Extra Concepts

RestController -- RestController is **a Spring annotation that is used to build REST API in a declarative way**. RestController annotation is applied to a class to mark it as a request handler, and Spring will do the building and provide the RESTful web service at runtime.

@EnableAutoConfiguration -- The @EnableAutoConfiguration annotation **enables Spring Boot to auto-configure the application context**. Therefore, it automatically creates and registers beans based on both the included jar files in the classpath and the beans defined by us.

@Component and @ComponentScan are for different purposes. **@Component indicates that a class might be a candidate for creating a bean. It's like putting a hand up. @ComponentScan is searching packages for Components**.

ResponseEntity is used **when you need to change HTTP headers or HTTP status code based upon your business logic or incoming request**. ResponseEntity wraps the original object as its body which is optional. If you want to return an object or null, ResponseEntity will work in either way.

When Spring Boot finds an argument annotated with @Valid, **it automatically bootstraps the default JSR 380 implementation — Hibernate Validator — and validates the argument**. When the target argument fails to pass the validation, Spring Boot throws a MethodArgumentNotValidException exception.

The @RequestParam is used to extract query parameters while @PathVariable is used to extract data right from the URI.

@RequestParam is **a Spring annotation used to bind a web request parameter to a method parameter**. It has the following optional elements: defaultValue - used as a fallback when the request parameter is not provided or has an empty value. name - name of the request parameter to bind to.

BindingResult **holds the result of a validation and binding and contains errors that may have occurred**. The BindingResult must come right after the model object that is validated or else Spring fails to validate the object and throws an exception.

@Controller -- The @Controller annotation **indicates that a particular class serves the role of a controller**. This is a class level annotation.

@ResponseBody -- This is required so that our rest api return the complete response to the html or complete http response.

@RestController -- It is same as the controller but now we donot have to explicitly add the @ResponseBody annotation on each method of mapping.

The main difference between these annotations is that **@ComponentScan scans for Spring components while @EnableAutoConfiguration is used for auto-configuring beans present in the classpath in Spring Boot applications**.

The @ResponseBody annotation **tells a controller that the object returned is automatically serialized into JSON and passed back into the HttpResponse object**. When you use the @ResponseBody annotation on a method, Spring converts the return value and writes it to the HTTP response automatically.

Note -- Whenever client uses a api and sends request to the api
first layer it access is the controller layer, because controller handles all the mapping in java and provides perfect http response to the client from api(server).
Now the http response should be in the form of JSON and @ResponseBody converts our java object into json and sends that response to the client.
Clients sends request controller checks if the mapping request is matching with controller mapping methods and if it matches the implementation of the response occurs and a response is provided to the client's request.

@PathVariable -- It extracts the extra value from the given url.