’, ‘hostsettings.json’, ‘bower.json’ files etc by means of folder or file security settings.

**Samples automation**

After adding samples, we need to automate the samples to test if they are error free.

**Add default samples**

To make process simple, we could add default samples for all the controls and make it ready to edit in future.

**Performance**

To ensure improved performance, cache all the downloaded assets as much as possible

Optimize the assets by using the bundlers available

Avoid having images of generous size

**Security of generated links**

To make sure the links are redirected to the expected page in customer end, use ‘rel=’’noopener noreferer’ attribute in forums