What Is Angular?

AngularJS is a structural framework for dynamic web apps. It lets you use HTML as your template language and lets you extend HTML's syntax to express your application's components clearly and succinctly. Angular's data binding and dependency injection eliminate much of the code you would otherwise have to write. And it all happens within the browser, making it an ideal partner with any server technology.

[Template](https://docs.angularjs.org/guide/concepts#template) : HTML with additional markup

[Directives](https://docs.angularjs.org/guide/concepts" \l "directive): extend HTML with custom attributes and elements

[Model](https://docs.angularjs.org/guide/concepts#model): the data shown to the user in the view and with which the user interacts

[Scope](https://docs.angularjs.org/guide/concepts#scope): context where the model is stored so that controllers, directives and expressions can access it

[Expressions](https://docs.angularjs.org/guide/concepts#expression): access variables and functions from the scope

[Compiler](https://docs.angularjs.org/guide/concepts#compiler): parses the template and instantiates directives and expressions

[Filter](https://docs.angularjs.org/guide/concepts#filter) : formats the value of an expression for display to the user

[View](https://docs.angularjs.org/guide/concepts#view): what the user sees (the DOM)

[Data Binding](https://docs.angularjs.org/guide/concepts#databinding): sync data between the model and the view

[Controller](https://docs.angularjs.org/guide/concepts#controller): the business logic behind views

[Dependency Injection](https://docs.angularjs.org/guide/concepts#di): Creates and wires objects and functions

[Injector](https://docs.angularjs.org/guide/concepts#injector): dependency injection container

[Module](https://docs.angularjs.org/guide/concepts#module): a container for the different parts of an app including controllers, services, filters, directives which configures the Injector

[Service](https://docs.angularjs.org/guide/concepts#service): reusable business logic independent of views