- Co to jest **Node.js** i **npm**:

To już chyba wiecie :) Ale w formie przypomnienia:

Node.js – środowisko uruchomieniowe zaprojektowane do tworzenia wysoce skalowalnych aplikacji internetowych, szczególnie serwerów www napisanych w języku JavaScript. Umożliwia tworzenie aplikacji sterowanych zdarzeniami wykorzystujących asynchroniczny system wejścia-wyjścia. Jest aplikacją open source.

npm – (node package manager) - narzędzie które będzie dla nas instalowało wszystkie potrzebne paczki.

- Co to jest **Angular**:

AngularJS – otwarty framework oparty na języku JavaScript, wspierany i firmowany przez Google, wspomagający tworzenie i rozwój aplikacji internetowych na pojedynczej stronie. Zadaniem biblioteki jest wdrożenie wzorca Model-View-Controller (MVC) do aplikacji internetowych, aby ułatwić ich rozwój i testowanie.

Biblioteka wczytuje plik HTML zawierający dodatkowe specyficzne dla tej biblioteki tagi. Podążając za instrukcjami wydawanymi przez owe znaczniki, biblioteka przypisuje wejściowe i wyjściowe elementy strony do modelu, zapisanego jako zestaw zmiennych języka JavaScript. Wartości tych zmiennych można ustawić ręcznie lub pobrać z otrzymywanego statycznie lub dynamicznie źródła JSON-a.

AngularJS został stworzony z przekonaniem, że programowanie deklaratywne powinno być używane do budowy interfejsów i łączenia komponentów oprogramowania, podczas gdy programowanie imperatywne znajduje zastosowanie w logice biznesowej[3]. Framework przystosowuje i rozszerza możliwości tradycyjnego HTML-a do lepszej obsługi dynamicznych treści, co umożliwia automatyczną synchronizację pomiędzy modelem i widokiem. W ten sposób Angular ogranicza manipulacje w DOMie i ułatwia testowanie.

Cele twórców:

Oddzielenie manipulacji w DOMie od logiki aplikacji. Ułatwia to testowanie kodu.
Ukazanie testowania kodu jako tak samo ważnego, jak pisanie kodu. Testowanie kodu jest dramatycznie ograniczane przez jego złą strukturę.
Oddzielenie warstwy klienckiej aplikacji od warstwy serwerowej. Umożliwia to pracę równoczesną i ułatwia pracę zespołową.
Pokazanie twórcom oprogramowania całej ścieżki: od projektowania interfejsu, poprzez pisanie logiki biznesowej, aż do testowania.
Angular korzysta z wzorca MVC i promuje utrzymywanie słabych zależności (loose coupling) pomiędzy warstwą logiki, prezentacji i danych. Angular wprowadza tradycyjne serwerowe technologie, takie jak kontrolery zależne od widoku do aplikacji klienckich. W wyniku takiego

działania, można w łatwy sposób uprościć back-endową część aplikacji, odchudzając cały projekt.

Dwukierunkowe wiązanie danych (two-way binding):

Dwukierunkowe wiązanie danych w AngularJS jest jego najważniejszą funkcją, która redukuje ilość kodu napisanego w trakcie uwalniania backendu serwera z odpowiedzialności za szablony. Szablony są stworzone w prostym HTMLu zgodnie z danymi zawartymi w zakresie (scope) zdefiniowanym przez model. Serwis $scope w Angular wyłapuje zmiany w modelu i modyfikuje HTML w widoku poprzez kontroler. Podobnie, wszelkie zmiany w widoku widać w modelu. To pozwala ominąć potrzebę aktywnego manipulowania DOMu i ułatwia samodzielne i szybkie tworzenie aplikacji internetowych. Angular wyłapuje zmiany w modelach przez porównanie wartości z wartościami zgromadzonymi we wcześniejszym procesie dirty-checking; w przeciwieństwie do Ember.js i Backbone.js, które czekają na zdarzenia zmiany wartości modelu.

Co to jest ***module***:
Co to jest ***directive****:*

Dyrektywa ng-click:
ng-click to dyrektywa która pozwala nam powiązać kliknięcie przez użytkownika z wywołaniem funkcji lub wyrażenia.

Dyrektywa ng-init:
Służy do przetwarzania wyrażeń w bieżącym zakresie

Dyrektywa ng-show i ng-hide:

Dyrektywa ng-if:

Dyrektywa ng-repeat:

Dyrektywa ng-class:
ng-class to dyrektywa która pozwala nam powiązać klasy elementu z modelem. Przyjmuje mapę nazw klas i wyrażeń, które muszą być spełnione, żeby dana klasa była dodana do elementu.

Dyrektywa ng-style:

Dyrektywa ng-cloak:

Jeżli w kodzie strony HTML zostaną zastosowane szablony Angulara, to podczas wyżwietlania strony w przeglądarce — tuż przed wczytaniem AngularJS i skompilowaniem kodu — na ekranie przez chwilę będą widoczne podwójne nawiasy klamrowe. Aby tego uniknąć, można tymczasowo ukryć szablon, aż do chwili jego
pełnego przetworzenia.
W Angularze służy do tego dyrektywa ng-cloak . Definiuje ona dla wczytywanego elementu dodatkowę reguły w postaci display: none !important; .

Aby mieć pewność, że podczas wczytywania strony nic nie będzie migać, należy pamiętać o załadowaniu Angulara w nagłówku (<head>) strony HTML.

Więcej o dyrektywach znajdziesz: https://docs.angularjs.org/api/ng/directive

Co to jest ***controller***:
Moduł aplikacji Angulara wykorzystywany do pisania logiki biznesowej aplikacji. Dane kontrolera są związane z konkretnym modelem. Kontroler jest inicjowany przez wbudowaną dyrektywę ng-controller
Co to jest ***service/factory***:

- Co to jest **bower**:

Bower jest menedżerem paczek klienckich i umożliwia nam pobieranie takich paczek jak jQuery, angular, itp.

By móc z niego korzystać musimy jednak go zainstalować:

npm install bower -g

lub dodać odpowiednia wersje w package.json

od tej pory, do naszej dyspozycji będzie dostępna komenda bower

Podobnie jak z npm mamy do dyspozycji klika komend – dosłownie, te podstawowe w ogóle się od siebie nie różnią :)

bower init

stworzy nam plik bower.json w którym możemy opisać naszą paczkę, jeżeli będziemy chcieli ją publikować jak i da nam możliwość przechowywania zależności (od jakich paczek nasz projekt jest uzależniony) – nie jest to wymagane, ale jest przydatne. Tak naprawdę możemy to co instalujemy wrzucić do repozytorium, ale trzeba zwrócić uwagę na plik .gitignore bo często się zdarza, paczki bower zawierają nazwy plików lub katalogów, które są przez GIT ignorowane.

bower install <package name>

zainstaluje nam package name paczkę w katalogu bower_components (wyszukać paczkę możemy przez stronę jak i za pomocą komendy search). Dodatkowo w bower istnieje możliwość konfiguracji gdzie będą pobierane nasze zależności
Wystarczy utworzyć plik: .bowerrc i wpisać:

```
{
   "directory": "sciezka/do/katalogu"
}
```

Teraz, jeżeli instalujemy paczkę za pomocą install, trafi ona do katalogu który określiliśmy.

Podobnie jak w npm, jeżeli chcemy zaktualizować plik bower.json by zawierał on zależności to:

bower install <package name> --save

bower install <package name> --save-dev

Ponownie, --save wtedy kiedy jest to niezbędne do uruchomienia aplikacji, --save-dev jeżeli do jej rozwoju.

Jeżeli nasz plik bower.json już zawiera odpowiednie wpisy to możemy wykonać polecenie

bower install

tak jak w npm spowoduje to zainstalowanie wszystkich zależności z uwzględnieniem *.bowerrc.*

Aby zaktualizować paczki używamy polecenia:

bower update

Aby w szybki sposób odnaleźć najnowsza wersje paczki, użyj polecenia:

bower info <package name> np. bower info lodash

Pamiętaj, że po instalacji pakietów musisz je dodać do głównego katalogu aplikacji (w naszym wypadku index.html)

- Co to jest **less**:

To dynamiczne arkusze stylów.

LESS rozszerza CSS o elementy dynamicznych języków, takie jak zmienne, domieszki (mixins), operacje i funkcje. LESS działa zarówno w przeglądarkach (Chrome, Safari, Firefox) jak i po stronie serwera, w środowiskach Node.js i Rhino.

Aby skompilować lessy użyj polecenia:
grunt less

więcej informacji o less'ie : http://ciembor.github.io/lesscss.org/

- Co to jest **bootstrap**:

Został stworzony przez programistów Twittera, można by z kolei określić jako zestaw klas CSS, dzięki którym możliwe jest budowanie w pełni „pływających" interfejsów użytkownika, nie martwiąc się o różnego rodzaju haki w celu dostosowania wyglądu strony pod różne przeglądarki.