Wszystkie poniższe funkcjonalności warstwy prezentacji zostały zaimplementowane za pomocą frameworka AngularJS (patrz…). Każda z funkcjonalności została podzielona na 4 warstwy :

1. **Kontroler –** warstwa odpowiedzialna za logikę działania aplikacji oraz za komunikowanie się z warstwą serwisu. Instancje poszczególnych serwisów są wstrzykiwane za pomocą wbudowanego mechanizmu Dependency Injection
2. **Serwis –** warstwa odpowiedzialna za komunikowanie się z serwerem ASP .NET MVC. Każdy z serwisów również wykorzystuje wbudowany mechanizm DI w celu korzystanie z innych serwisów lub wbudowanych w Framework gotowych rozwiązań
3. **Widok –** warstwa odpowiedzialna za prawidłowe wyświetlanie informacji użytkownikowi oraz interakcję z tymże użytkownikiem. Z założenia widok nie ma bezpośredniego połączenia z serwisem. Odpowiednia komunikacja jest realizowana poprzez kontroler.
4. **Dyrektywy –** Narzędzia które z założenia mają być samodzielnymi modułami. Mogą jedynie komunikować się z warstwą serwisu w celu pobierania lub wysyłania odpowiednich danych.

W celu lepszej organizacji i skalowalności kodu każda z warstw została umieszczona w osobnym folderze, a każdy obiekt swojego typu w osobnym pliku z rozszerzeniem .js

1. Mechanizm paczkowania

W celu poprawnego działania każdej aplikacji napisane w JavaSript należy do strony HTML dodać link wskazujący na plik z rozszerzeniem .js. Platforma .NET w łatwy i przyjemny sposób ułatwia to zadanie. Dzięki klasie BundleConfig jesteśmy w stanie paczkować wiele plików js jaki css. Zwiększa to wydajność ładowania strony jak i minimalizuje wielkość plików ładowanych na stronie. Na poniższej ilustracji przedstawiono kilka przykładowych paczek plików js



Paczki dołączane są do widoku poprzez użycie metody Scripts.Render Co widać na poniższej ilustracji.



1. Moduł aplikacji



Cała warstwa prezentacji została zawarta w specjalnym obiekcie dostarczanym w frameworku AngularJS o nazwie app. Obiekt odpowiada za działanie całej aplikacji, Abu korzystać z kontrolerów AngularJS należy je umieścić wewnątrz elementu HTML który jest opatrzony specjalną dyrektywą ng-app co pokazano na poniższej ilustracji.



1. Serwisy

Każdy serwis w aplikacji komunikujący się z serwerem ASP .NET MVC jest zaimplementowany w podobny sposób. Jest to obiekt zawierający zbiór metod które wysyłają i odbierają informacje z serwera. Każda z takich metod korzysta z wbudowanego w framework serwisu $http który odpowiedzialny jest za samą komunikację z serwerem. Metody te korzystają również z obiektu $q który wspomaga asynchroniczne operacje.



1. Walidacja formularzy

Walidacja formularzy odbywa się za pomocą wbudowanego mechanizmu AngularJS. W elemencie HTML form umieszczane są kontrolki z nadaną nazwą. W ten sposób w obiekcie formu udostępniana jest lista obiektów z błedami walidacyjnymi. Jeśli w danym obiekcie pojawi się jakikolwiek błąd wyświetlana jest dla niego wiadomoś. Gdy jakikolwiek z elementów jest nie poprawny to cały formularz również staje się nie poprawny i za pomocą dyrektywy ng-disabled blokowany jest przycisk do potwierdzenia danych. Mechanizm ten przedstawia poniższa ilustracja:



1. Podstawowe operacje CRUD

Każda z encji dostępnych w aplikacji ServiceCMS posiada swój kontroler który udostępnia podstawowe metody obsługujące operacje typu CRUD. Ze względu na duże podobieństwo tych operacji wybrano jeden kontroler NewsController który posłuży za przykład. Akcje są wywoływane za pomocą przycisków i specjalnej dyrektywy ng-click



* 1. Odświeżanie

Podstawową funkcjonalnością związaną z każdą encją jest wyświetlanie tychże encji.

W każdym kontrolerze odpowiedzialna jest za to metoda refresh() która za pomocą serwisu pobiera aktualne encje.



Metoda NewsService.getNewsestNewses() (jak każda inna metoda serwisu) zwraca tak zwany obiekt – obietnicę dzięki opisanemu wcześniej obiektowi $q. Obiekt ten posiada metodę then() która jako argumenty przyjmuje kolejno : metodę kiedy żądanie się powiedzie i metodę kiedy żądanie się nie powiedzie. Metody tego typu przypisują obiekt zwrócony z serwisu do lokalnego obiektu dostępnego w ramach aktualnego kontrolera, a następnie dzięki specjalnemu, wbudowanemu mechanizmowi wiązania są wyświetlane w widoku za pomocą dyrektyw ng-repeat i ng-bind.



* 1. Dodawanie

Operacja dodawania encji jest obsługiwana za pomocą metody add(). Metoda ta korzysta z obiektu $modal w celu otwarcia nowego okna i przekazuje kontrole do podanego kontrolera oraz za pomocą metody then() dostępnej w obiekcie $modal.result ustawia zachowanie po zamknięciu okna, w tym przypadku jest to dświeżenie listy danych encji.



Kontroler nowego okna posiada 2 podstawowe metody send() i cancel(). Metoda send() wysyła poprzez serwis dane wprowadzone przez użytkownika i zamyka okno. Natomiast metoda cancel() jest odpowiedzialna za zamknięcie okna bez wysyłania danych.



Przykładowy widok dodawania przedstawia poniższa ilustracja:



* 1. Edytowanie

Edytowanie encji jest realizowane za pomocą metody edit(). Metoda ta działa w analogiczny sposób co metoda add() z tą różnicą że nie tworzy nowego obiektu a przekazuje (poprzez obiekt resolve) do nowego okna obiekt wybrany przez użytkownika.



Kontroler edycji encji działa w podobny sposób co kontroler dodawania z tymże wywołuje inną akcję serwisu.



Przykładowy widok edycji:



* 1. Usuwanie

Za usuwanie danej encji odpowiada metoda delete(). Otwiera ona nowe okno potwierdzające chęć usunięcia, do którego przekazywana (tak jak w przypadku edycji poprzez obiekt resolve) jest encja wybrana przez użytkownika. Tak jak w przypadku akcji dodawania i edytowania po zwróceniu kontroli przez nowe okno lista encji jest odświeżana.



Kontroler usuwanie encji posiada 2 metody confirm() która potwierdza usuwanie i poprzez serwis wysyła żadanie do serwera oraz metodę decline() która zamyka okno bez podejmowania jakiejkolwiek akcji.



Widok okna potwierdzania chęci usunięcia:



1. PopUp

Kontroler Popup oprócz podstawowych operacji CRUD posiada metodę activatePopUp() która jest odpowiedzialna za realizację założenia zgodnie z którym tylko jeden popup może być aktywny.



1. Statystki

Moduł statystyk ze względu na swoją specyfikę jest zorganizowany w odmienny sposób.

Użytkownik wybierając (za pomocą menu) statystyki do wyświetlenia wywołuje odpowiednią akcję która podmienia kod HTML. Wszystkie akcje kontrolera StatisticsController za pomocą serwisu pobierają odpowiednie strony w postaci kodu HTML i przypisują je do zmiennej pageHtml



Zmienna pageHtml w widoku jest wykorzystywana przez dyrektywę bind-html-compile (patrz xxx) aby podmieniać i jednocześnie kompilować wewnętrzy kod HTML danego elementu



Każda ze stron zawierająca statystki korzysta z zewnętrznego modułu angular-charts (patrz xxx ) służącego to rysowania diagramów przy pomocy Tagu <canvas>. Przykładowa strona statystyk:



1. Animacja ładowania

Animacja ładowania danych jest realizowana za pomocą tzw ?interceptora? . Przechwytuje on każde żądanie realizowane za pomocą obiektu $http i odpowiednio pokazuje lub chowa animację ładowania.



Sama animacja jest realizowana z pomocą dyrektywy LoaderDirective zaprezentowanej poniżej:



1. Edycja tekstu

Edycja tekstu została zrealizowana za pomocą zewnętrznego modułu text-angular (patrz xxx).

Aby używać wyżej wymienionego modułu po zaimportowaniu go wystarczy opatrzyć odpowiednią kontrolkę atrybutem text-angular.



1. Wstawki

Wstawki ze względu na to iż mogą zostać dodawane tylko w tekście zostały dodane do modułu text-angular za pomocą udostępnianego poprzez niego API.



Sam system wstawek (patrz xxx) w warstwie prezentacji został zaimplementowany za pomocą kontrolera i tzw pickerów. Kontroler kieruje przepływem informacji pomiędzy konkretnymi pickerami a serwisem który waliduje wstawkę. Każdy z pickerów posiada swój widok z formularzem który pozwala wygenerować odpowiednią wstawkę.

InsetController posiada metody pozwalające na zmianę typu wstawek (poprzez podmianę formularz )oraz zamianę konkretnych danych otrzymanych z pickera na odpowiadający im Tag wstawki.



Każdy z pickerów został zaimplementowany w podobny sposób. Jest to dyrektywa która na odpowiednie zdarzenie (np. kliknięcie w kontrolkę) otwiera nowe okno które umożliwia wybranie odpowiednich elementów do wstawki np. zdjęcia czy strona. Następnie po potwierdzeniu dane są zwracane do kotrolera wstawek.

