Introduction :

**Spring Rest : Best practice**

**1st: Separation of concern : =** Controller class should call internally service class and service class should call persistence layer internally . Meaning we should not write business logic in controller class . suppose if we need to validate the customer then we should write validating method in service class and should invoke by controller to validate .

**2nd: Correct usage of HTTP method GET POST DELETE etc**

= @RequestMapping is itself by default get methhood . so after spring boot 4.3 , we do use of variant of RequestMapping : GetMapping , PostMapping , DeleteMapping or PatchMapping

Use of variant of RequestMapping is best practice

**3rd: URL naming :** In rest each resources are considered as URL . We considere each resources is considered as noun and each HTTP Method is considered as verb .

Eg . lets consider customer as one collection and order is another collection

Same url can be used for both get and post method for customer collection .

@GetMapping(“/{custID}”)

Public customer getCustomerDetail(@PathVariable int custID){ }

@PatchMapping(“/{custID}”)

Public customer updateCustomerDetail(@PathVariable int custID, @RequestBody Customer cust){ }

**4th: data Transfer** :

We should not use Persistence Entity class to carry the data from repository to controller or controller to repository .

Spring MVC automatically binds request parameter of method as argument to bean which is annotated with @RequestMapping . So it is possible to feed some unexpected fields in the arguments . As we know persistence objects are linked to underlying database by the help of persistence framework such as HiberNate , JPA or Spring DATA etc . So if we use the entity object as Argument in restMethod or RestController then it is possible to get changed the content in our expected fields in database .

**So always use Data transfer Object which transfer the data around in application .**