Modules Ioc
AOP
DAO

Date 17/08/22 Page no. 54

PROJECT 10

Java EE - Enterprise Edition.

SPRING FRAMEWORK.

What is Spring ?

- Spring is a dependency injection framework to make java application loosely coupled.
- java framework.
- ~~ioc~~ provides ioc - (inversion of control).

spring ioc container is the core of spring framework. It creates objects, configures & assembles their dependencies, manages their entire life cycle.

Light weight container, factory of all beans.

Dependency Injection

- IOC & dependency injection are used to remove dependency from the programming code.
- IOC makes the code loosely coupled.
- In Spring framework, IOC container is responsible to inject the dependency.
- We provide metadata to IOC container

* By using new keyword, object gets tightly coupled, thus we cannot make further changes.

* spring creates object by itself by dependency injection (IOC container).

Modules, Services, API

Date 29/8/22 Page no: 55

Inversion of Control

Control of object creation is being passed to Spring & spring will create object of all req depend dependencies during Runtime & inject them in program execution.

⭐ IOC container will create obj of another class.

We will provide metadata of which obj or dependency to use through xml file or annotations.

J2EE

UI LAYER — Strut / JSF → spring mvc

↓

Business / Services Layer — security logic, transaction

↓

Data Access Layer

↓

Data Base

Spring

Product Controller

↓

Product Service

↓

Product Dao.

(Product Controller, Product Service, Product Dao. — java class)

Spring JDBC

Spring ORM

internationalization
event propagation
resource loading
transparent creation of context.

Spring Modules.

1) spring core modules -
   - core, beans — IOC, Constructor injection, property injection
   - Context — inherit features from beans (IOC, factory)
   - SpEL (Spring Expression Language)
     - query & manipulate object graph at runtime.

REST VS SOAP?
REST API's?

@Controller
Rest Controller - boot.

2) AOP → Aspect Oriented Programming.
↳ allows us to define method interceptors & point cuts for decoupling.

Aspect
Instrumentation → Decoupling of code so that implementation func should be separated.
Messaging → server application
↳ messaging applications (annotations to map methods to msgs).

3) Data Integration / Access Layer Modules. (DB related)
↳ JDBC → abstract JDBC Layer
↳ ORM → integration layer to integrate ORM too
↳ JMS → java messaging services.
↳ OXM → supports objects xml mapping - castor JAX xtream.

4) Web Module - Rest API's, MVC, web oriented integration features.
↳ web
↳ Servlet
↳ Portlet
↳ Web Socket

5) Test Module - Unit testing Integration Testing
↳ Junit / Test ng.

Spring IOC Container

- component or predefined program
- objects creation, hold objects in memory, dependency injection.
- maintain obj lifecycle.
- 1) Beans (which) ?] Both info.
  2) Config file (xml) ?

APPLICATION → can use these objects

Application Context

- Interface which represents IOC Container.
- Bean Factory extended.
- sub classes
  - ClassPathXmlApplicationContext.
  - AnnotationConfig " "
  - FileSystemXml " "

a) ClassPathXml → search xml config from java class path

b) AnnotationConfig → search beans on which annotations are used.

c) FileSystemXML → search config file from file system.

In How many ways we can inject dependencies?

can be done in 2 ways -

1) Setter Injection (Property Injection)
2) Contructor Injection. * (config file ko bta denge)

Configuration Files

- spring xml file

namespace in config file.

Date 30/08/22 Page no. 56

↳ The classes which we provide to IOC Container are called Beans.

↳ Beans (XML) where we declare beans & it's dependency using tags.

↳ <beans> </beans>

↳ <bean> </bean>.

Data Types (Dependencies) to provide to IOC Container.

1) Primitive
2) Collection - list, set, map & properties.
3) Reference (user defined [objects])

STEPS TO CREATE SPRING PROJECT.

- SOFTWARES

1) ECLIPSE
2) Tomcat server
3) MySQL
4) workbench

- STEPS.

1) Create maven project
2) Add dependencies (spring core, spring context)
3) create Beans → pojo.
4) creating configuration files → config.xml
5) setting injection
6) Main class → To pull object & use.

2 ways to create spring boot project ?

1) Spring initializer
2) eclipse
3) STS
4) cmd

- If we are setting values using property tag in config file, then we are using setter injection.
  - value as element
  - value as attribute
  - value as p schema

Constructor Injection with ambiguity problems.

- By default it checks for string constructor
- To resolve ambiguity we can provide type & index property while defining beans in cfg files.

Life cycle Methods of Spring Beans

- public void init(); [initialization code, config]
- public void destroy(); [clean up code]

Configure Techniques

1) XML
2) Spring Interface
3) Annotation.

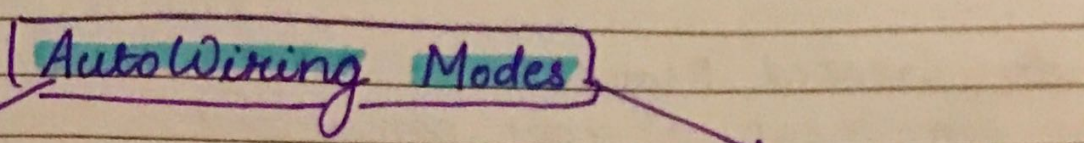
Managing Lifecycle of Beans through Annotations.

1) @PostConstruct → init
2) @PreDestroy → destroy

Autowiring in Spring

- Feature of Spring framework in which spring container inject the dependencies

automatically
- autowiring can't be used to inject primitive & string values. It works with reference only.
- Autowiring can be done in two ways.
  1) xml
  2) Annotations.



XML
- no (default)
- byName
- byType
- Constructor
- autodetect (deprecated)

Annotations
- @Autowired

Autowiring advantages
- automatic
- Less code

Autowiring disadvantages
- No control of programmer
- It can't be used for primitive & string values.

DAO VS DTO
DTO VS SERVICE
DAO VS SERVICE

Date 17/09/22 Page no. 61

ANGULAR

To create Project -
1) command - ng new (project name)
2) ~~ng s~~ cd (Project name)
3) ng s
4) code . → To open in editor

Inside Created Project
↳ app component → root component.
↳ inside component these four files are created - app.component.css, app.component.html, app.component.ts, app.component.spec.ts.
↳ app.routing.modules - we define paths or routes
↳ app.module.ts - is configuration file.
↳ app.component.ts - works as a controller.
↳ app.spec.ts - for testing.

↳ app.module.ts
↳ configuration file. (frontend)

@ ~~service~~ injectable
↳ To make service.

↳ @NgModule Decorator
↳ we pass components (mapping) @component.

app.routing.module.ts.
↳ routing file.

Q. Routing path kaise set kiya hai?
↳ app-routing.module.ts mai @NgModule({ imports: [RouterModule.forRoot(routes)],

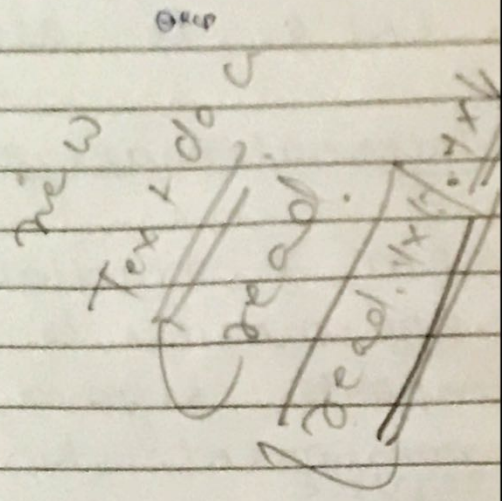
endpoints → urls of servers.

Date 21/8/22 Page no. 62

Q single Page Application ?

Ans. app-routing module.ts mai components ki routing kii hai aur app component. html mai <router-outlet></router-outlet> tag ka use kiya hai.

<router-outlet> → tag decides which components loads or which page will run at runtime.



SERVICE [BACKEND]

1.) Business operations
2.) Handling Transactions
3.) Handling Exceptions.

DAO.

1) CRUD OPERATIONS

DTO.

1) single object used to transfer data.

FORMS

1) Input Messages (Validations) for Business - Direct set kiye hai.
2) Frontend → Backend (Data) also in form of form data.

API Testing

Date / / Page no.

•> Startup class - automatically created.
Project ORS Application.
@ Spring Boot Application

We have enabled Cors to connect frontend to backend.

CORS - Cross Origin Resource Sharings
↳ Provides or has security Policy that allows URL's to access backend.

•> Internationalization (I18N)

1) npm x translate
2) app.module.ts mai translate module ko import kiya
3) component k constructor mai translate service module ko as parameter constructor pass kiya hai.
navbar Component.ts
4) assets folder mai ek I18N nam ka folder banaya aur usme do files banayi EN.JSON HI JSON.
5) pipe directive k through translate kiya hai & interpolation k through msg show kiya hai
6) method - changeLocale (Navbar Component.ts)
By Default - English.

image ka flow

Preload list search image

★ INPUT VALIDATIONS

Date 25/9/22 Page no. 64

1) LoginCtl mai baseCtl ko extend kiya hai form pe humne validations check kiye hai By using @valid, @email, @NotEmpty. usme msg pass kiye hai (form pe validations set kiye hai)

2) LoginCtl pe ORSResponse type ki login method banayi. usme as a parameter humne LoginForm aur BindingResult ka object binding result pass kiya.

3) then BaseCtl ki validate method ko call kiya then usme BindingResult ka obj pass kiya then usko ORSResp ke obj res mai hold karaya.

4) Condition check kari if(!res.isSuccess()){ return res; } ——x——

5) LoginComponent.ts mai signIn() method mai resp.result.inputerror mai msgs ko get kiya aur form.msg mai hold karaya then LoginComponent.html pe interpolation ki help se particular field pe msgs ko print kiya.

——x——x——x——x——x——

1) session ko handle kiya → JWT Token (Backend).

2) Exceptn propagate then transaction?
↳ transaction rollback.
Tx ↳ rollback
↳ commit

Q Exceptn handle hogi toh transaction
↳ transaction commit.

Q "DAO se exception kis type ki?
A DAO → Service
Exception → Flow.

Q URL se parameters ko get kaise karte hai?
@PathVariable Long userId

× Front Ctl. × ×

1) FrontCtl class banayi extend kiya hai HandlerInterceptor Adaptar class.
2) Uski do methods prehandle, post handle ko override kiya aur unme servlet HttpRequest object ka object req, Http servlet response ka object resp aur Object class ka object handler pass kiya hai.
3) then StarterClass mai addInterceptors method banayi hai usme interceptor registry ka object registry pass kiya hai.
4) Registry.Add Interceptor mai (frontCtl) ~~Interceptor~~ pass kiya then. add Path Patterns mai ("/**"). excludePathPatterns ("/Auth/**").

CORS ENABLE

1) Startup file mai (Project ORS Application) ~~M~~ WebMVCConfigurer type ki CORS-Configurer() method banayi hai, ~~usme~~
2) Usme humne WebMvcConfigurer ka obj banaya hai jisme humne addCORS.

JWT TOKEN ?? <UserForm> ?

Date 26/3/22 Page No. 66.

Mappings() method banayi hai aur usme CORSRegistry ka object registry as parameter pass kiya hai.

3) Then usme CORSRegistration ka object CORS Banaya hai then humne registry. AddMapping mai ("/**") pass kiya aur isko humne CORSRegistration ke object CORS mai hold karvaya.

4) Then humne cors.allowed Origins mai ("http://localhost:4200"); pass kiya hai.

BUSINESS VALIDATIONS

1) ~~LogisCtl mai BaseCtl ko extend kiya hai~~ LogisComponent.html mai Login aur pwd ki field mai button pe click event mai signIn() method ko call kiya hai.

2) Then LogisComponent.ts ki signIn() meth mai this.httpService.post(endpoint, formda aur callback function pass kiya hai.

[this.httpService.post(this.endpoint + "/login", this.form, function(res) {....

⟵ x BACKEND x ⟶.

3) LogisCtl ko baseCtl se extend kiya. hai aur generics mai <UserForm, UserDTO, UserServic

4) ORSResponse type ki method Logis banayi hai aur usme @RequestBody @Valid annotation pass kiye hai aur BindingRes ka object bindingResult pass kiya hai.

5) validate Method mai binding result ka object pass kiya hai aur ORSResponse ke obj res m hold kiya hai.

CSRF → CROSS SITE REQUEST FORGERY.

6) Base Service ki authenticate method ko call kiya hai usme form.getLogidId() aur form.getPassword() as parameter pass kiya hai. aur UserDTO ke obj dto mai hold karvaya hai.

7) then condition dto == null milne pe res.addMessage kiya hai ("Invalid ID aur password") aur res ko return kiya.

— x Frontend x — .

8) response.result.message iko hold karvaya humne form.message then logic Component .html pe interpolation se print karwaya

Security in Application

JWT Token
JSON Token

1) Dependency add kari hai pom.xml mai spring boot Starter security & JJWT.
2) class banayi WebSecurityConfig & annotation diya hai @EnableWebSecurity and extend kiya WebSecurityConfigurerAdapter
3) ~~its~~ passwordEncoder, AuthenticationManager and Configure ko override kiya hai
4) then configure method mai HttpSecurity ka object http Security pass kiya hai then csrf ko disable kiya hai
5) Session ko stateless kiya hai

6) Every User Request pe token ko invalidate karne ke liye filter add kiya.
7) humne ek aur class banayi JWTAuthentication Entry Point jisme humne implement kiya AuthenticationEntryPoint
8) Uske bad uski commence method ka use kiya hai & usme httpServlet request HttpServletResponse aur AuthenticationException authException pass kiya.
9) Uske bad response ki send error method mai unauthorized user hone par "Your Session has expired" msg set kiya.
10) humne JWTTokenUtil class banayi hai usme methods banayi hai get username, IssueDate, expiry date, validity methods banayi hai
11) Then dogenerate Token mai Token ko generate karke return karvaya.

1

Folder Structure -

↳ Java Resources
  ↳ src/main/java
    ↳ com.rays.
    ↳ com.rays.common
    ↳ com.rays.common.attachment
    ↳ com.rays.common.mail.
    ↳ com.rays.common.message.
    ↳ com.rays.config
    ↳ com.rays.ctl.
    ↳ com.rays.dao
    ↳ com.rays.dto
    ↳ com.rays.exception.

DAO classes ki api testing.

Date 29 9 22 Page no. 69

↳ com.rays.form
↳ com.rays.service

↳ src/main/resources
↳ application.properties.

Root → JPA objects

Libraries are index html angular

* Image Flow. (Userlist)

Two events
↳ click()
↳ change()

UserComponent.html me input type file diya h. & change event pr myfile() ko call kiya. then UserComponent.ts pr myfile() method me hmne onSubmit() method ko call kiya.

onSubmitProfile

then onSubmitProfile() method me hmne profile ~~fota~~ pick ki API ko hit kiya.

Back-End :- B.E. Pr UserCtl me ORSResponse type ki uploadPic() method bnai h. usme @pathVariable userId, Multipartfile file, httpServlet ka obj req as parameter pass kiye h.

then BaseService ki findbyId() method ko call kiya usme userId pass kiya. then UserDTO k object dto me hold kiya. then attechmentDTO ka obj bnaya aur usme multipartfile ka obj file pass kiya.

* ST_USER ⇒

firstname	ln	Pass	login	imageid
kapil	mal	123	kapil	5

Att. ⇒

id	image name
5	m.jpg

I then dto k obj se path ko set kiya, discription, & userId ko set kiya.
then hmne condition check ki Userdto.getImageId != null than doc.setId me ImageId ko pass kiya. then attachment service ki save() ko call kiya usme doc, UserContext pass kiya. then orsResponse ko obj bnaya. response.addresult me key-value pair me ImageId ko pass kiya aur return res kr diya.