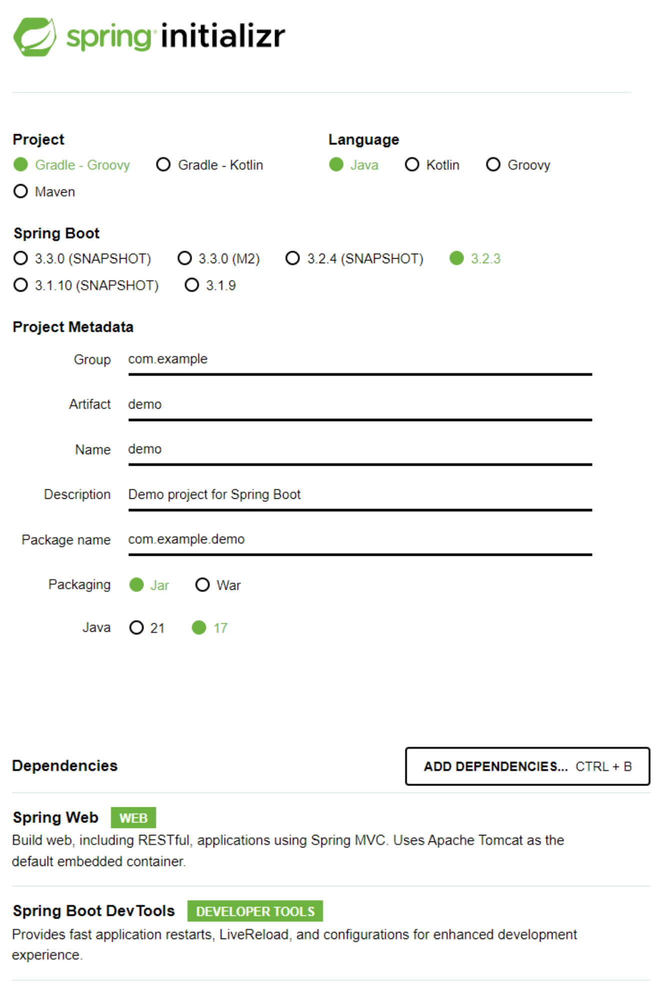
INSTRUCTIONS FOR EXECUTING THE CODE:

IDE : Visual Studio

Java version: Java 17.0.4.1

Testing: Postman

download the starter from https://start.spring.io/



spring initializr

Project
[✓ Gradle - Groovy] Gradle - Kotlin Maven

Language
[✓ Java] Kotlin Groovy

Spring Boot
3.3.0 (SNAPSHOT) 3.3.0 (M2) 3.2.4 (SNAPSHOT) [✓ 3.2.3] 3.1.10 (SNAPSHOT) 3.1.9

Project Metadata

Group	com.example
Artifact	demo
Name	demo
Description	Demo project for Spring Boot
Package name	com.example.demo
Packaging	[✓ Jar] War
Java	21 [✓ 17]

Dependencies ADD DEPENDENCIES... CTRL + B

Spring Web WEB
Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

Spring Boot DevTools DEVELOPER TOOLS
Provides fast application restarts, LiveReload, and configurations for enhanced development experience.

Choose the above specifications.

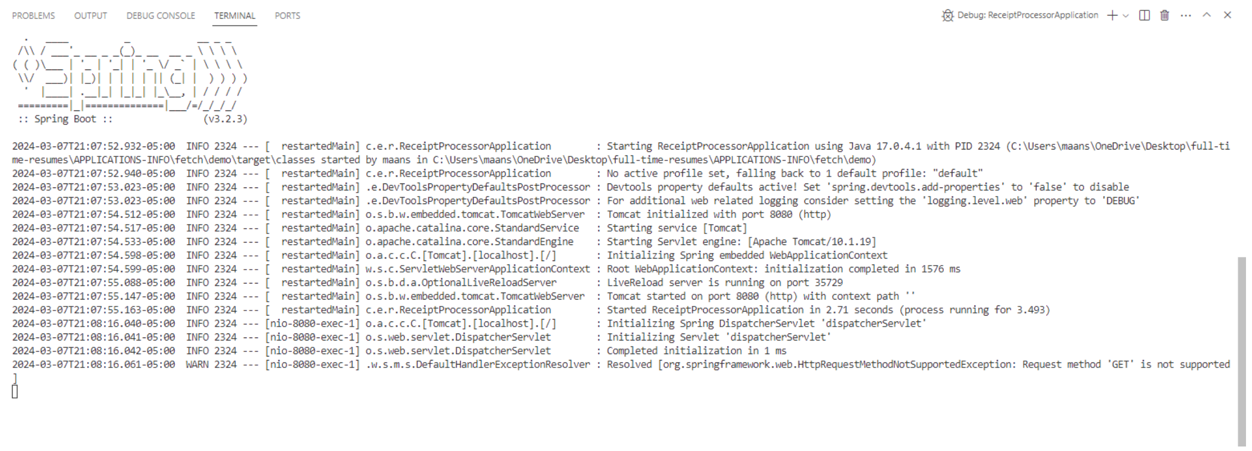
Open visual studio IDE.

Make sure JDK is already installed in your IDE.

Choose open folder and select the desired folder where you have this application saved.

Click on run from the horizontal menu and choose to start debugging.

The terminal below shows something like this.



```
  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/
 :: Spring Boot ::                (v3.2.3)

2024-03-07T21:07:52.932-05:00  INFO 2324 --- [  restartedMain] c.e.r.ReceiptProcessorApplication        : Starting ReceiptProcessorApplication using Java 17.0.4.1 with PID 2324 (C:\Users\maans\OneDrive\Desktop\full-ti
me-resumes\APPLICATIONS-INFO\fetch\demo\target\classes started by maans in C:\Users\maans\OneDrive\Desktop\full-time-resumes\APPLICATIONS-INFO\fetch\demo)
2024-03-07T21:07:52.940-05:00  INFO 2324 --- [  restartedMain] c.e.r.ReceiptProcessorApplication        : No active profile set, falling back to 1 default profile: "default"
2024-03-07T21:07:53.023-05:00  INFO 2324 --- [  restartedMain] .e.DevToolsPropertyDefaultsPostProcessor : Devtools property defaults active! Set 'spring.devtools.add-properties' to 'false' to disable
2024-03-07T21:07:53.023-05:00  INFO 2324 --- [  restartedMain] .e.DevToolsPropertyDefaultsPostProcessor : For additional web related logging consider setting the 'logging.level.web' property to 'DEBUG'
2024-03-07T21:07:54.512-05:00  INFO 2324 --- [  restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat initialized with port 8080 (http)
2024-03-07T21:07:54.517-05:00  INFO 2324 --- [  restartedMain] o.apache.catalina.core.StandardService   : Starting service [Tomcat]
2024-03-07T21:07:54.533-05:00  INFO 2324 --- [  restartedMain] o.apache.catalina.core.StandardEngine    : Starting Servlet engine: [Apache Tomcat/10.1.19]
2024-03-07T21:07:54.598-05:00  INFO 2324 --- [  restartedMain] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring embedded WebApplicationContext
2024-03-07T21:07:54.599-05:00  INFO 2324 --- [  restartedMain] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed in 1576 ms
2024-03-07T21:07:55.088-05:00  INFO 2324 --- [  restartedMain] o.s.b.d.a.OptionalLiveReloadServer       : LiveReload server is running on port 35729
2024-03-07T21:07:55.147-05:00  INFO 2324 --- [  restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port 8080 (http) with context path ''
2024-03-07T21:07:55.163-05:00  INFO 2324 --- [  restartedMain] c.e.r.ReceiptProcessorApplication        : Started ReceiptProcessorApplication in 2.71 seconds (process running for 3.493)
2024-03-07T21:08:16.040-05:00  INFO 2324 --- [nio-8080-exec-1] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring DispatcherServlet 'dispatcherServlet'
2024-03-07T21:08:16.041-05:00  INFO 2324 --- [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet        : Initializing Servlet 'dispatcherServlet'
2024-03-07T21:08:16.042-05:00  INFO 2324 --- [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet        : Completed initialization in 1 ms
2024-03-07T21:08:16.061-05:00  WARN 2324 --- [nio-8080-exec-1] .w.s.m.s.DefaultHandlerExceptionResolver : Resolved [org.springframework.web.HttpRequestMethodNotSupportedException: Request method 'GET' is not supported
]
```

To ensure smooth functionality, please install any missing plugins or extensions in Visual Studio as prompted.

Once your application is running, you can proceed to test it using Postman. Follow the steps below:

Install Postman:

If you haven't installed Postman yet, download it from the official website: Postman Downloads.

Open Postman:

Launch the Postman application on your system.

Testing Your Application:

In Postman, create a new request to test your application's API endpoints.

Enter the appropriate URL for the endpoint you want to test. Make sure to include the hostname, port, and endpoint path.

Choose the HTTP method (GET, POST, etc.) based on the functionality you want to test.

Add any necessary headers, request parameters, or request body data.

Click the "Send" button to execute the request.

Review the response returned by your application to ensure it meets the expected behavior.

By following these steps, you can effectively test your application using Postman, ensuring that it functions correctly according to your requirements.

Here is an example of how I have tested the API –

Choose Request Method:

From the dropdown menu next to the URL bar, select the appropriate HTTP method (e.g., POST).

Enter URL: Type the URL of your API endpoint into the URL bar.

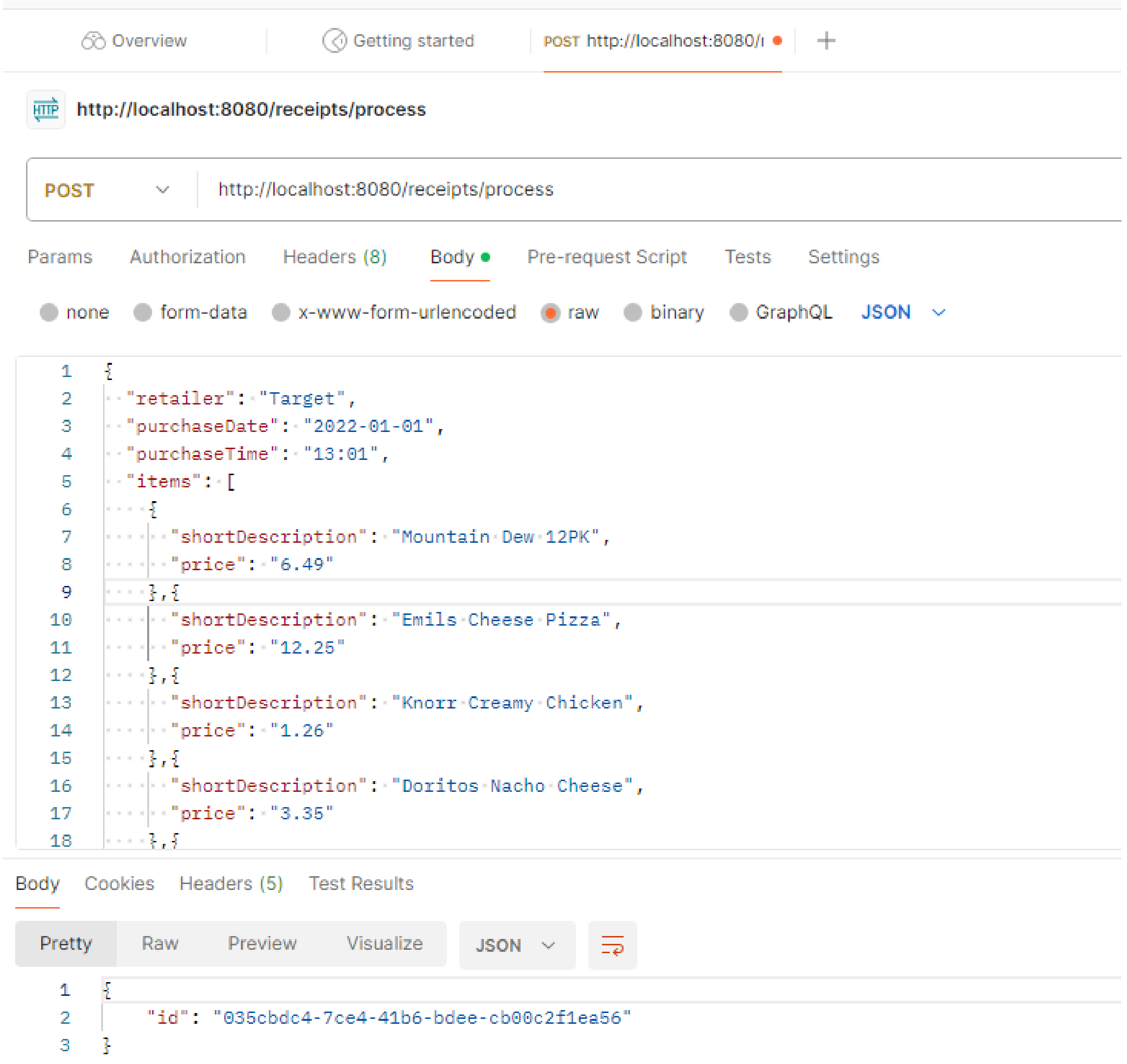
Enter Input Data:

Click on the "Body" tab located below the URL bar. Choose the "Raw" option.

Paste your input data into the text area. Ensure it is formatted correctly according to the expected data format.

Send Request: Click on the "Send" button adjacent to the URL bar to execute the request.

Review Output: The response from the server will appear in the "Response" tab below the request a



POST http://localhost:8080/receipts/process

```
{
  "retailer": "Target",
  "purchaseDate": "2022-01-01",
  "purchaseTime": "13:01",
  "items": [
    {
      "shortDescription": "Mountain Dew 12PK",
      "price": "6.49"
    },{
      "shortDescription": "Emils Cheese Pizza",
      "price": "12.25"
    },{
      "shortDescription": "Knorr Creamy Chicken",
      "price": "1.26"
    },{
      "shortDescription": "Doritos Nacho Cheese",
      "price": "3.35"
    },{
```

```
{
    "id": "035cbdc4-7ce4-41b6-bdee-cb00c2f1ea56"
}
```

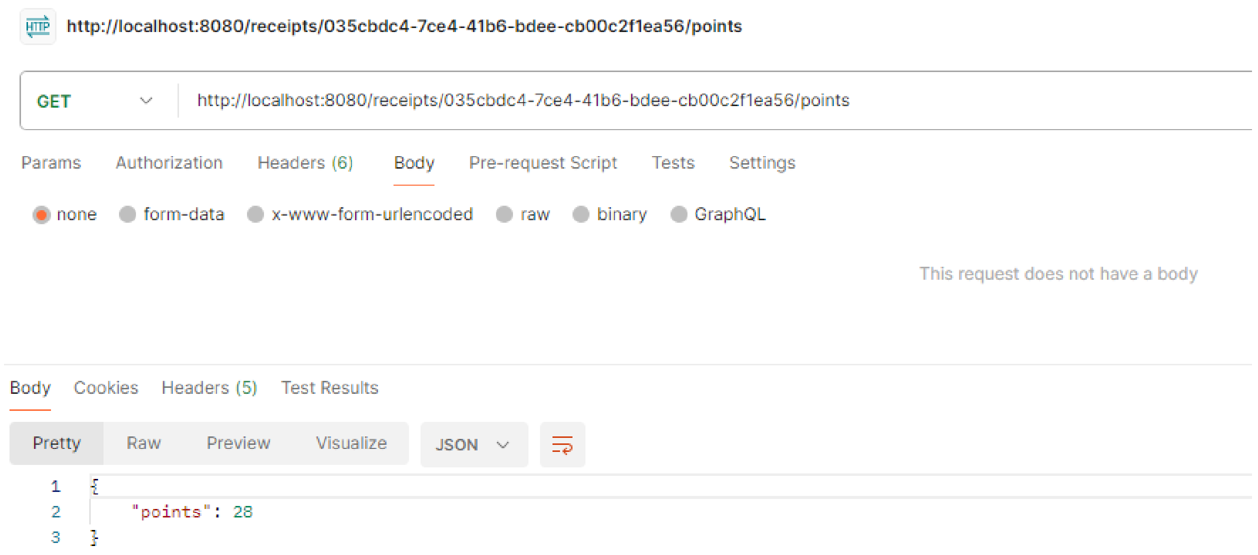
After retrieving the ID, follow these steps to check the points calculated:

Select GET Method: From the dropdown menu next to the URL bar, choose the GET method.

Enter URL with ID: Paste the URL with the retrieved ID into the URL bar.(body is not required here)

Send Request: Click on the "Send" button to execute the request.

Review Points Calculated: The response, containing the calculated points, will be displayed below in the "Response" tab.



http://localhost:8080/receipts/035cbdc4-7ce4-41b6-bdee-cb00c2f1ea56/points

GET http://localhost:8080/receipts/035cbdc4-7ce4-41b6-bdee-cb00c2f1ea56/points

Body: ● none

This request does not have a body

```
{
    "points": 28
}
```