Inside Spring Framework we have Spring Web module which provides basic features for building a web-application like handling HttpServletRequest and response, integrating with various other technologies, managing session etc. Using Spring-web Module, if we try to build web-application, it will be challenging as we have to write extensive boiler plate code for things like routing, request handling and response formatting.

So these problems are solved by SpringMVC module

SpringMVC module which is built on top of Spring Web module means it have basic features of Spring Web module also and some advance features also used for building web-application

Some advance features of SpringMVC module

1. Annotation-based Controllers like @Controller or @RequestMapping
2. Routing and request Mapping - You’d need to handle request routing manually through the servlet API or implement a custom dispatcher.
3. Input Binding - Features like binding HTTP request parameters to Java objects (@ModelAttribute) or validating them are unavailable without Spring MVC.
4. Integration with REST APIs:-Building RESTful APIs requires features like @RestController and built-in JSON serialization, which are absent in the core web module.

Using SpringMVC module it is easy for a programmer to build web-application because of the advance features and also it separates Data Layer(Model), Presentation Layer(View), and Business logic layer(controller) which helps in

* Loose-coupling – as each layer is separate from each layer can work independently with minimal dependency on each other. For ex- one view template can display data from multiple controllers , we can change the view(jsp to react) without changing controller, migrate to Hibernate to any other ORM in the model without affecting the controller or view.
* Maintainability – as each layer can be separately maintainable or can tested separately.

Spring MVC is a module which is built on top of Spring Web Module. It internally uses Servlet and class Annotations.

It provide separation of layers like Model, View, Controller

Model defines the data that get transfer between middleware and frontend

Controller are the classes which handles the client request, which helps in loose coupling.

View represents the frontend part which is visible to user.

@GetMapping

* It is used to map the incoming HTTP GET request to particular method.

**Model Interface**

Model interface is a holder which holds data as a key-value pair which will be sent to view page from the controller.

Method to add attributes in Model interface

addAttribute(String attributeName, Object attributeValue)

For ex - model.addAttribute("message", "Hello, World!") ;

**ModelMap class**

ModelMap class is an implementation of Model Interface that provides Map-like structure to hold attributes for sharing data between a controller and a view.

Method to add attributes in ModelMap Class

addAttribute(String attributeName, Object attributeValue)

For ex - model.addAttribute("message", "Hello, World!") ;

**ModelAndView Class**

ModelAndView class binds both model(data) and view both inside the single object. And this object is returned to the view page.

Learn difference between Model, ModelMap, ModelAndView