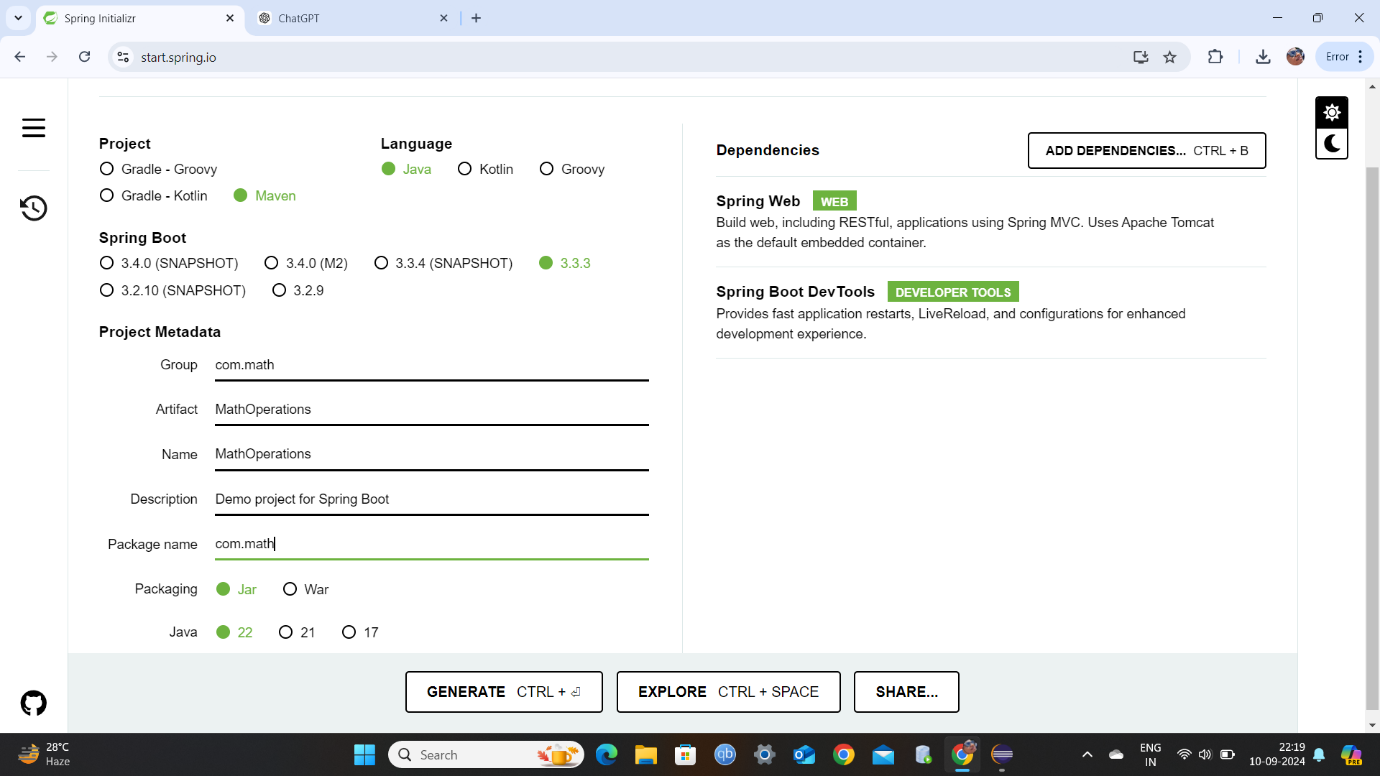
A Simple Spring Application completely written and interpreted by me



The dependencies added: -

**1)Spring Web** -> helps in building web application and handle RESTful API

What is RESTful API? ->

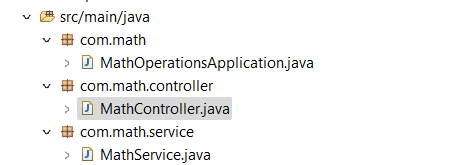
A **RESTful API** means Representational State Transfer API is a way for a software to communicate over the internet using simple HTTP requests, such as GET, POST, PUT, and DELETE. It allows applications to interact with a server (like a website or a database) to perform tasks like retrieving data or sending data. But those hardcore requests wont be used here, only GET.

**2)Spring Boot Dev Tools ->**

It was not mandatory to add but yeah after starting the server if I make changes, the changes will be automatically by this dependency in the server. Or else I had to stop the server then re-run the whole thing again.

Where these dependencies will come into play will explain that…

Project Structure after unzipping the .zip file generated by Spring Initializer



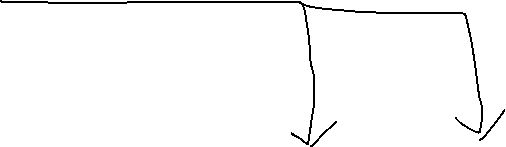


package

Com



math



MathControllerApplication.java

main class to run controller service



MathController.java MathService.java

Operations to be done

Controlled by the service class

**Service Class(name of our service class is MathService.java)**

In a typical **Maven Spring project**, a **Service Class** handles all the **business logic** of the application. It sits between the controller and repository layers (if present) and processes requests, applies logic, and prepares the data.

The **service class** is where all the computations, transformations, and decision-making happen based on what the user wants to see or do.

#1(first line of code)

package com.math.service;

(the java file is in this PACKAGE…simply class path)

#2

import org.springframework.stereotype.Service;

mainly helps to include @Service annotation in this class (will be added below)

#3

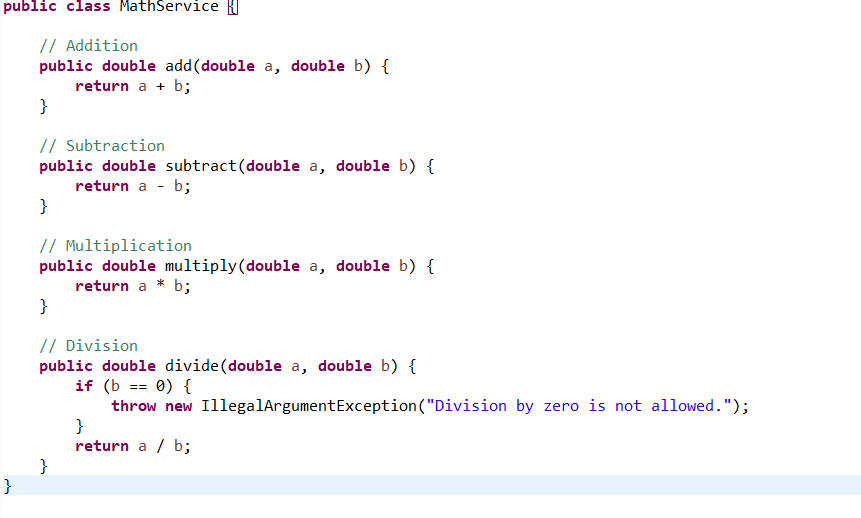
@Service

**@Service** is inbuilt module in maven used in Service type classes

*“The stuffs that have @ in the beginning are called annotation, different annotation has different jobs”*…but let’s focus on @Service annotation for now

Now @Service will tell maven that this class will be recognized as a Service class….what operations shall be done or seen are written here. Its like a tag….if it is present then this class is Service class

#rest of the code



I believe you can understand this part…no need to explain linewise..

NOW FROM MAVEN’S POINT OF VIEW

WE HAVE A PACKAGE COM.MATH.SERVICE

INSIDE THERE IS A CLASS *MathService.java* WHICH HAVE @Service ANNOTATION

WILL HANDLE THE SERVICES DONE BY THE APPLICATION

THIS *MathService.java* have 4 methods to do addition, subtraction, multiplication and division.

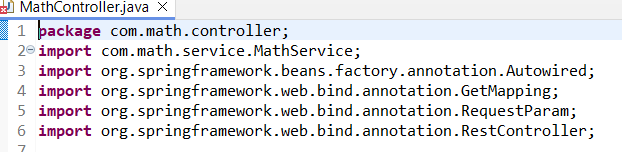
Each of the methods takes two input and returns an output.

Service Class Over.

**Controller Class (name of our controller class is MathController.java)**

Note: Here many annotations will be used.

#1-6



1. Package name….
2. We need to use the *MathService.java* class here so we are importing that class here….
3. Just like previously we called @Service with that special import statement in line #2 of *MathService.java class*

same to get @Autowire annotation this line invokes that special in-built annotation or module from maven.

The @Autowired annotation is used for **dependency injection** in Spring. It tells Spring to automatically inject an instance of a class into another class where it's needed.

For example, if you have a service class (MathService), you can inject it into a controller class using @Autowired, and Spring will handle the rest not your headache.

1. This line invokes the @GetMapping annotation from Maven and use it in this class.

This is used to handle GET …GET actually used to get data.

Why it is used …because this application will be like this: -

If I write <http://localhost:8080/add> in browser with the 2 number….it will return the addition value.

If I write <http://localhost:8080/subtract> in browser with the 2 number….it will return the subtracted value. And so on…it will take my data get to the service class with the ‘subtract’ …check for the subtract method in service class …pass the 2 numbers I gave and finally give return whatever the method returns.

1. This line invokes the @RequestParam annotation from Maven and use it in this class.

Like I said we will write two numbers in browser url…so this will allow us to do so….like just like that we can’t input something in browser url…but using this we can

<http://localhost:8080/subtract> this is normally valid url….@RequestParam is used to add values like this <http://localhost:8080/add?a=5&b=3>

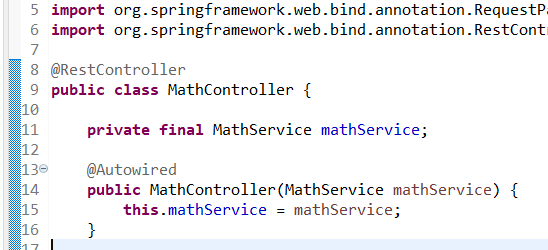
this will passed to the Service class methods…..

subtract method will return 5-3 which is 2

or else without this type of operation we don’t need it

1. This line invokes the @RestController annotation from Maven and use it in this class.

@RestController…it will tell the maven this is Controller class…just like @Service



8)@RestController

9)Class starts

11)We know MathService is a class and we cant readily treat a class like a variable …

So this is called Dependency Injection. That’s why maven

MathService class is injected into MathController class…now they are partners…..more of a business partners…. Both class will work together

Before this line u can think all those importing as formalities before this partnership starts

**Private** (to keep it private in this class only) (public bhi ho sakta tha because mini project hain with 2 classes)in big projects it is important…

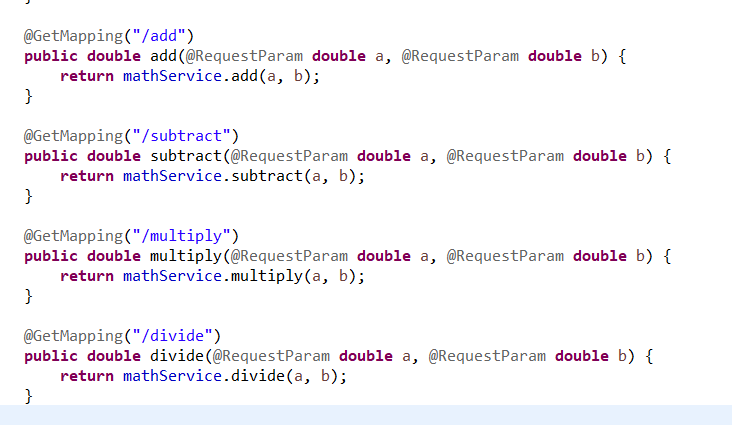
**Final** so that once initialized it wont be changed

**MathService mathService** u can consider it as ….int b….String s…..just like that

13) @Autowired

14) constructor of *MathController.java* ….dependency injection is done through constructors

MathService mathService Parameters of the constructors because they will be injected here



@GetMapping as said earlier will help to fetch methods in Service class and what will be written in the url

Will explain only one

**@GetMapping(“/add”)**

Annotation work u must know….

/add should be same as the method name in MathService there is no rule like that…it can be chayan\_add too but then the url will be

<http://localhost:8080/chayan_add>

**public double add(@RequestParam double a, @RequestParam double b) {**

firstly here the ‘add’ should be same as add method in *MathService.java*

the parameters will be passed with @RequestParam (usage discussed earlier)

like

@RequestParam double a

@RequestParam double b

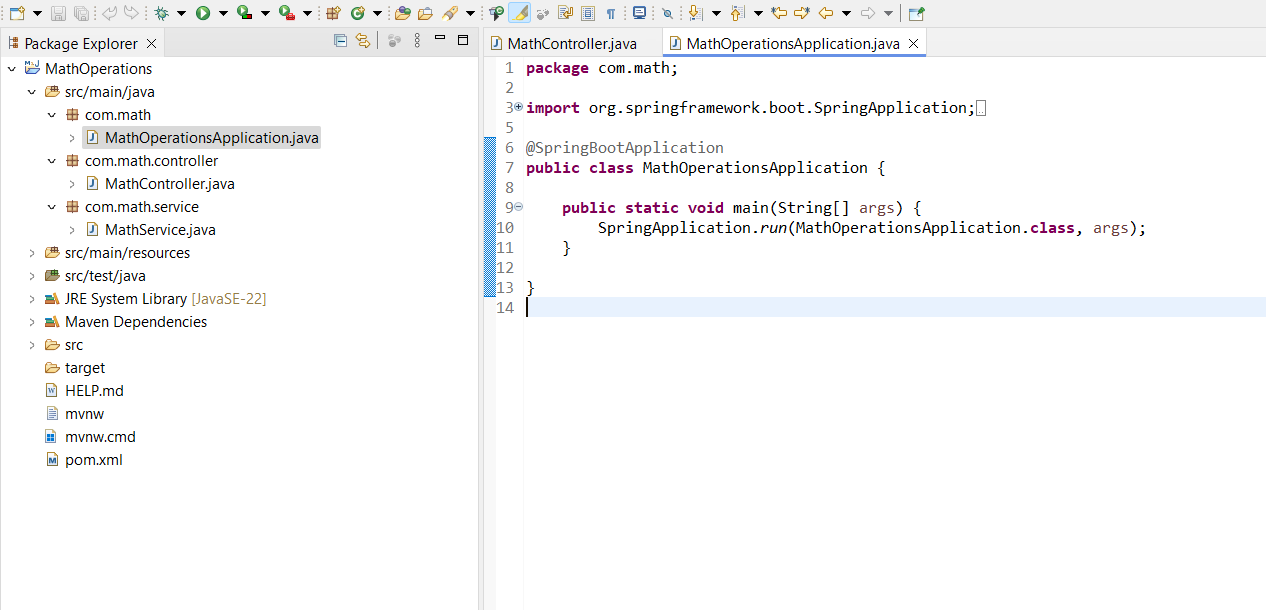
**return mathService.add(a, b);**

mathService (memorize we discussed about this in page 6) the dependency injection…a kind of variable of the whole class……

.add is the add method in *MathService.java* with parameters

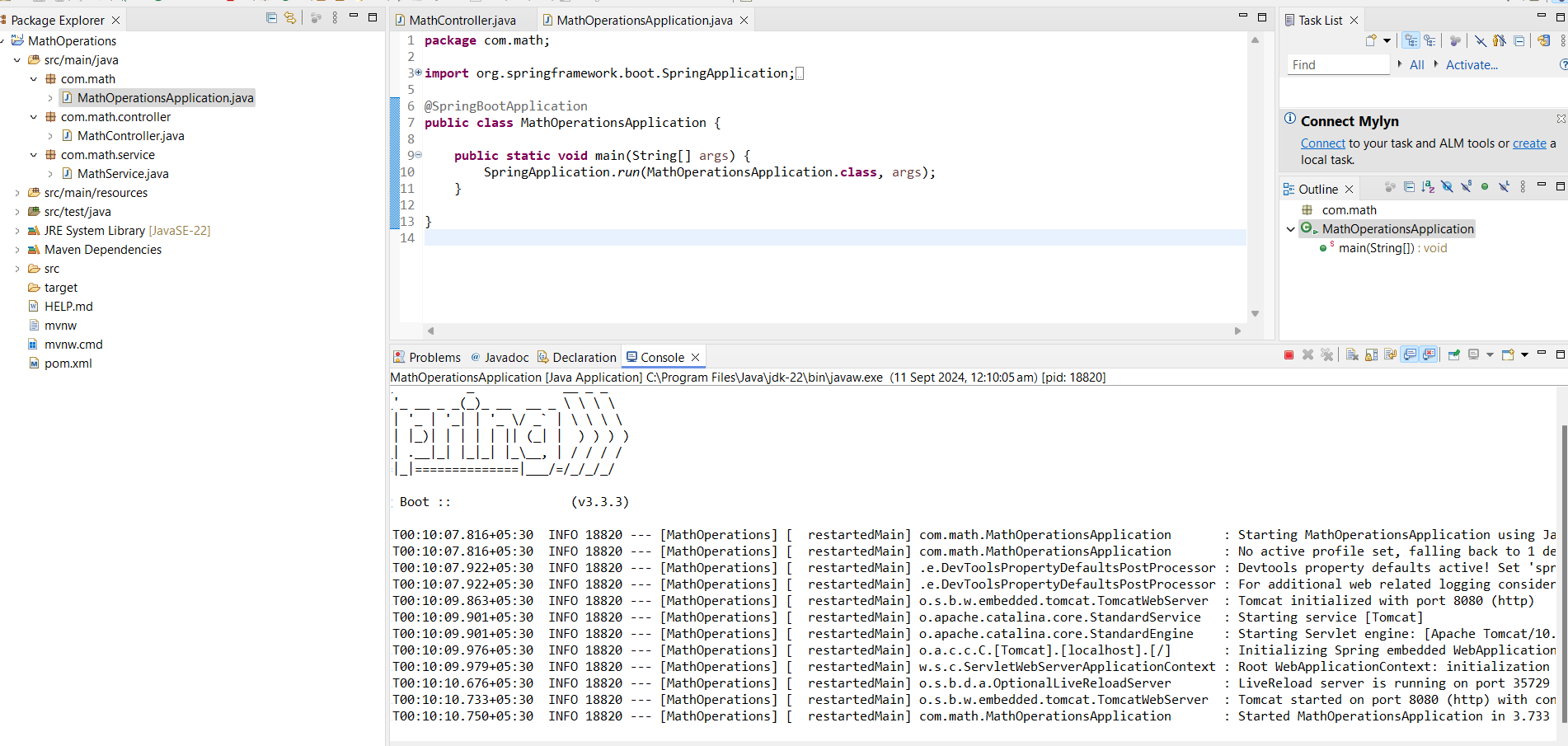
and another thing 8080 is the port number, by default it takes 8080 (if free) or else the port number will be different in your PC..where to check will say

to manually input your desired port do in application.properties(not done here)



Run this class…nothing to explain here…same for majority projects

Output:

 the port number server started too.



