Angular.JS

Which architectures exist for web applications ?

Threre are 2 main arch:

1. Server-side-rendering
2. Client-side rendering

Rendering – which side makes the html files, the html will finally look like a pattern and the content will be produced in runtime.

For ex:

We have a list:

<table>

<tr>

<td>….</td>  
<td>….</td>  
<td>….</td>

</tr>

</table>  
  
the contents of the list comes from an object array, creating html tags in JS is problematic, we want the html to have the structure of our page.  
when we talk about rendering it mean that in the html there is a code that in runtime the real html will be created.  
the html code that was described above it can be create either in option 1 or 2.

Framework for 1 – rails for ruby, Django (python), asp.net(.NET), jsp (Java) or spring, php, erb

Framework for 2 – AngularJS (build on javascript, by google), react.js (facebook), vue (similar to angularjs)  
\* Angular 2 is a failure

In client-side rendering it is easier to make a Single Page Application, which in user experience it feels more like a desktop application (like gmail, google docs etc.)  
the Angular framework beside the render on the client, makes the SPA much easier to create, maintain and change the url.

SEO – how the site fits the search engines.

Angular works in MVC or MVVM

MVC(Model, View, Controller) – the **view** reads data from the **model**, the model doesn’t know the view, the **controller (business logic)** knows the view and the model.

Ex: a button was clicked on the view (like adding data) the controller sends the message to the model, to update, the view updates because the model has changed and it watches it.

MVVM (Model , ViewModel, View) – the business logic is in the **model** the model is larger than the model in MVC, the **view model** has view logic and all event are connected to it, in contrary to MVC the VM doesn’t know what to do with the event only to pass it on and handle any view concerning code.

AngularJS characteristics:

SoC: Separation of Concerns, separation between the BL and the presentation

DRY: Don’t Repeat Yourself – use of patterns to reduce the use of HTML, use of directives (an element of ui that can be used repeatedly – ex navigation bar)

AngularJS components:

Model / Service – our data (on the side of the model, BL)

View – the presentation, html files in form of templates

Data Binding – connects the view and the view-model ($scope) in angular

Controller – connects the view and the data

Router – routes between the views, gets the url and returns the view and the controller that knows to present the view.

Directive – UI component that can be reused in all html files.

MVC Components connects : Model (App) -> Routes ->View ->Directives

->Controller ->Services  
View <- Scope -> Controller

for ex:  
dashboard - dashboard.html

dashboard.js

the view only presents data on the scope, and the controller updates the scope, in every view can be 0 or more directives and in parallel the Services(data model and BL) is connected to the controller, and can be shared among controllers.