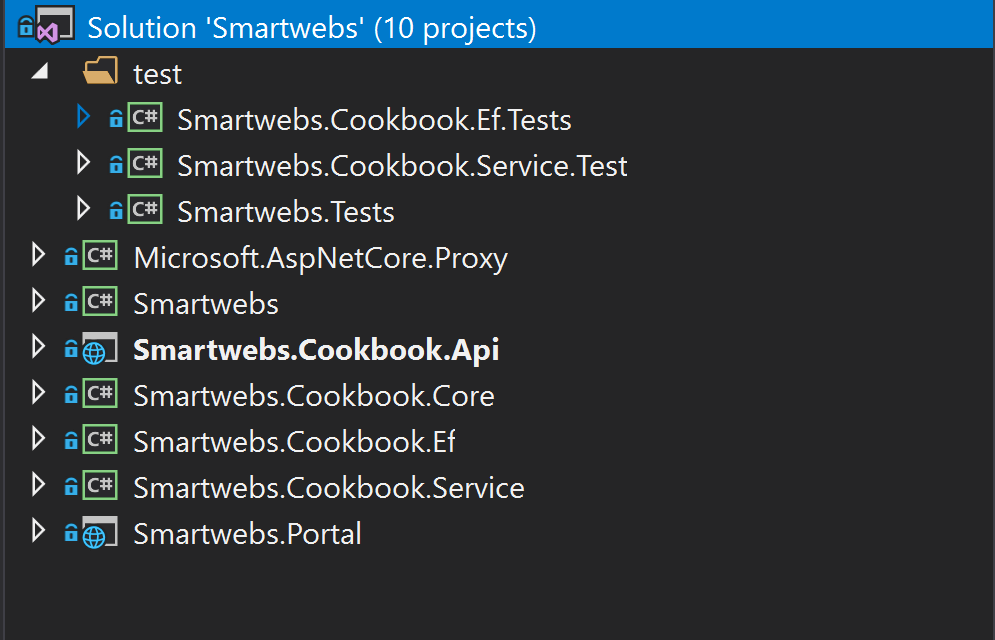
1. **Structure**

Base on this structure we can implement with multiple services such as Payment (Smartweb.Payment.xxx ) and deploy to docker.

* Framework:
  + Smartwebs: base framework project includes all share class and methods.
* Domain layer:
  + Smartwebs.Cookbook.Core: contain domain or entity
  + Smartweb.Cookbook.Ef: using EF6, you can use EF Core and inject it into Smartweb.Cookbook.Api. This project implement CRUD.
* Business layer:
  + Smartweb.Cookbook.Service: implement business logic.
* Apis:
  + Smartweb.Cookbook.Api: public apis.
* Presentation layer:
  + Smartwebs.Portal: using angular 5. In this portal, we can call multiple services/apis by using Microsoft.ApsNetCore.Proxy or can use Træfik for configuration.

1. **Database structure**

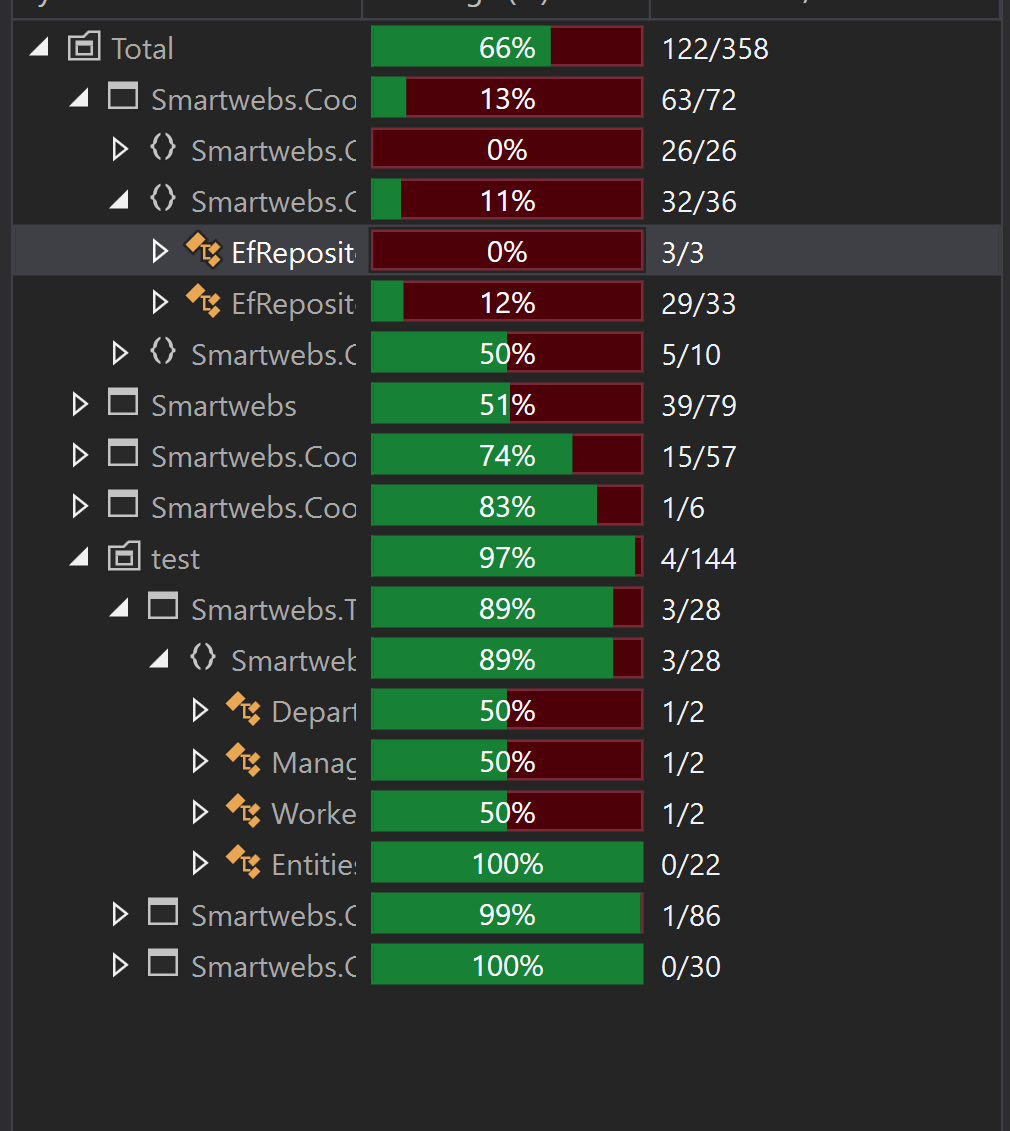
The first, I want to use only one table to store Recipes and RecipeVersions but I think it is not good for update recipe. Then I change to use two tables:

* Recipes: store last recipe when you create/update.
* RecipeVersions: store all historical of recipe include original recipe when you insert (RecipeId is null)

The benefit here for performance update and get list current recipes but is not good you have to store original recipe in RecipeVersions table (but I think it is not important than you have to store in 1 table or two tables without original recipe – have to join)

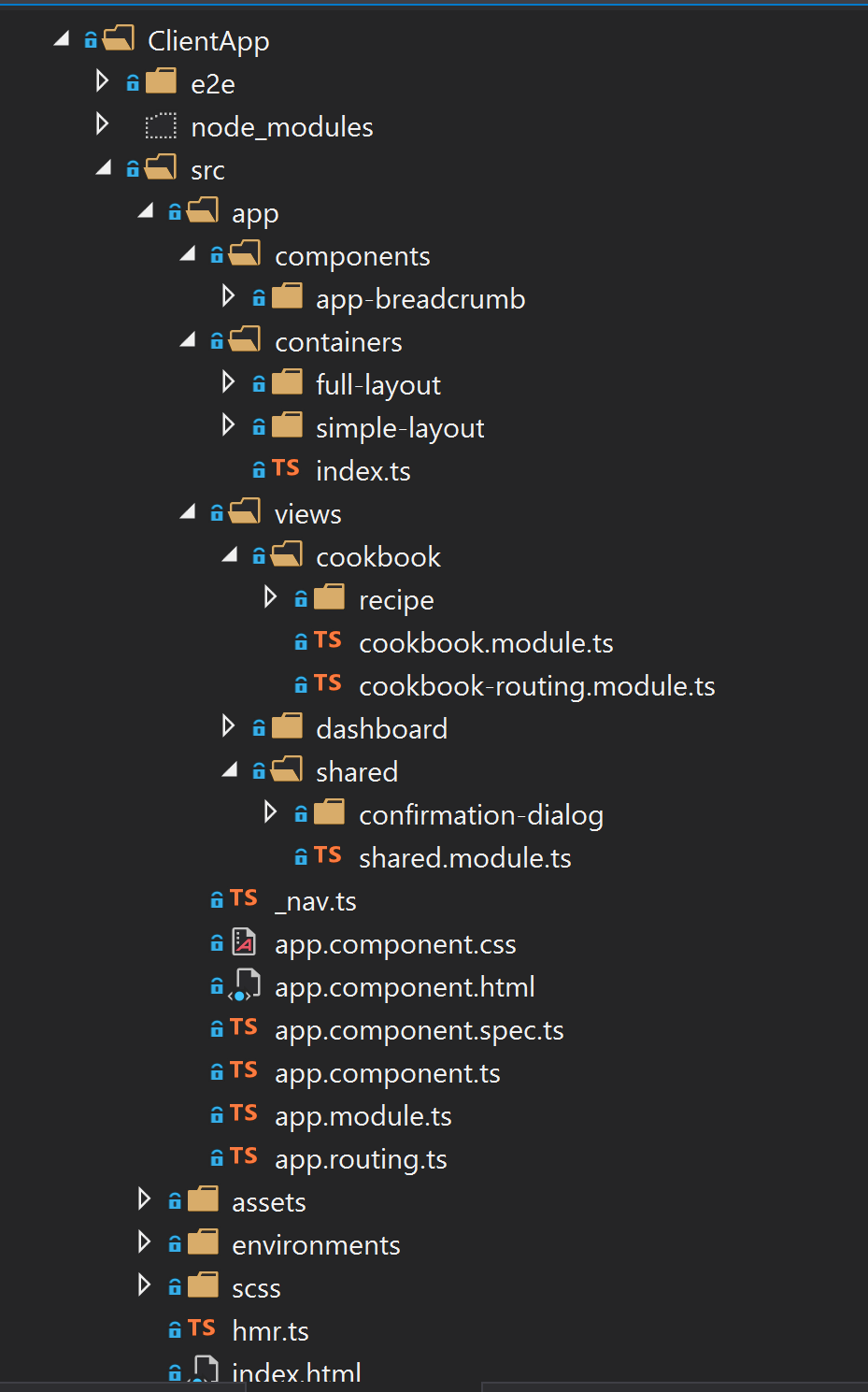
1. **Test**

* Server side:
  + Using XUnit test for unit test and FakeItEasy for integration test.
  + Coverage about 66% because I do not have time but I can write more



* Client side: I am so busy for my work in this week. So I only write about 10 unit test.

1. Front-end

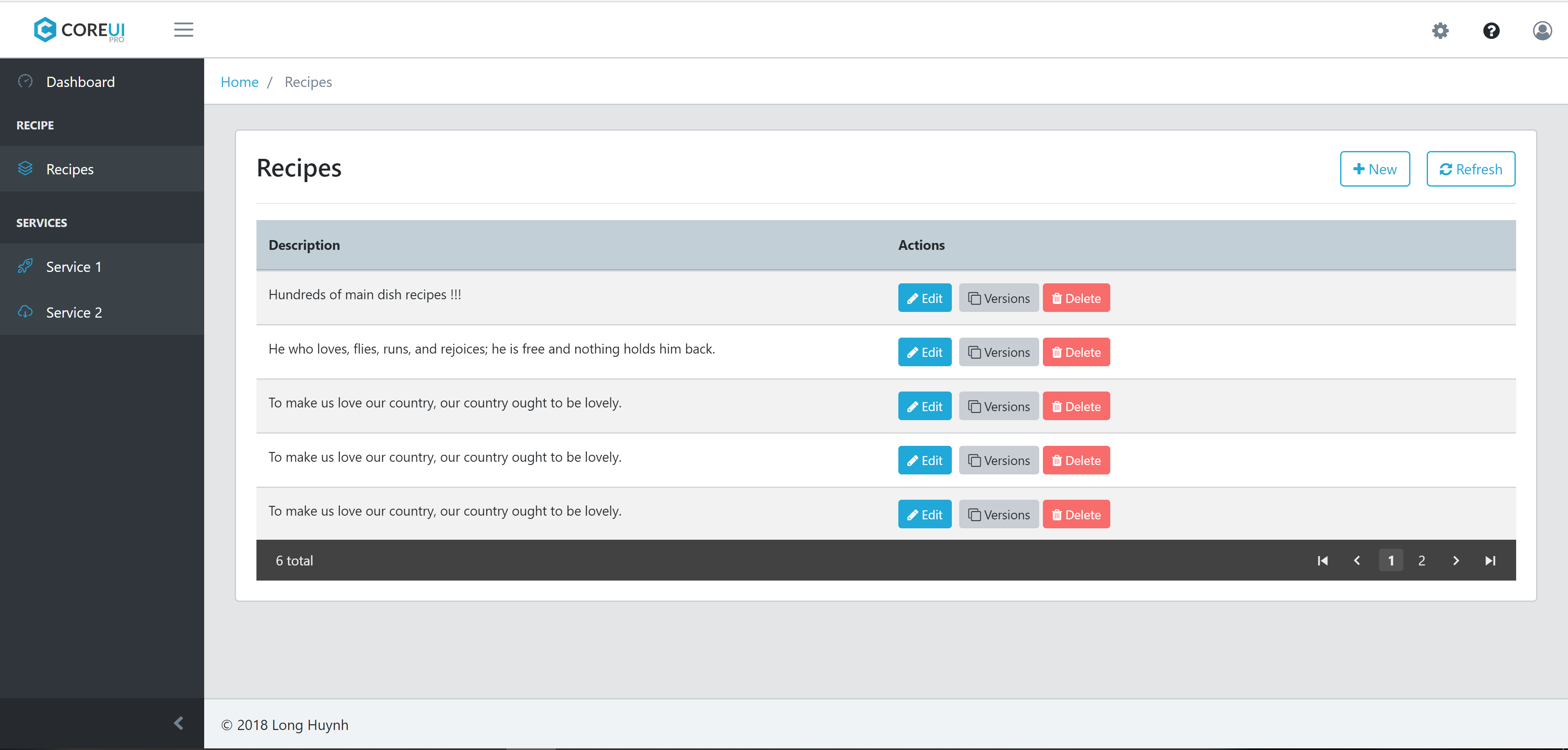
* Using Angular Asp.Net Core include webpack and bundle
* Using CoreUI
* Structure of project
  + We can use multiple layout (**containers** folder)
  + Every feature view define by module eg. cookbook, payment (in **views** folder)
  + Other module/components will be contained in **shared** folder.

1. How to run

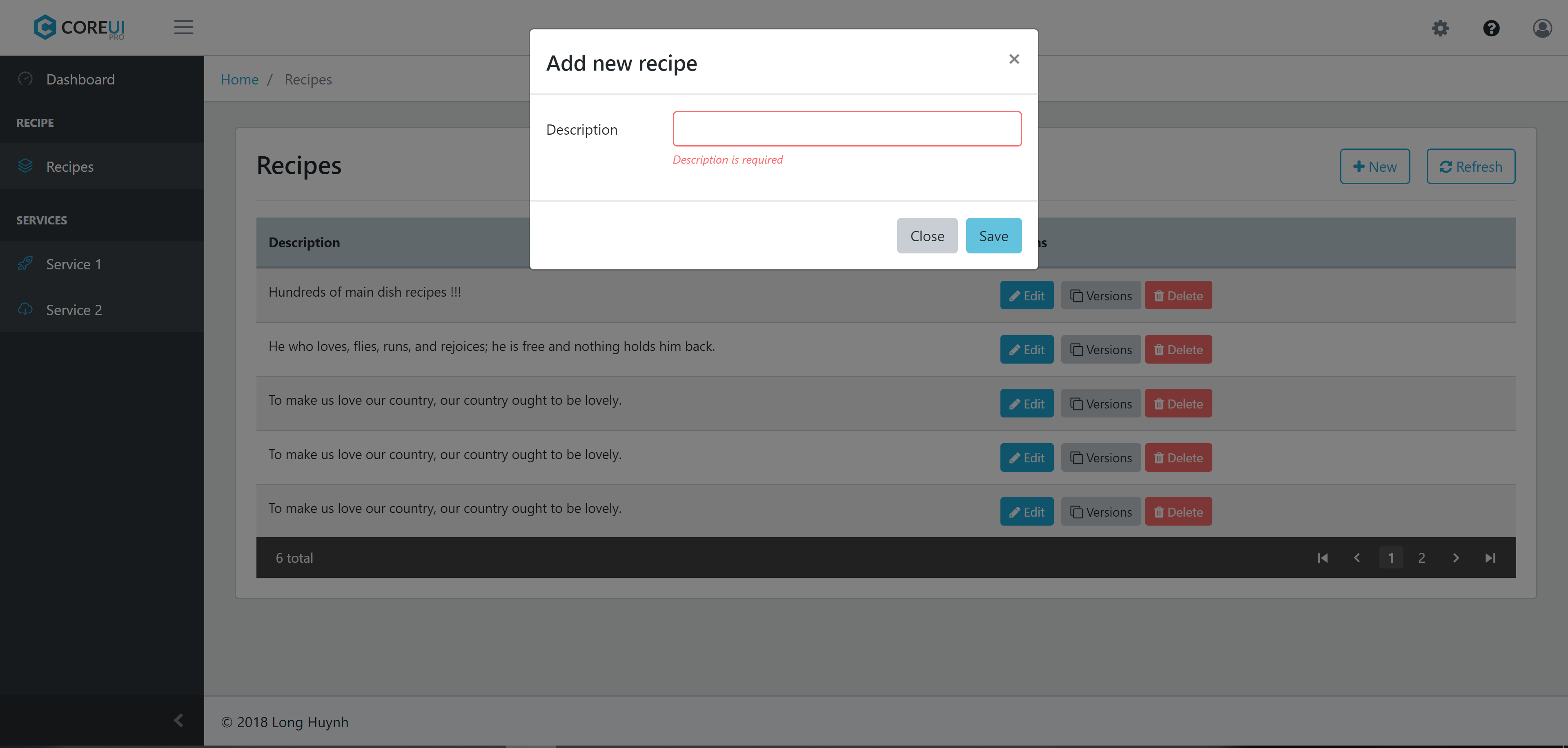
* Update-Database with **Smartwebs.Cookbook.Ef** project.
* Run cmd on Smartwebs.Cookbook.Api folder with command “dotnet run” to run api.
* Navigate to ClientApp folder in Portal project and run “npm install”
* “dotnet run” to run Portal.

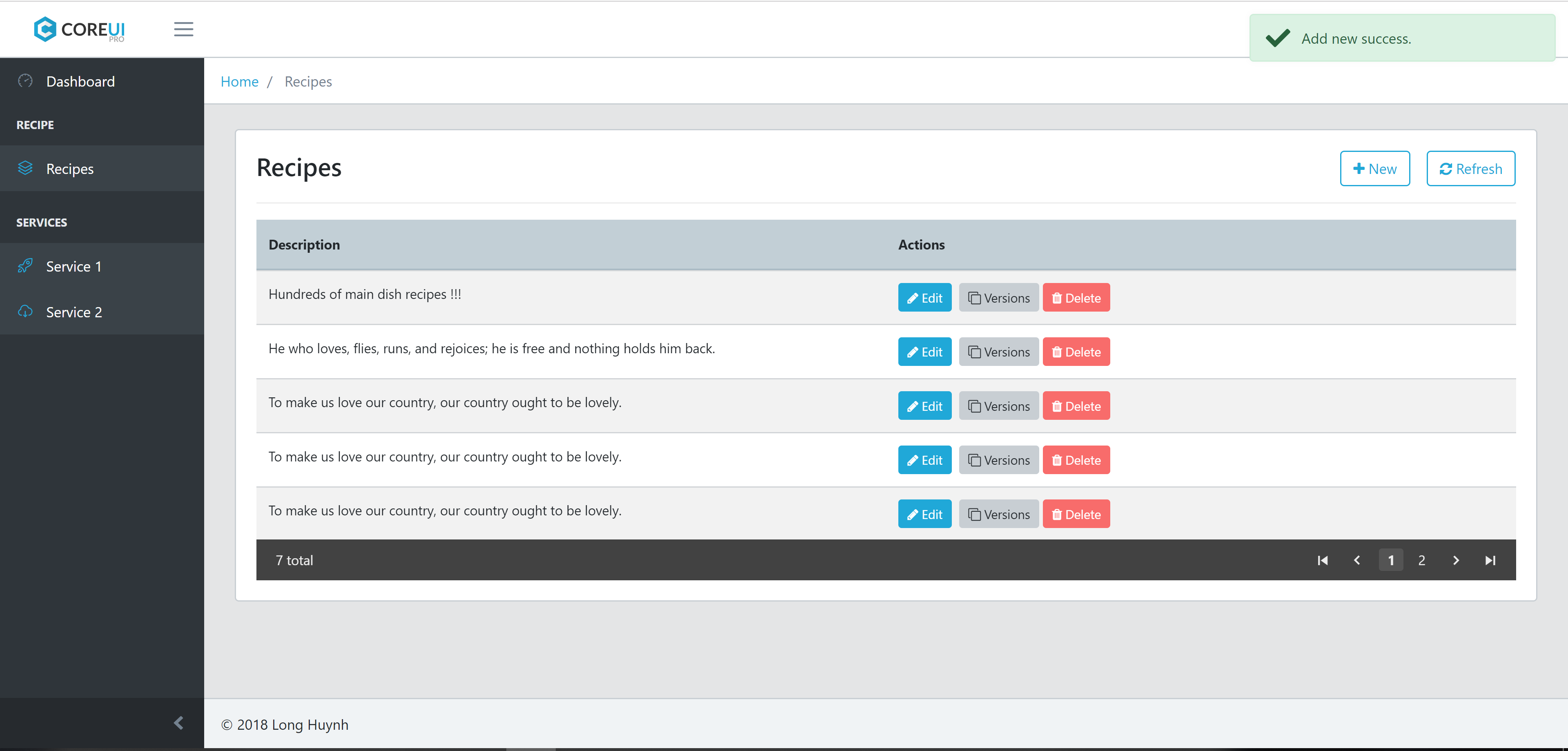
1. UI

* Recipe list

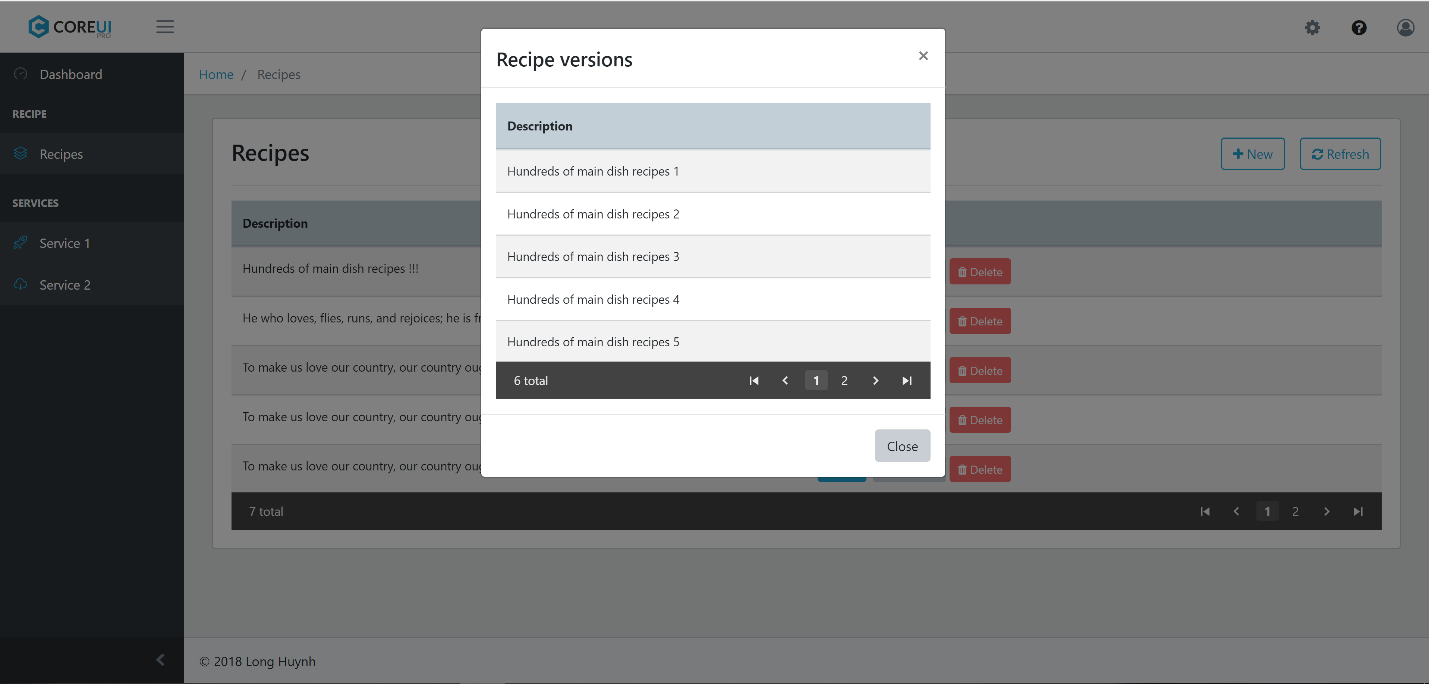


* Add new recipe





* View recipe versions



* Delete with confirmation UI (share for reuse)

