Take ways from Spring config:

* @Component (@Service, @Service and @Repository) are used to auto-detect and auto-configure beans using classpath scanning. There's an implicit one-to-one mapping between the annotated class and the bean (i.e. one bean per class). Control of wiring is quite limited with this approach, since it's purely declarative.
* @Bean is used to explicitly declare a single bean, rather than letting Spring do it automatically as above. It decouples the declaration of the bean from the class definition, and lets you create and configure beans exactly how you choose.
* The Controller classes have the @Controller annotation
  + Inside the controller class, you inject the Component by using the ApplicationContext’s getBean method.
  + If the Setter dependency injection is used, make sure to add @Autowire method to the setter method on the bean
  + If different profiles are used, add the @Profile(“something”) annotation to the Controller, and use “spring.profies.active” property to set the active profile.
  + By using the profile, the database used for test and dev
* Spring has capability to scan packages for components and so you have to be cautious of which packages it scans at startup. When it scans those packages, it will look for @Compoent, @Service etc. annotation.