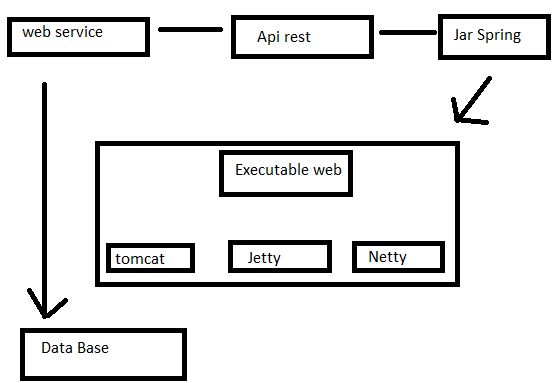
Apuntes Springboot Curso YouTube

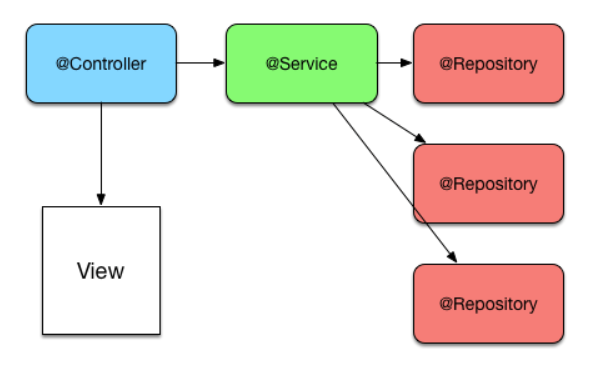
Spring (Proyectos principales)

1. Spring boot
2. Spring Framework
3. Spring Data
4. Spring Security
5. Spring Cloud
6. Spring Cloud Data Flow
7. Spring Batch

Arquitectura de SpringBoot



* Inyección de Dependencias:
* Es el responsable de crear objetos,configurarlos y ensamblarlos ( **Es una gran Factoría )**
* A los objetos manejados por la Factoría se les denomina beans
* La interface ApplicationContext es el responsable de construir la factoría a partir de metadatos mediante ….
  + Anotaciones (tendencia actual)
  + XML (<beans id=”miBean” class=”packague.miBean”)
  + Codigo de Java
* Cuando pones @Component => automaticamente pasa a ser un bean y se puede inyectar , se pueden poner alias a partir de un @Component como @Component ,@Service ,@Repository,@Controller estos son llamados Stereotypes



* Scope de Beans :
  + Singleton => una instancia de objeto
  + Prototype => mas instancias por objeto
* Se puede inyectar de 3 formas
  + Por constructor (en el constructor como antiguamente )
  + Setter de la variable
  + Variable con la anotación @Autowired
* Para inyectar se usa@Autowired , en caso tengan más de una implementación el service(estos están anotados con @service) se usa @qualifer(“bean\_que\_quieres\_usar”)
* En el Main de la clse principal , se anota con @SpringBootApplication

Persistencia .

Spring Data : proporciona sub proyectos en base a una BD en especifica relacional o no relacional , JPA Modelo de persisatencia para objetos javaen base de datos relacioneles

Jpa es un conjunto de interfaces => javax.persistence.\* , estos tienen implemtaciones como :  
hibernate , eclipseLink ,TopLink,OpenJPA

Entidades en JPA

La clase entidad representa una tabla de modelo relacional y una instancia representa una fila de la tabla , y los atributos de la clase entidad representan los campos de la tabla

Para ser una entidad de jpa debe tener requisitos :

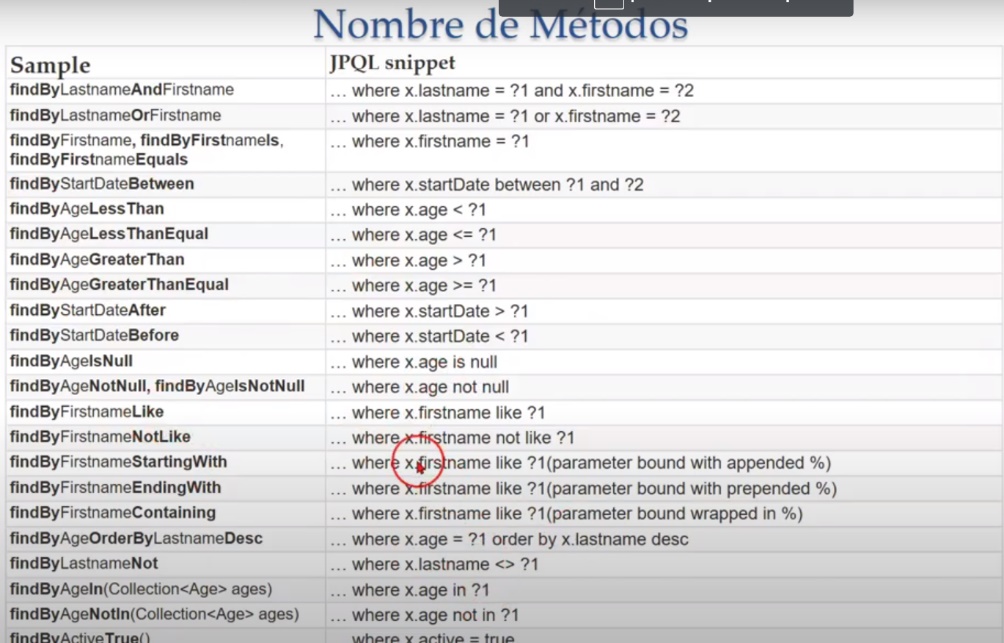
* @Entity anotado
* Debe ser publica
* Debe tener un contructor publico sin parámetros
* Debe tener una clave primaria (@Id) con sus getters y setters
* Recomienda implementar HashCode() , equals

Anotaciones recomendqadas => @GenerateValue al id ,@Column y el cambio el nombre del campo en bd , @Enumerado para valores enteros (@Enumerate(EnumType.STRING)) , @Temporal para las fechas Calendar (@TemporalType.TIMESTAMP o Date) , @Lob es de tipo gigante varchar grande

Consultas: Nombres de método => find , delete , By ,OrderBy .JPQL => SE USA cuando ya son consultas mas personalizadas , similar a SQL , maneja entidades en lugar de tablas , y atributos de objetos en lugar de campos

Obtener una lista de todos los clientes “SELECT c FFROM Cliente c”

Ejemplo de consulta =>>>@Query(“Select u.id from other\_name u where u.id> ?1 ”)  
 List<Integer> findIdByIdBetween(int initial ,int end )  
Si este fuera un delete usara un @Modifying @Query(value=”delete from ….)



BD NoSQL: MongoDB , esta orientado a documentos con un esquema libre , tiene como modelo de almacenamiento BSON mas ligero e eficiente que Json ,pero el DTO es Json

Conceptos:

* Campo: dato en concreto
* Documento: Similar al registro de una tabla, formado por un conjunto de campos
* Colección: Similar a una tabla pero sin esquema fijo .