Name : Clovis Da Silva

Project Name : Online-Bookstore with CI/CD Pipeline

Assigment: Create a complete DevOps CI/CD Pipeline with Jenkins for a Use Case (different from the Calculator application) by using GIT, Maven, Puppet/Ansible/Terraform, Docker, JUnit, Graphite, and Grafana. You can add more tools if you want.

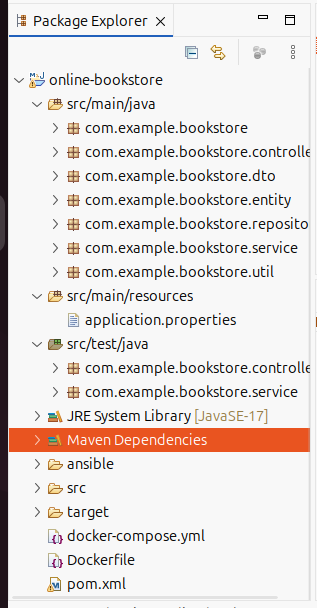
Objective: Create an Online Bookstore using Spring Boot in java and create a complete DevOps CI/CD Pipeline with Jenkins for a this Bookstore by using GIT, Maven, Ansible, Docker, JUnit, Graphite, and Grafana.

Prerequisites:

| **Tool / Technology** | **Purpose** |
| --- | --- |
| **Java (JDK)** | To build and run the Spring Boot application |
| **Maven** | To compile the code, manage dependencies, and build the JAR |
| **Spring Boot** | Framework for building REST API for Online Bookstore |
| **Docker** | To containerize all components (app, Grafana, Graphite) |
| **Docker Compose** | To orchestrate multi-container setup (bookstore, graphite, grafana) |
| **Git** | To version control the project and integrate with Jenkins |
| **Jenkins** | For CI/CD pipeline automation and rollback |
| **Ansible** | For deployment automation (with deploy.yml and rollback.yml) |
| **Grafana** | For monitoring system and visualizing metrics |
| **Graphite** | Metrics storage and forwarding for the bookstore app |
| **Web Browser** | To access the app (http://localhost:8081), Grafana (:8181), and Graphite (:8182) |

Procedure:

Create a new maven project in eclipse with Artifact ID: bookstore and Group ID: example.com



Open the root of the project and run these command to make the directory a git repository

cd /path/to/your/project

git init

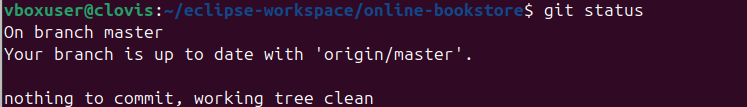
git add .

git commit -m "Initial commit"

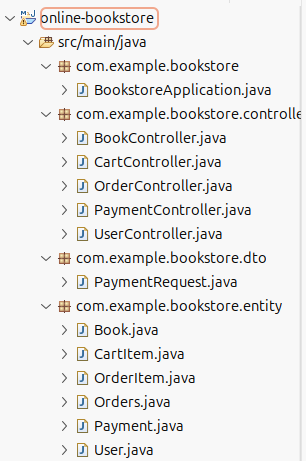
git remote add origin <https://github.com/your-username/your-repo.git>

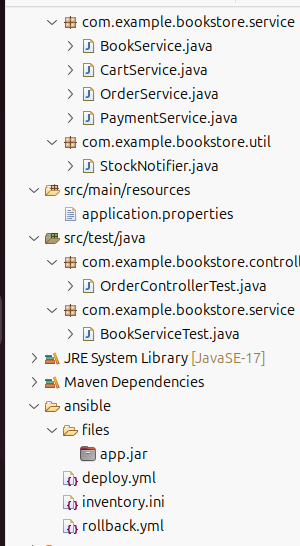
git push -u origin main

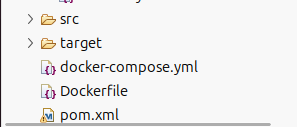
\* use git status to check if your directory is a git repository



Here is the complete structure of my application







Here is the github link to my online bookstore :  
<https://github.com/clovisdasilva11/online-bookstore>

Files in the git repository related to CI/CD pipeline

1. pom.xml : The pom.xml file is used by **Maven** to manage your Spring Boot project's dependencies, build configuration, and plugins. It ensures the application compiles, runs, and packages correctly with all required libraries.
2. Doply.yml :Ansible playbook that **builds and deploys** your Docker-based Spring Boot + Graphite + Grafana application using docker-compose.
3. Inventory.ini: Defines target **host(s)** (like localhost) where Ansible will run the deployment or rollback tasks.
4. Rollback.yml : Ansible playbook to **revert** to the **last working Docker image** (bookstore-app:backup) if the deployment fails.
5. Dockerfile :The **Dockerfile** is used to build a Docker image for your Spring Boot app. It packages the app with Java so it can run consistently in any environment. This enables easy deployment using Docker or Docker Compose.
6. Docker-compose.yml :The docker-compose.yml file defines and manages multi-container Docker applications. In your project, it sets up and runs **Graphite**, **Grafana**, and the **Bookstore** app together with their configurations, networks, ports, and dependencies—using a single command.

**Online Bookstore Application Workflow**

**1. User Interaction**

* Users browse available books via REST API.
* They can add or remove books from the shopping cart.
* An order is placed.
* The inventory stock is updated.

**2. Spring Boot Application (Backend)**

* Handles REST API endpoints (/books, /cart, /checkout, /orders).
* Manages book inventory and order processing.
* Sends low stock warnings using StockNotifier if available quantity < 5.
* Application metrics (e.g., memory, HTTP requests) are sent to Graphite for monitoring.

**4. CI/CD Pipeline**

* **GitHub**: Source code is hosted on  
  https://github.com/clovisdasilva11/online-bookstore
* **Jenkins**:
  + Fetches the latest code from GitHub.
  + Executes a shell script to:
    1. Clean up containers:  
       docker rm -f $(docker ps -aq) || true
    2. Run the Ansible deployment:

cd ansible

ansible-playbook -i inventory.ini deploy.yml --become

* **Ansible (deploy.yml)**:
  + Backs up current Docker image.
  + Stops and removes existing containers.
  + Rebuilds and starts services using Docker Compose.
  + Performs a health check on the app (http://localhost:8081/actuator/health).

**5. Docker Compose Setup**

Services involved:

* **bookstore**: Spring Boot application (port 8081)
* **graphite**: Receives metrics (ports 2003, 8182)
* **grafana**: Visualizes metrics from Graphite (port 8181)

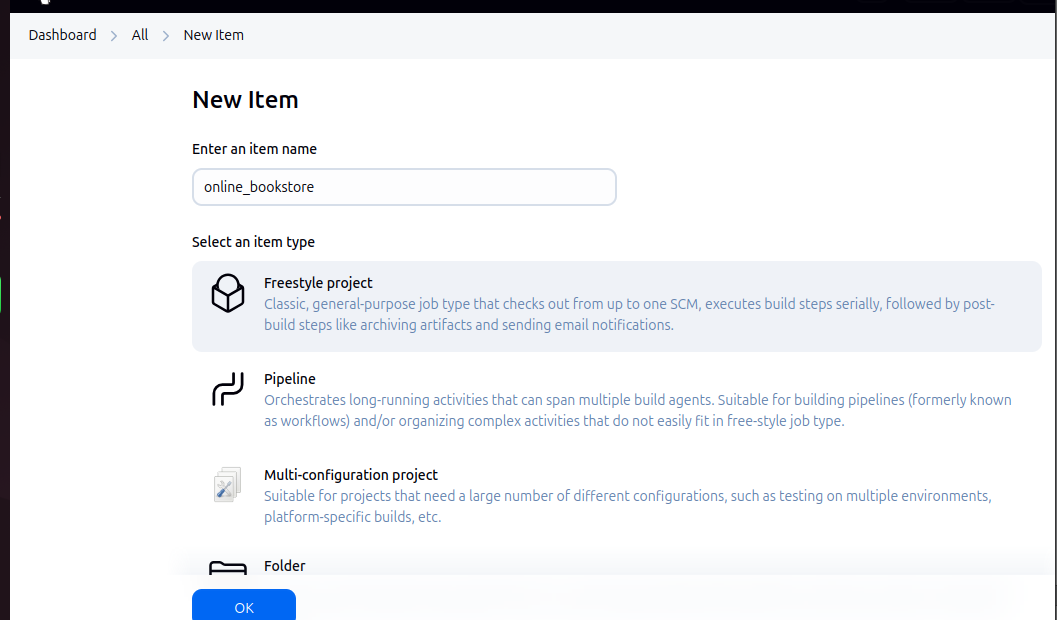
**6. Monitoring**

* Spring Boot app is integrated with Micrometer and sends metrics to Graphite.
* Grafana dashboards display:
  + API usage trends
  + System memory/cpu usage

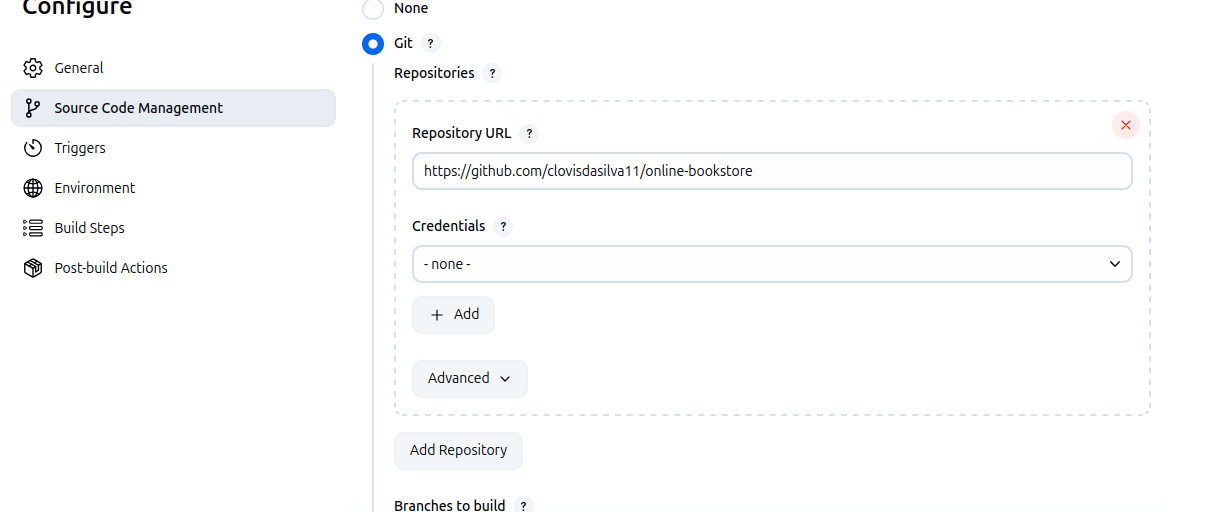
Open Jenkins at <http://localhost:8080> (or your Jenkins setup URL)

In Dashboard select New Item

Enter an item name : (eg:online-bookstore)  
select Freestyle project and click ok



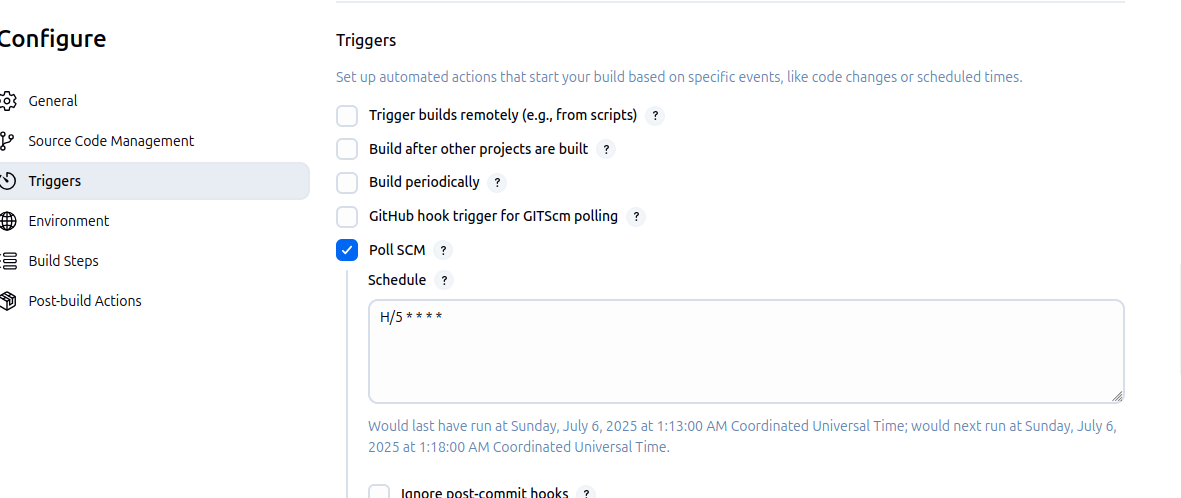
Then under Source Code Management  
 choose Git and your github repository link



Under Trigger

Choose Poll SCM to monitor for change in git repository

Add H/5 \* \* \* \* to poll every 5 minutes



Under Enviroment

Tick Delete workspace before build start

Under Build Steps

Add build step, choose Execute Shell and paste the following commands  
#!/bin/bash

echo "Killing all running containers to free up ports..."

docker kill $(docker ps -q) 2>/dev/null || true

docker rm -f $(docker ps -aq) 2>/dev/null || true

echo "Checking for any process using port 2003..."

if lsof -i :2003; then

echo "Port 2003 is in use! Manually stopping process..."

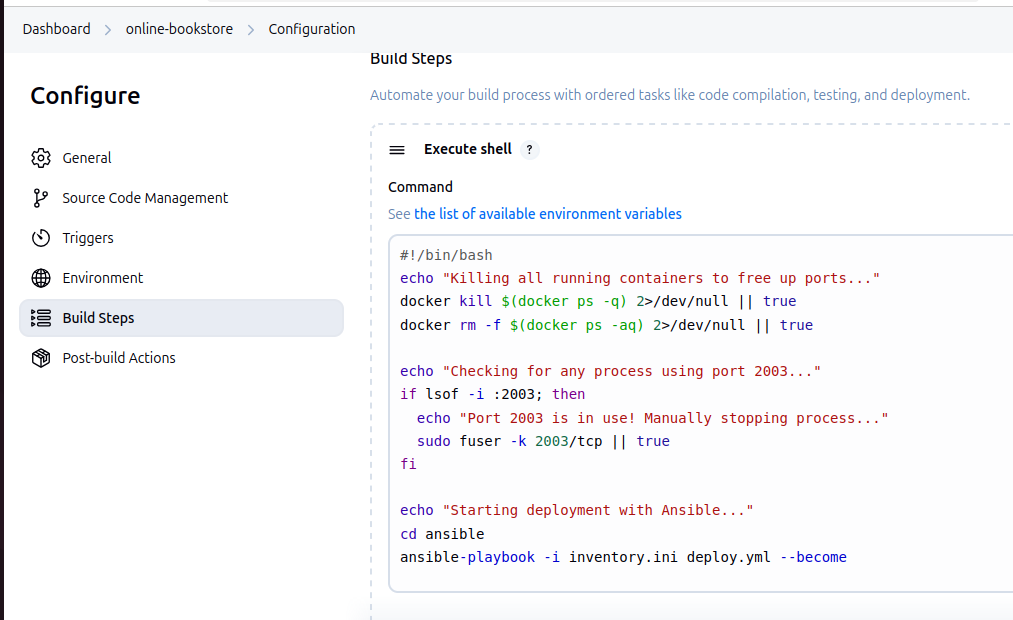
sudo fuser -k 2003/tcp || true

fi

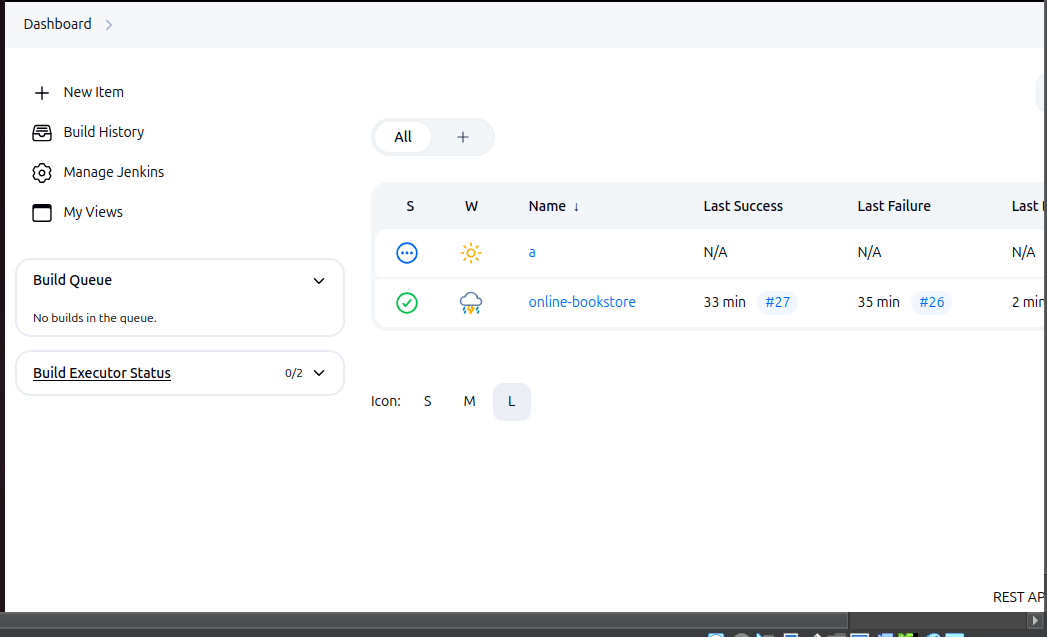
echo "Starting deployment with Ansible..."

cd ansible

ansible-playbook -i inventory.ini deploy.yml –become



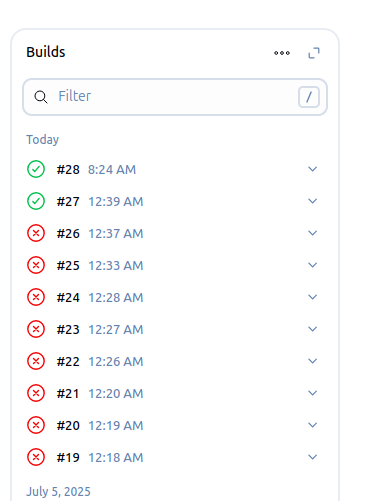
After this in the Dashboard u will see your new item (eg: online-bookstore)

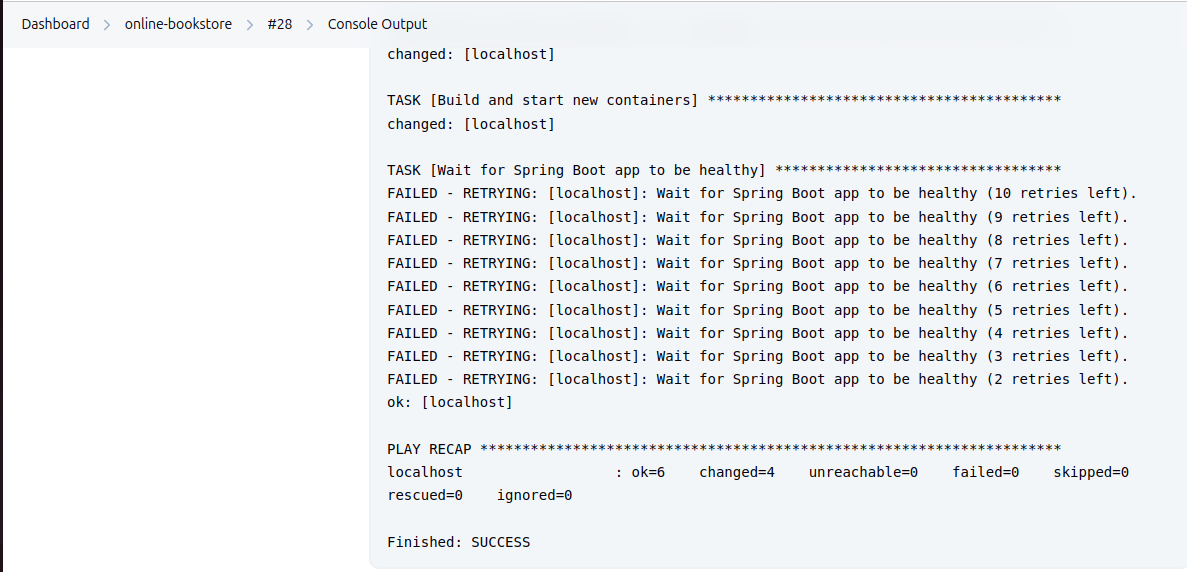


Enter your New Item(online-bookstore) and click on build now

Under Builds

It will show the build number click on the latest one and open console output

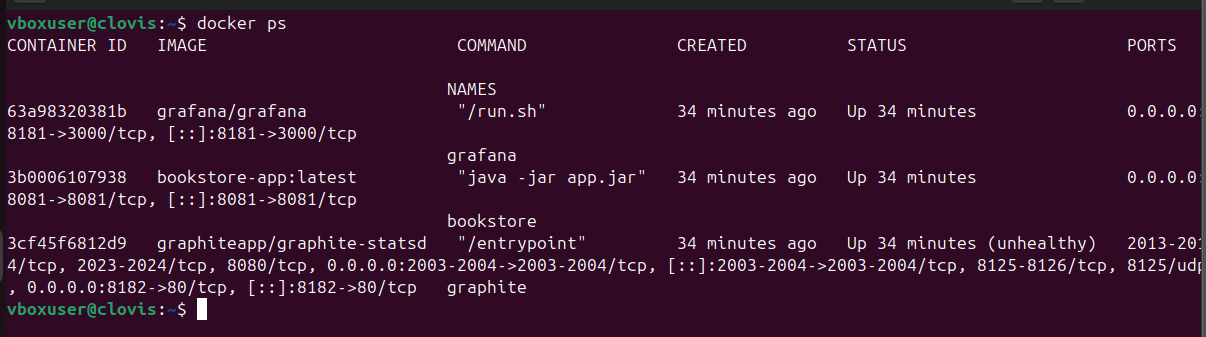


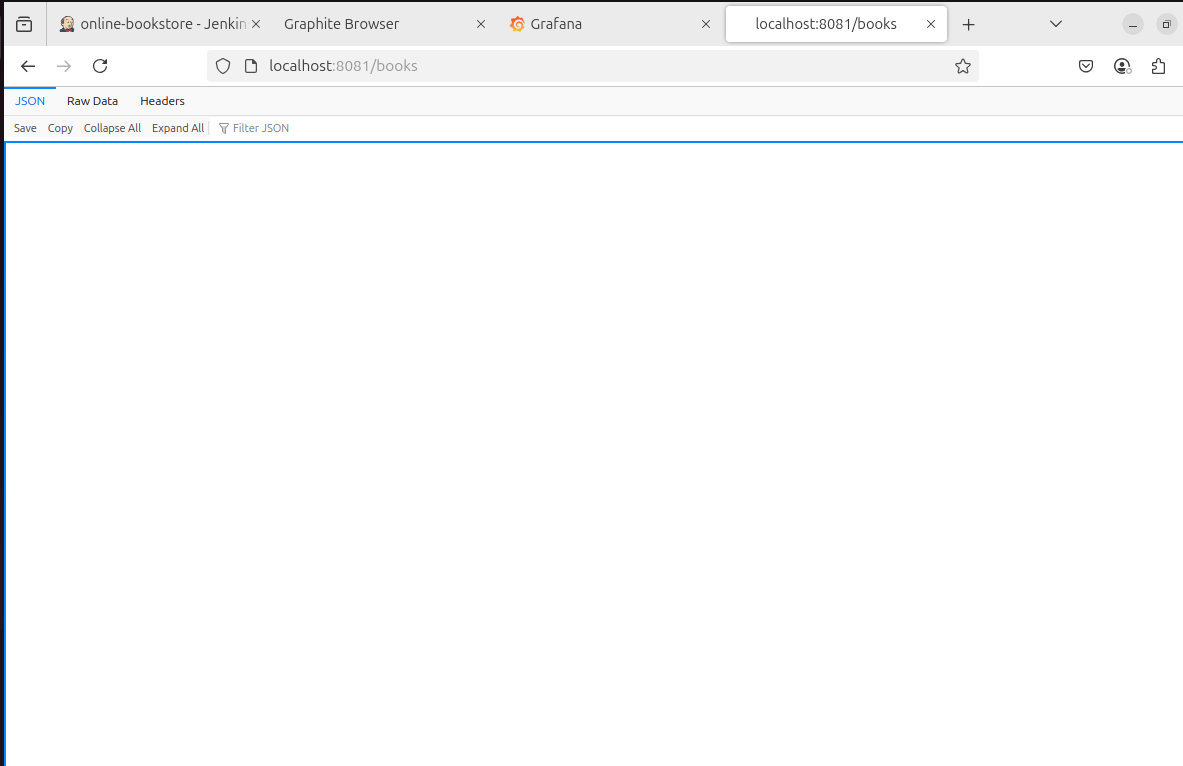
If the build Succesed it will show BUILD SUCCESSFULL at the end 

With this your online-bookstore using Spring Boot is now running along with grafana and graphite

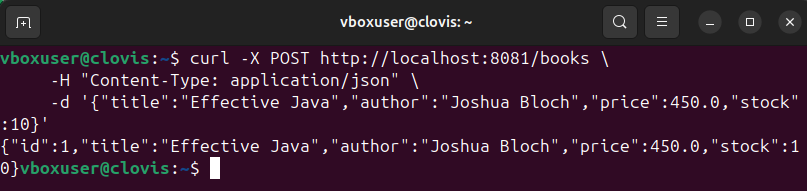
(the task waiting for Spring Boot app to be healthy failed 9 as graphite takes 10s step to send data as programmed in the application.properties)

We can check if the jenkins build was successful by running docker ps command in the terminal we should we 3 running docker containers ie: bookstore ,graphite and grafana

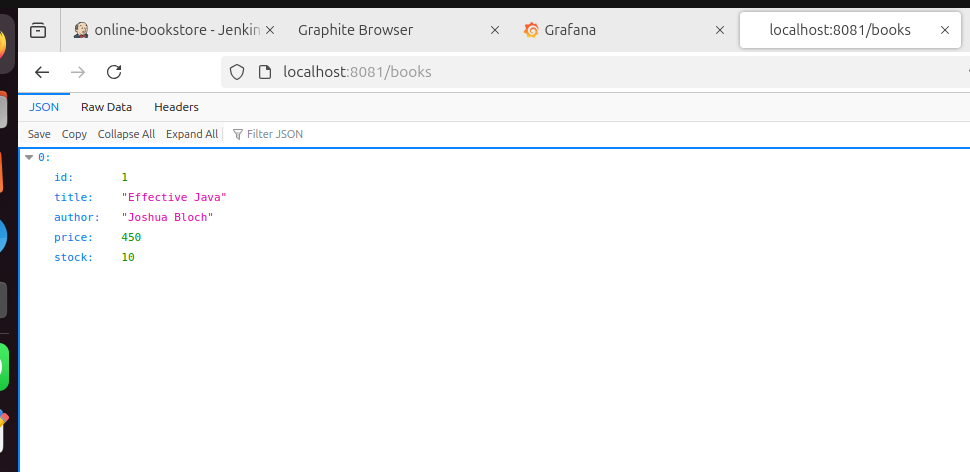
Now if you open htttp://localhost:8081/books you will see the Spring Boot online-bookstore with an empty page



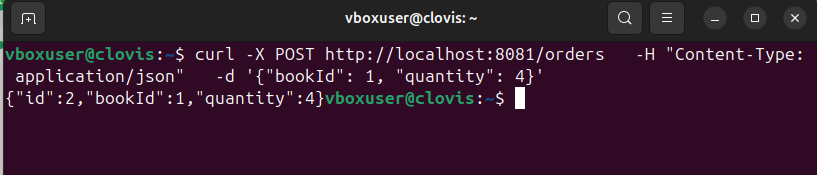
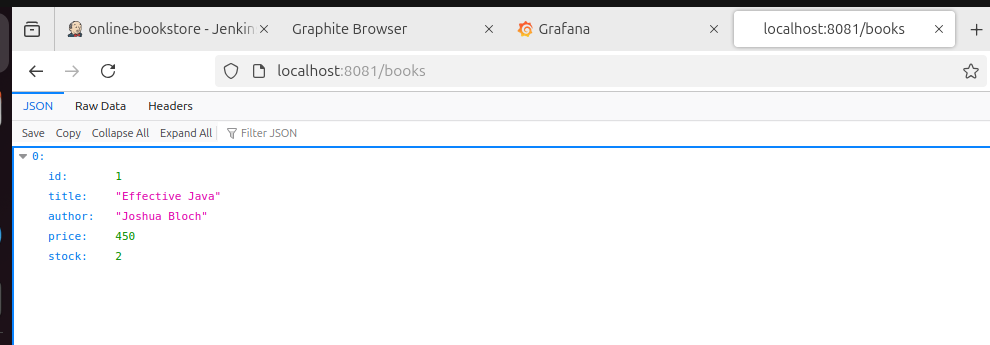
Now we can add books to the online-bookstore using POST



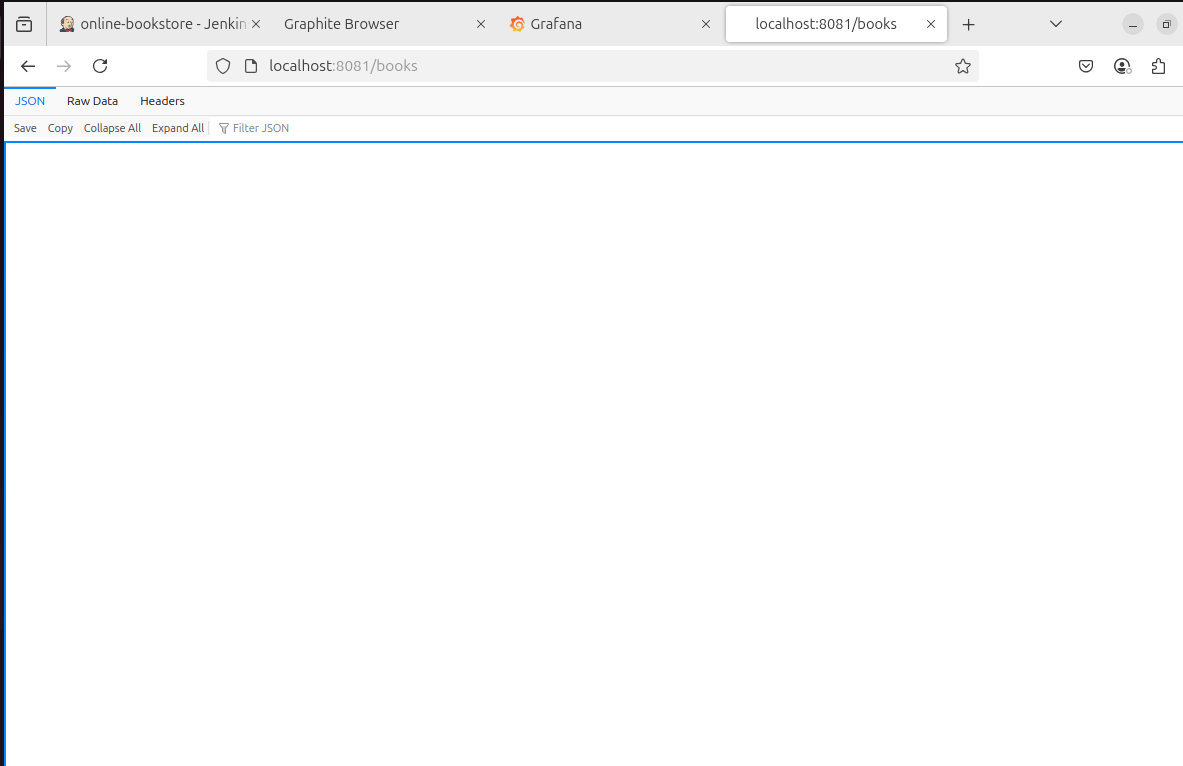
This will add the book “Effective Java” to our bookstore



We can place Orders for books by their ID number (please note: I ran this command multiple times hence the discrepancy in stock left)

We can also Delete books   

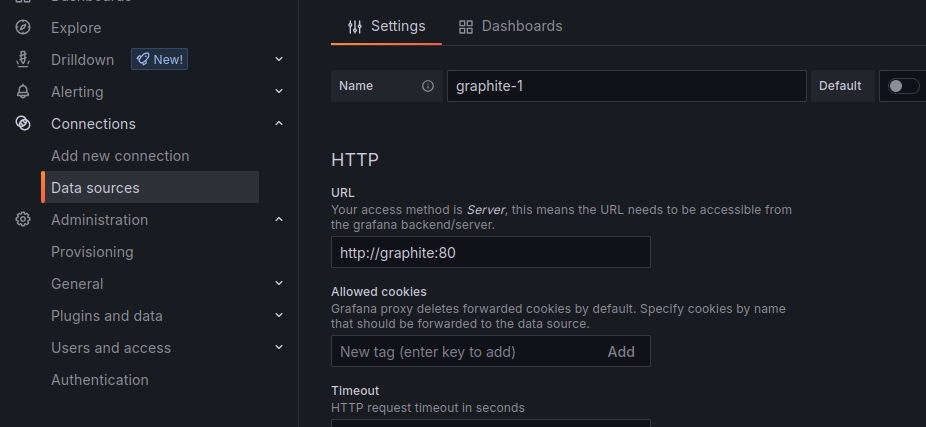
Next we will monitor our application using graphite to collect and store the data and Grafana to visualise it

Open <http://localhost:8181>

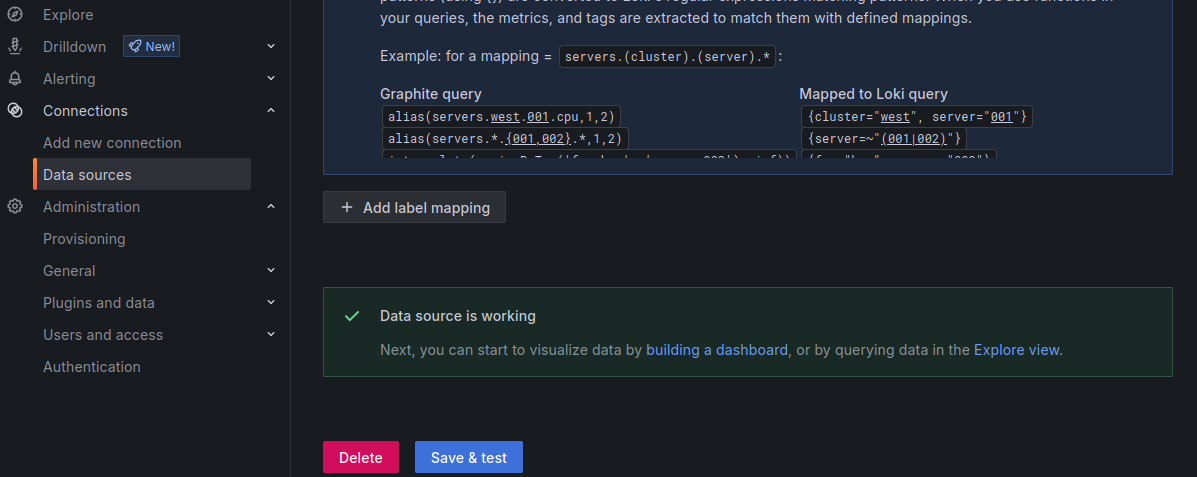
Select Data sources

Add a data source and choose Graphite

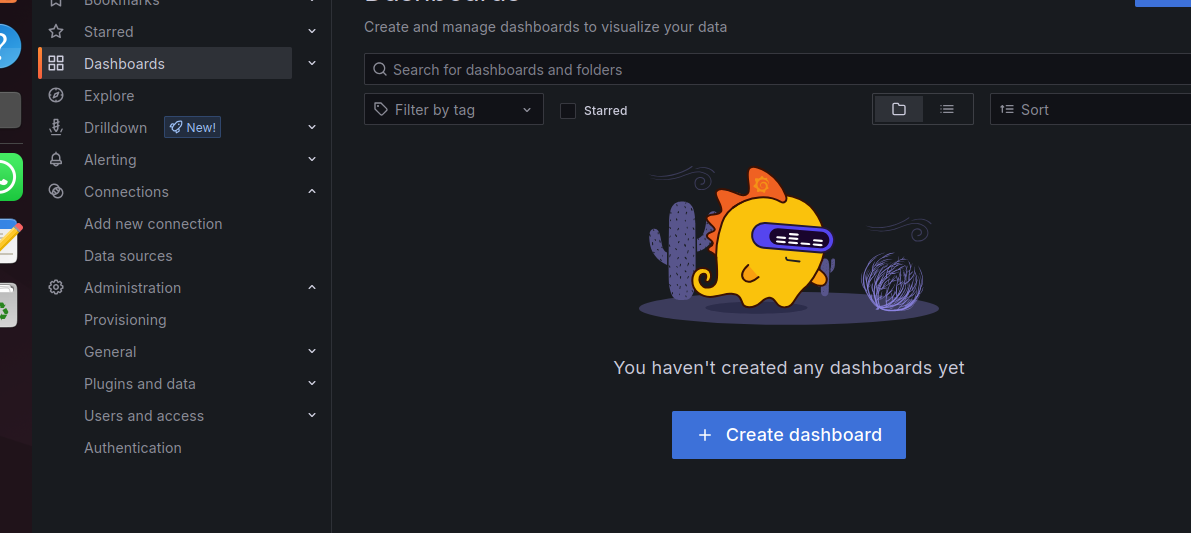
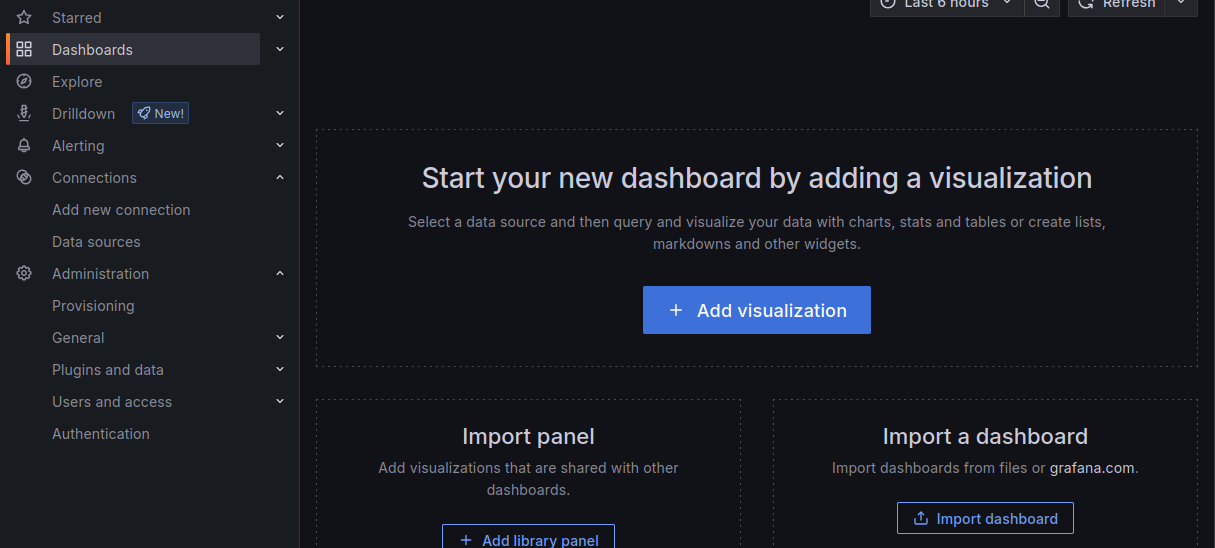
Give the URL as <http://graphite:80> and click save and test



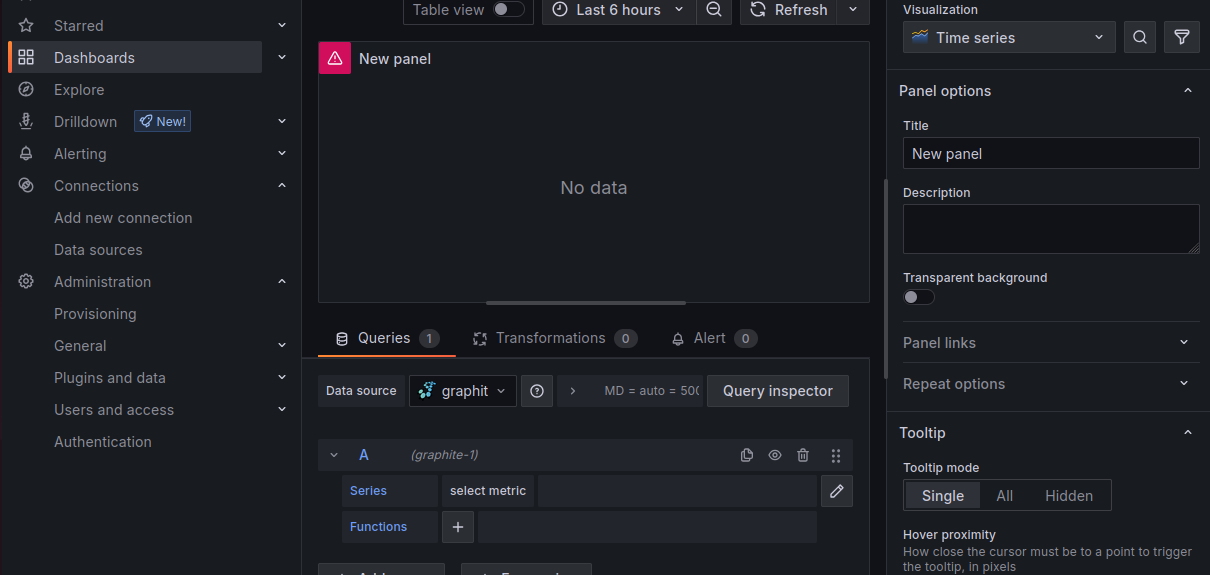
It should say Data source is working



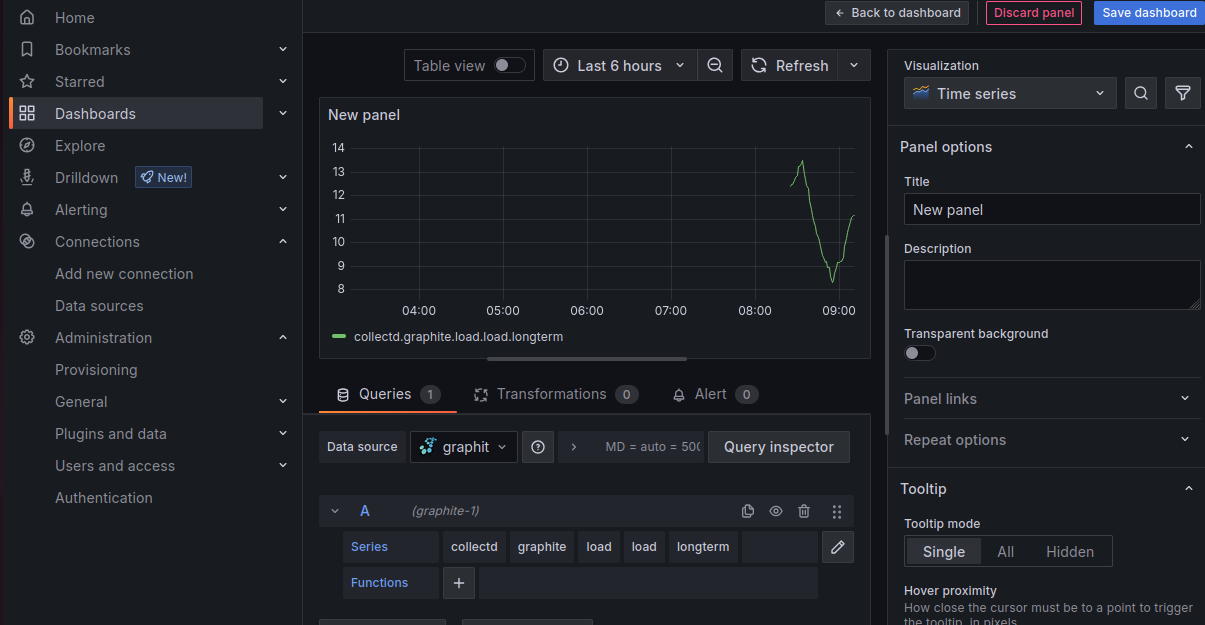
Next we will create a Dashboard to see the data  
go to Dashboard and click on Create dashboard and click on add visualization

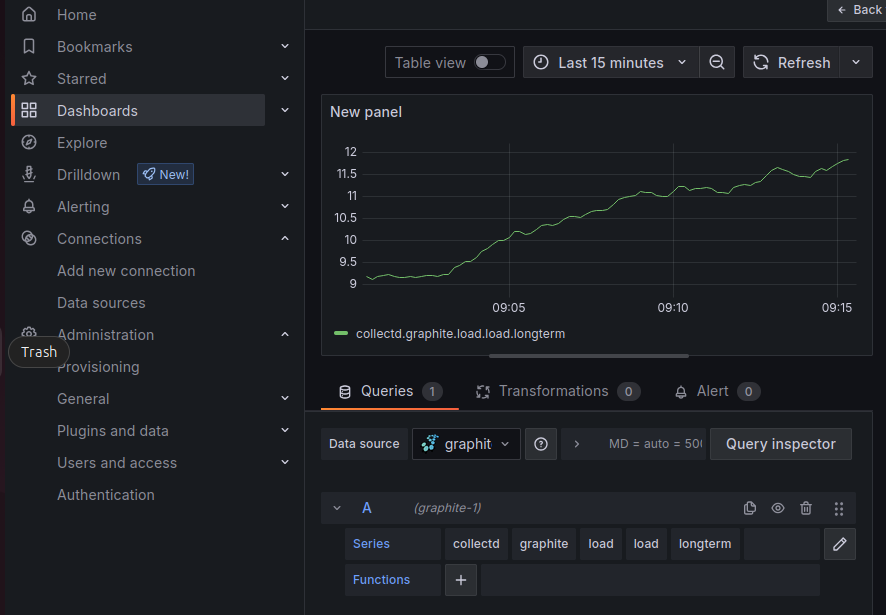
Then select the data source you just created (eg:graphite\_1)

And besides Series click on select metric  


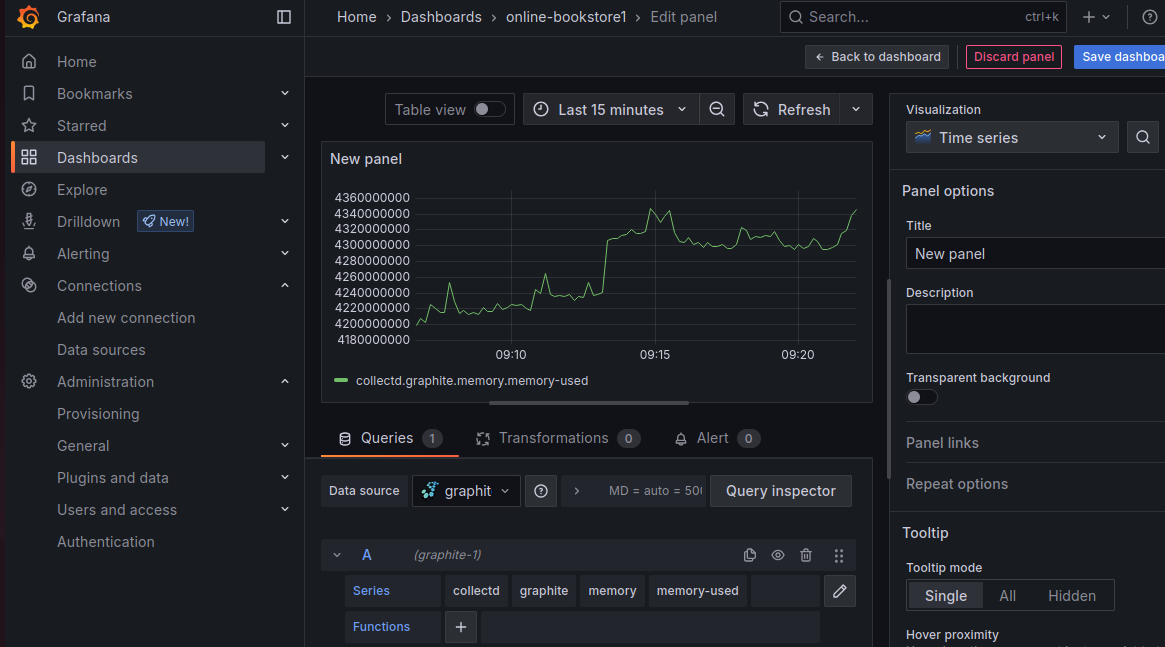
Then select collectd then graphite then load then load then longterm (here collectd is the service pf collectd graphite we are using to collect data from our application)



Then change the duration to last 15 minutes for better understanding (as the application has not been running for 6 hours)



Do the same to monitor more data about your application



Save and go back to dashboard and view your dashboard  
