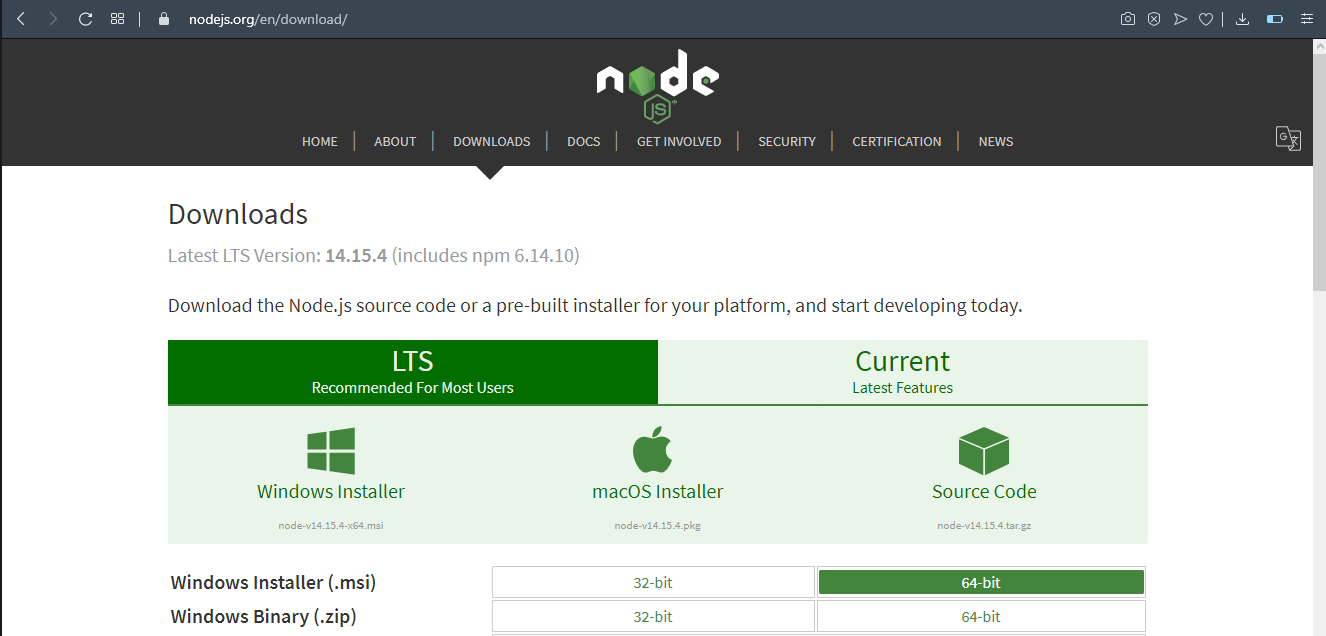
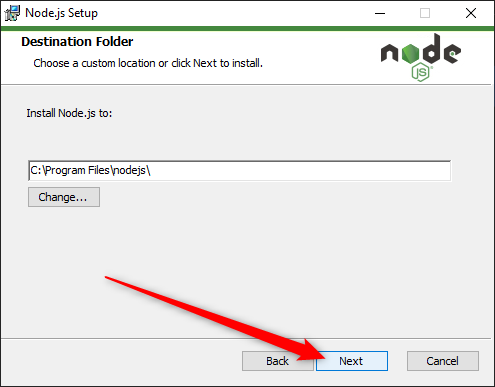
Podczas pracy z Angular warto zainstalować Framework node.js . Można go pobrać na różne platformy z:

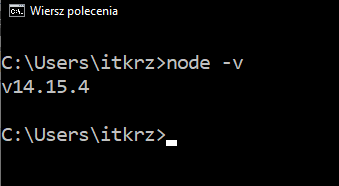
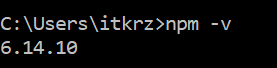


Framework ten jest wbudowany w przeglądarkę Chrome. Jego zadaniem jest wspomaganie programowania w JS.

Instalujemy pobrany plik

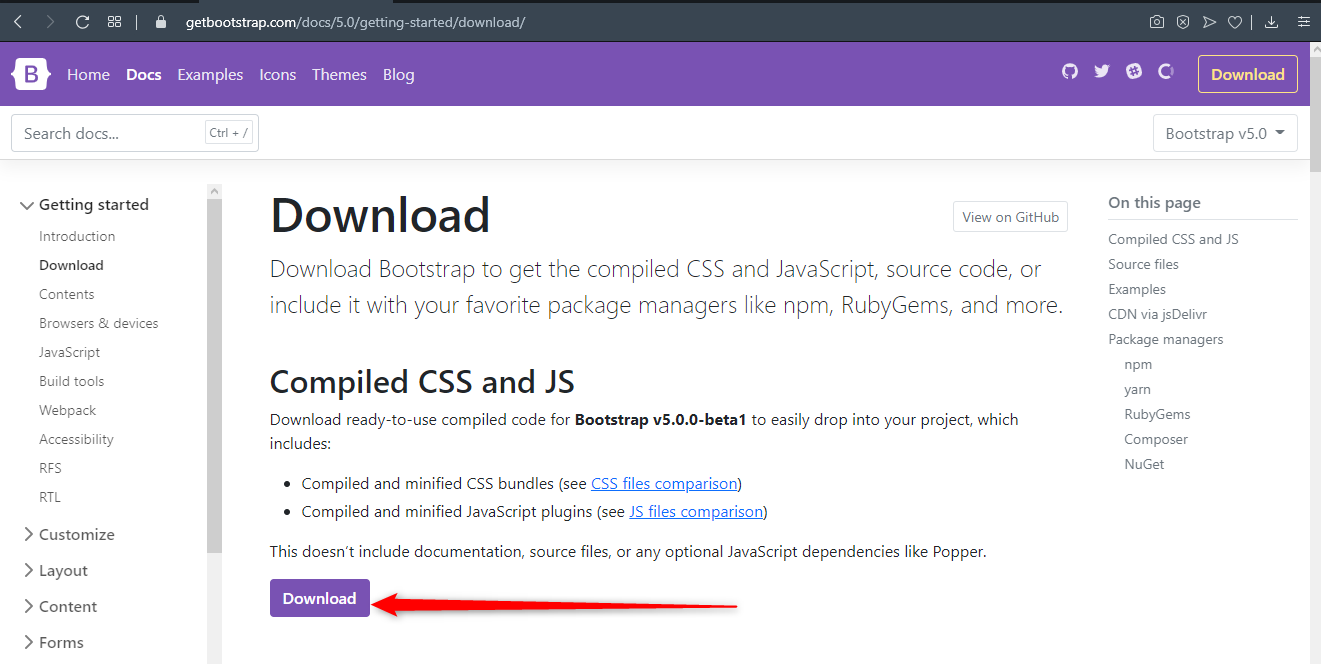


Teraz, aby przekonać się o prawidłowej instalacji trzeba w wierszu poleceń wpisać polecenia:

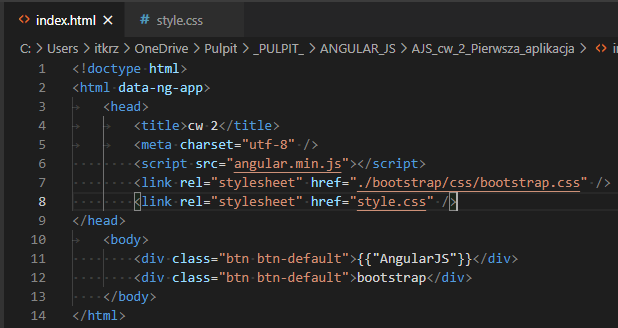
 

Drugim ważnym elementem na który warto zwrócić uwagę to Framework Bootstrap CSS

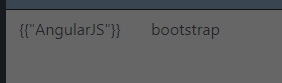
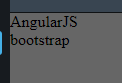
Pobieramy go ze strony:



Po pobraniu i rozpakowaniu umieszczamy oba foldery (CSS, JS) w naszym katalogu w nowoutworzonym podkatalogu o nazwie ‘bootstrap’. Dodatkowo podłączamy style pliku: bootstrap.css do naszego HTMLa. Aby się przekonać czy bootstrap i AngularJS działają wystarczy napisać ‘apkę’: (ważna jjest kolejność lonijek)



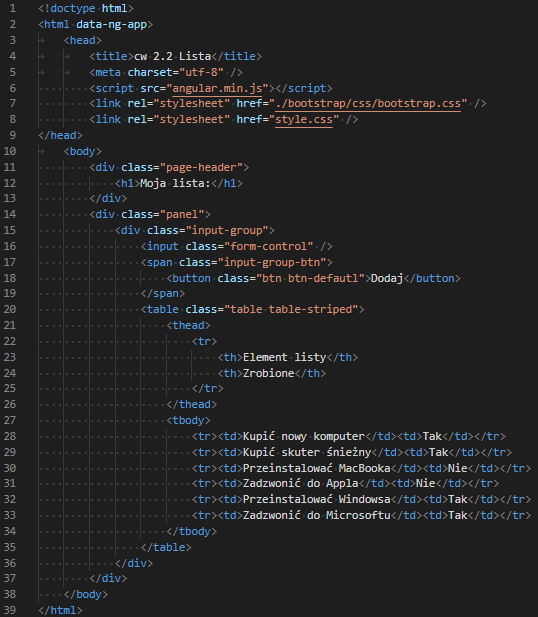
Jeżeli Angular lub bootstrap nie działają to na stronie się zobaczymy efekt:

Brak AngularJS:  ;brak bootstrap: 

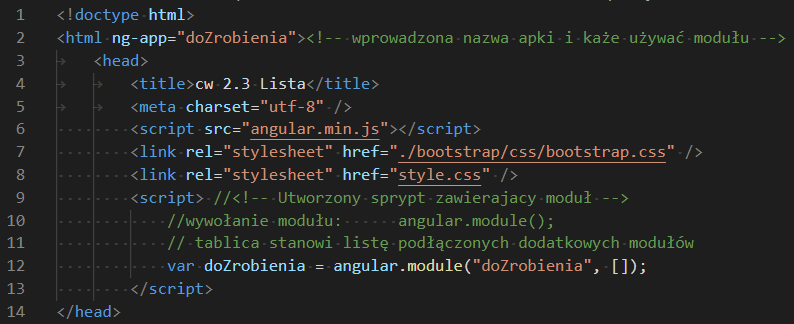
# 

# PIERWSZA APLIKACJA

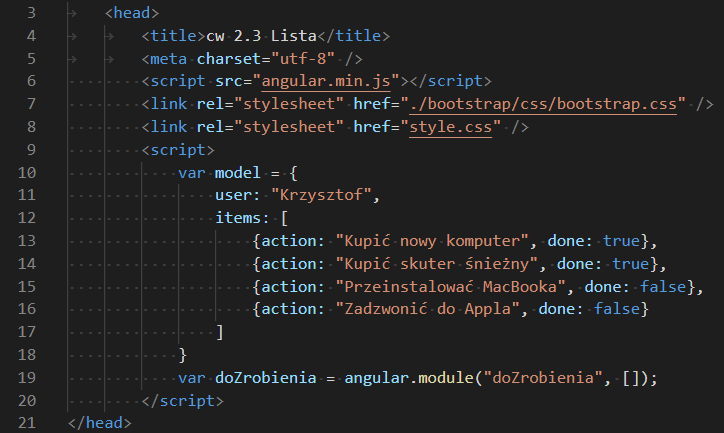
Przygotowanie projektu korzystającego z klas bootstrap.css:



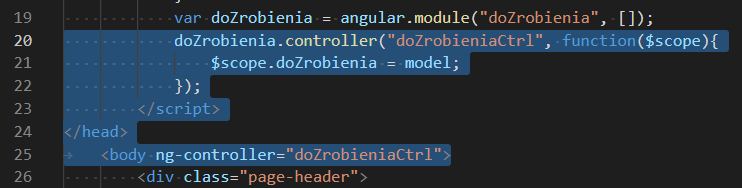
Zmiany w headerze dokumentu tworzące odwołanie do modułu oraz utworzenie modułu o nazwie ‘**doZrobienia**’:



Utworzenie modelu danych (obiektu) – warto go wstawić przed modułem doZrobienia:

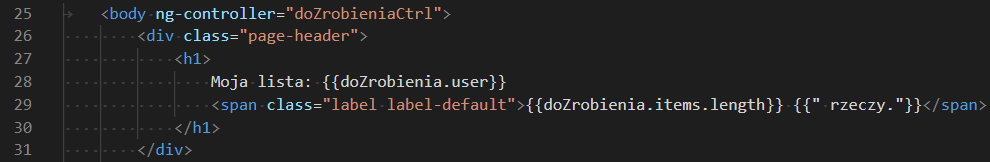


Kolejnym krokiem jest utworzenie kontrolera i dodanie go do obsługi ciała dokumentu:

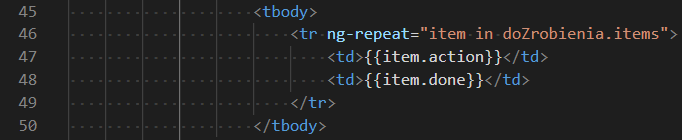


Kolejnym krokiem jest utworzenie widoku:

(wszystkie zmiany będą robione w <body> -> h1, tbody)

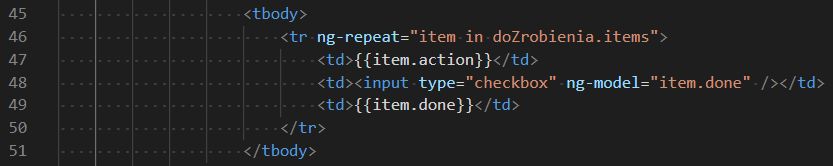


Teraz modyfikujemy listę w tabeli aby była automatycznie uzupełniana na podstawie modelu danych



## Interakcja z użytkownikiem

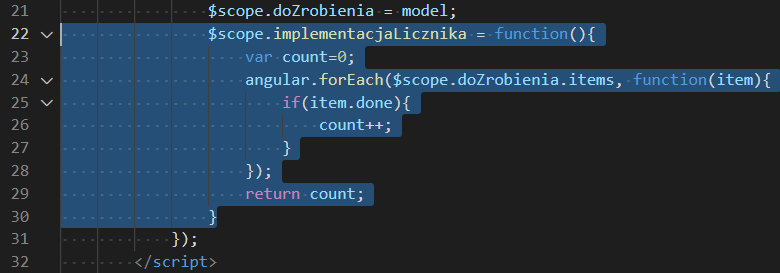
Dotychczasowa postać aplikacji przedstawia sam jej wygląd, ale jeszcze nie obsługuje interakcji z użytkownikiem. Należy więc zastosować tzw. Dwukierunkowe zachowanie dołączonego modelu. W tm celu można użyć ‘checkboxy’ dopisując komórkę w tabeli.



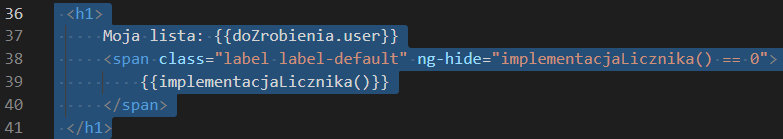
(w tym momencie wyświetlane wartości ‘true’ oraz ‘false’ są opcjonalne i nie muszą być wyświetlane, ale warto zaobserwować ich zachowanie na stronie podczas obsługi checkboxa). Można w kodzie usunąć linię 49.



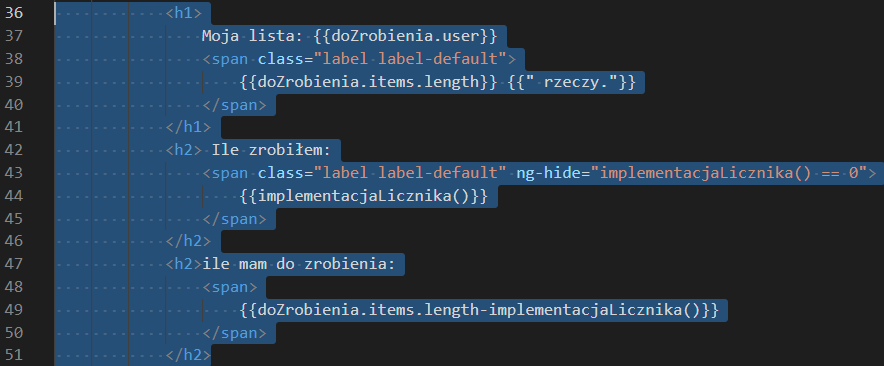
Teraz w celu zautomatyzowania możliwości wprowadzania danych można rozbudować kontroler o funkcję. Funkcje mogą kontrolować zachowanie modelu danych.



Teraz można użyć powyższą funkcję do wyświetlania ilości rzeczy do zrobienia. Licznik będzie zmieniać wartość w zależności od zaznaczonych checkbox-ów. W tym celu przerabiamy znacznik <span> znajdujący się w znaczniku <h1>. Używamy tutaj dyrektywy ukrywającej ‘**ng-hide**’ korzystającej z warunku boolowskiego:

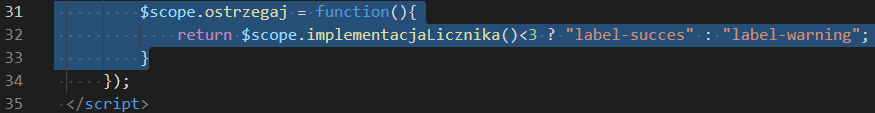


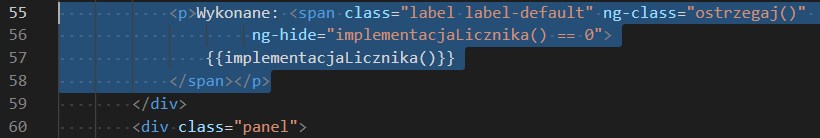
Warto połączyć powyższe rzeczy w jednej aplikacji i wyświetlić sobie pełną statystykę naszych poczynań. Przebudowujemy znacznik <h1> dopisując znaczniki niższego poziomu:



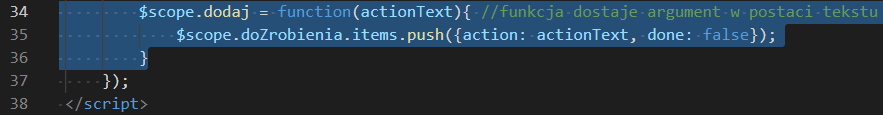
Ilość rzeczy zrobionych można zaimplementować jeszcze w inny sposób uzależniając nową funkcję od już utworzonej w kontrolerze funkcji. Nowa funkcję będzie używać warunku ‘IF’.

Rozbudowujemy kontroler oraz naszą statystykę:

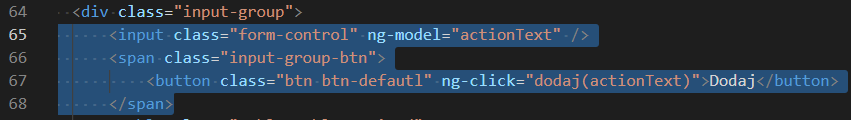




Kolejnym etapem jest wprowadzenie możliwości dodawania elementów listy do wykonania. Po raz kolejny należy rozbudować kontroler o nową funkcję.



Teraz należy przebudować formularz wprowadzania tekstu:



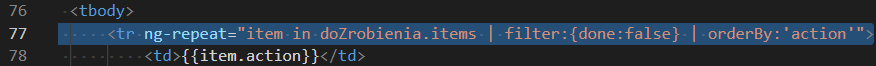
Uzyskujemy możliwość wprowadzania nowych pozycji listy:



## Filtry danych

Dane na stronie można również sortować, czyli zmieniać ich kolejność, wyświetlać tylko niektóre elementy, itp.. .

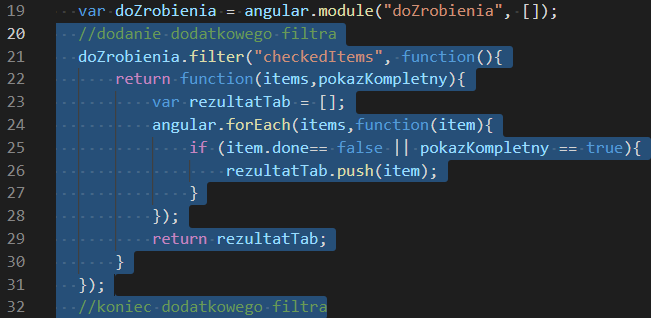
Do tej pory nowe wpisy będą się pojawiały na końcu listy. Można to zmienić. W tym celu można zmodyfikować znacznik <tr>. Jednocześnie ukryjemy rzeczy wykonane:



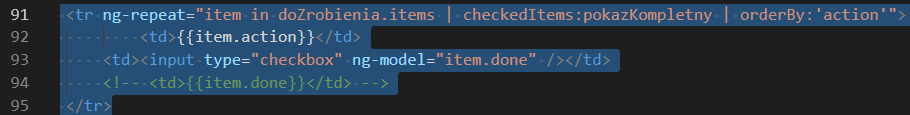
## Kolejny przydatny filtr

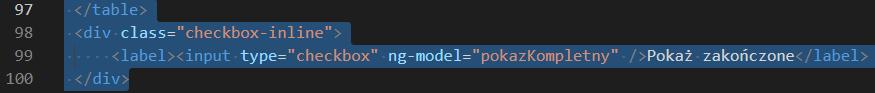
Wadą takiego rozwiązania jest to, że na stałe ukrywane są wszystkie rzeczy zaznaczone jako wykonane. Można to zmienić dokładając kolejny filtr umożliwiający za pomocą checkbox-a wyświetlenie również rzeczy wykoananych.

Tym razem przed kontrolerem do naszej aplikacji dodajemy filtr.



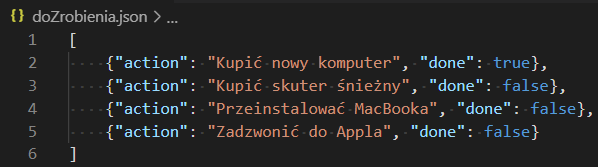
Teraz należy użyć powyższego dodatkowego filtra. W tym celu przerabiamy znacznik <tr> oraz pod tabelą dodajemy checkbox-a :





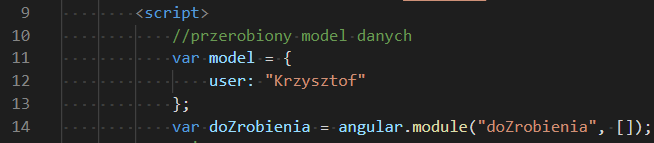
## Umieszczenie tablicy z danymi w pliku json, czyli Ajax przychodzi z pomocą

Tworzymy plik ‘doZrobienia.json’ z tablicą danych. Tablicę usuwamy z pliku html.

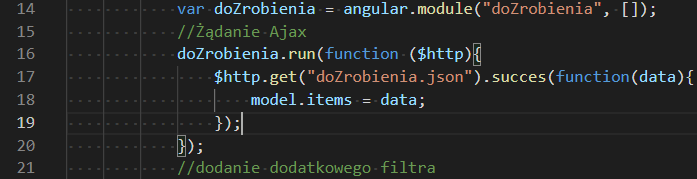


(zwróć uwagę na podobieństwo do zawartości w pliku HTML)

Teraz można przerobić model danych:



oraz dodajemy żądanie Ajax:



Jeżeli dobrnąłeś do tego miejsca i wszystko działa to BRAWO!!!