**Nimbus Architecture**

Providing the ability to build application through configuration.

Below is the brief descriptions on few of the technologies – (Spring Boot and Angular) used in Nimbus Architecture.

**Spring Boot**

**What is Spring Boot?**

Spring Boot makes it easy to create stand-alone, production-grade Spring-based Applications that you can run. We take an opinionated view of the Spring platform and third-party libraries, so that you can get started with minimum fuss. Most Spring Boot applications need very little Spring configuration.

**Key Features:**

* Takes away boilerplate configuration
* Embedded Tomcat or Jetty
* Starter dependencies
* Automatic configuration …creates beans as needed ...resolves dependencies (CLI) ...adds imports (CLI)
* No code generation / no XML configuration
* Absolutely no code generation and no requirement for XML configuration

**Why Spring Boot?**

* Enter the microservice arena the easy way
* To ease the applications building process.
* To reduce Development, Unit Test and Integration Test time by providing some defaults.
* To increase Productivity.
* It is very easy to integrate Spring Boot Application with its Spring Ecosystem like Spring JDBC, Spring ORM, Spring Data, and Spring Security etc.

**Spring Boot Components:**

* Spring Boot Starter

Spring Boot Starters are a set of convenient dependency descriptors that you can include in your application. You get all the Spring and related technology at one-stop-shop that you need without having to deal with sample code and copy paste loads of dependency descriptors. It minimizes the effort.

For example, to work with Spring & JPA for database access project include only the spring-boot-starter-data-jpa dependency in your project POM.

* Spring Boot Actuator

Spring boot actuator is one of the important features of Spring boot. It is used to access current state of running application in production environment. There are various metrics which you can use to check current state of the application.

Spring boot actuator provides restful web services end points which you can simply use and check various metrics. For example, /metrics end point will display metrics such as free memory, processors, uptime and other properties,

* Spring Boot CLI

The Spring Boot CLI is a Command Line Interface for Spring Boot. It can be used for a quick start with Spring. It can run Groovy scripts which means that a developer need not write boilerplate code; all that is needed is focus on business logic. Spring Boot CLI is the fastest way to create a Spring-based application.

* Spring Boot Initializer

Spring Boot Initializer is a Spring Boot tool to bootstrap Spring Boot or Spring Applications easily

**Spring Boot Disadvantages:**

It is very tough and time consuming process to convert existing or legacy Spring Framework projects into Spring Boot Applications. It is applicable only for brand new Spring Projects.

**Angular**

**What** **is Angular?**

Angular is a JavaScript based open-source framework for building client-side web applications. Angular is a TypeScript-based open-source front-end web application platform.

**Key Features**

1. Cross Platform
   * Progressive web apps

It uses modern web platform capabilities to deliver an app-like experience. It gives high performance, offline, and zero-step installation. So, working with Angular is pretty much easy.

* + Native

Can build native mobile apps

* + Desktop

Create desktop-installed apps across Mac, Windows, and Linux using the same Angular methods you’ve learned for the web plus.

1. Speed and Performance
   * Code generation

Angular turns your templates into code that’s highly optimized for JavaScript virtual machines, giving you all the benefits of hand-written code with the productivity of a framework.

* + Universal

You can use any technology with Angular for serving the application like node.js, .NET, PHP and other servers.

* + Code splitting

Angular apps load quickly with the new Component Router, which delivers automatic code-splitting, so users only load code required to render the view they request.

1. Productivity
   * Templates

Quickly create UI views with simple and powerful template syntax.

* + Angular CLI

Command line tools: You can easily and quickly start building components, adding components, testing them, and then, instantly deploy them using Angular CLI.

* + IDEs

Get intelligent code completion, instant errors, and other feedback in popular editors and IDEs like Microsoft’s VS Code.

* + Testing

With Karma for unit tests, you can identify your mistake on the fly and Protractor makes your scenario tests run faster and in a stable manner.

## **Building Blocks of Angular**

**Angular Blocks**

* **Modules**
  + Angular apps are modular and to maintain modularity, we have Angular modules or you can say NgModules. Every Angular app contains at least one Angular module, i.e. the root module. Generally, it is named as AppModule. The root module can be the only module in a small application. While most of the apps have multiple modules. You can say, a module is a cohesive block of code with a related set of capabilities which have a specific application domain or a workflow. Any angular module is a class with @NgModule decorator.
    - ***Decorators***are functions that modify JavaScript classes. Decorators are basically used for attaching metadata to classes so that, it knows the configuration of those classes and how they should work.
* **Components**
  + A component controls one or more section on the screen called a view. Inside the component, you define a component’s application logic i.e. how does it support the view—inside a class. The class interacts with the view through an API of properties and methods.
* **Templates**
  + You associate component’s view with its companion **template**. A template is nothing but a form of HTML tags that tells Angular about how to render the component.
* **Data binding**
  + Angular supports **data binding**, a mechanism for coordinating parts of a template with parts of a component. In two-way binding, a data property value flows to the input box from the component as with property binding. The user’s changes also flow back to the component, resetting the property to the latest value, as with event binding.
* **Directives**
  + Angular templates are dynamic. When Angular renders them, it transforms the DOM according to the instructions given by **directives**. A directive is a class with a @Directive decorator. A component is a directive-with-a-template; a @Component decorator is actually a @Directive decorator extended with template-oriented features.
* **Services**
  + Service is a broad category encompassing any value, function, or feature that your application needs. A service is typically a class with a well-defined purpose. Anything can be a service.
* **Dependency injection**
  + Dependency injection is a way to supply a new instance of a class with the fully-formed dependencies it requires. Most dependencies are services. Angular uses dependency injection to provide new components with the services they need.