**SPRING MVC (Modelo- Vista- Controlador)**

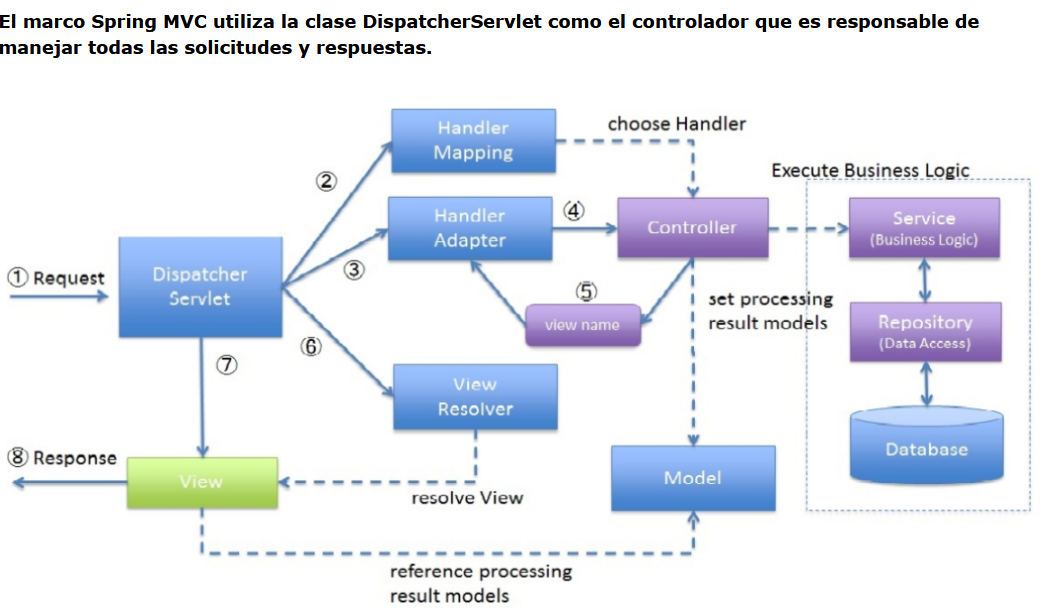
\* Separa la Lógica comercial, Lógica de presentación y la Lógica de navegación.

\* MODELO --> Se encarga de encapsular los datos de la aplicación.

\* VISTA --> Es responsable de representar los datos del modelo.

\* CONTROLADOR --> Es responsable de recibir la soliitud del usuario, construir el objeto modelo

y pasar el objeto modelo a la vista



**Flujo de ejecución del marco Spring MVC**

**­1.-** Recibir la solicitud del usuario.

2.- Elija el controlador con la ayuda del HandlerMapping

3.- El controlador procesa la solicitud llamando al método de servicio adecuado y devuelve un objeto ModeAndView al DispatchServlet que contiene los datos del modelo y el nombre de la vista.

4.- DispatchServlet envía el nombre de la vista y ViewResolve, que envía la vista real a DispatchServlet.

5.- DispatchServlet pasara los datos del modelo a la respuesta de la vista y renderiza.

**­CONTROLADOR**

**Un controlador es responsable de ejecutar la funcionalidad especifica de una solicitud.**

* **@controller** 🡪 Se usa para definir una clase como controlador Spring MVC
* **@RequestMapping 🡪** Se usa para agregar un URL de solicitud. Una URL de solicitud se puede asignar a una clase completa o a un método en particular.
* **@GetMapping 🡪** Es un atajo de @**RequestMapping(**method = RequestMethod.GET) y se usa para mapear solicitudes HTTP GET al controlador asignado.
* **@ResponseBody🡪** Vincula el valor de retorno de un método al cuerpo de la respuesta web.
  + - * + Cuando los controladores devuelvan datos como**return repository.findById()**, la respuesta se serializa en JSON antes de devolverla al cliente.
* **@ RestController** **🡪** Es una combinación de **@Controller** y **@ResponseBody** , se usa en lugar de hacer la combinación de Controller y ResponseBody.