## **Jenkins Build Pipeline: Graphical View and Chained Job Execution**

### **1. Upstream and Downstream Jobs (Job Triggering Logic)**

**Upstream Job**: A job that starts the build chain  
 **Downstream Job**: A job that is triggered automatically after the upstream job finishes

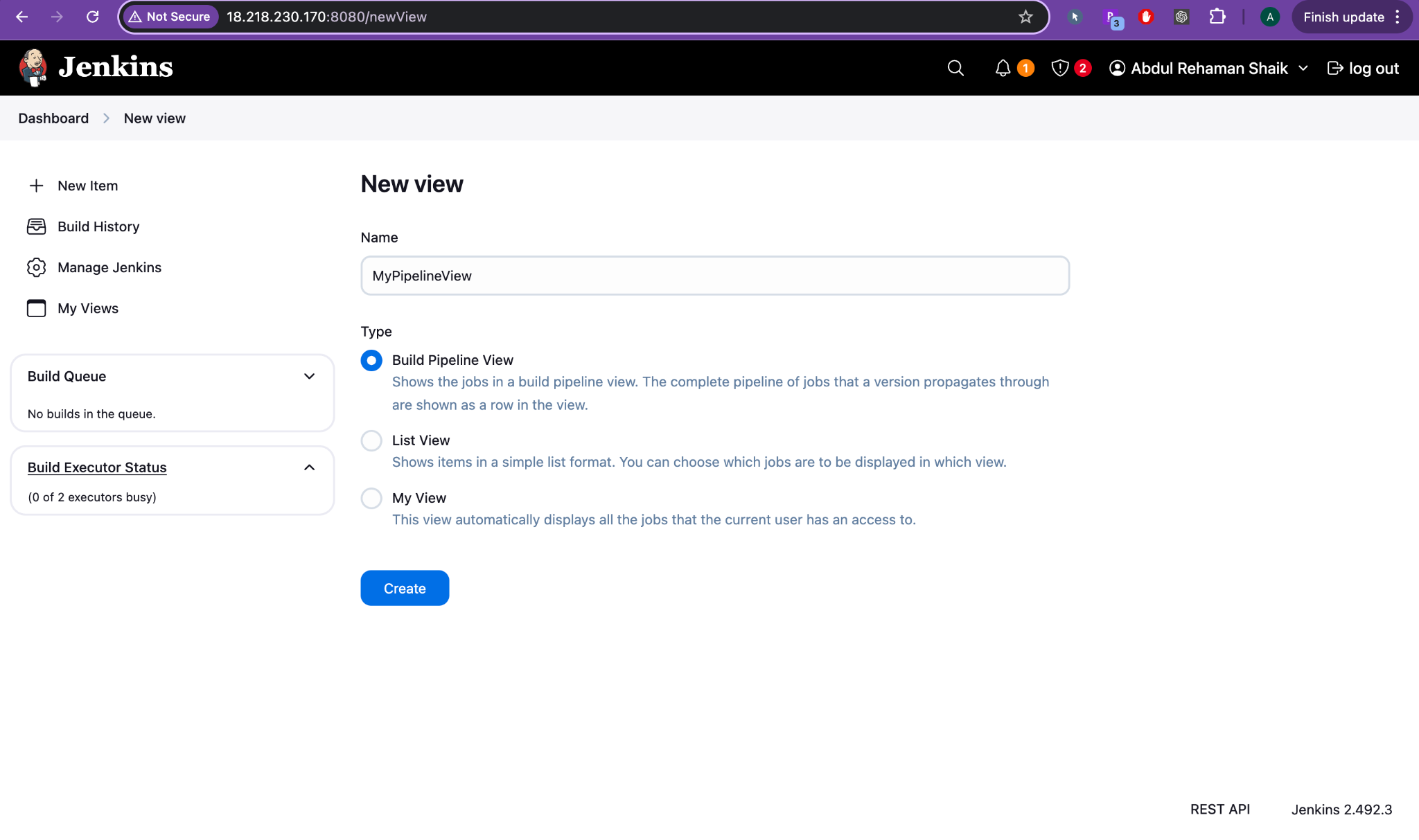
**Example:** If Job-A is configured as the upstream and Job-B is set to trigger after Job-A, then when Job-A builds successfully, Job-B starts automatically.

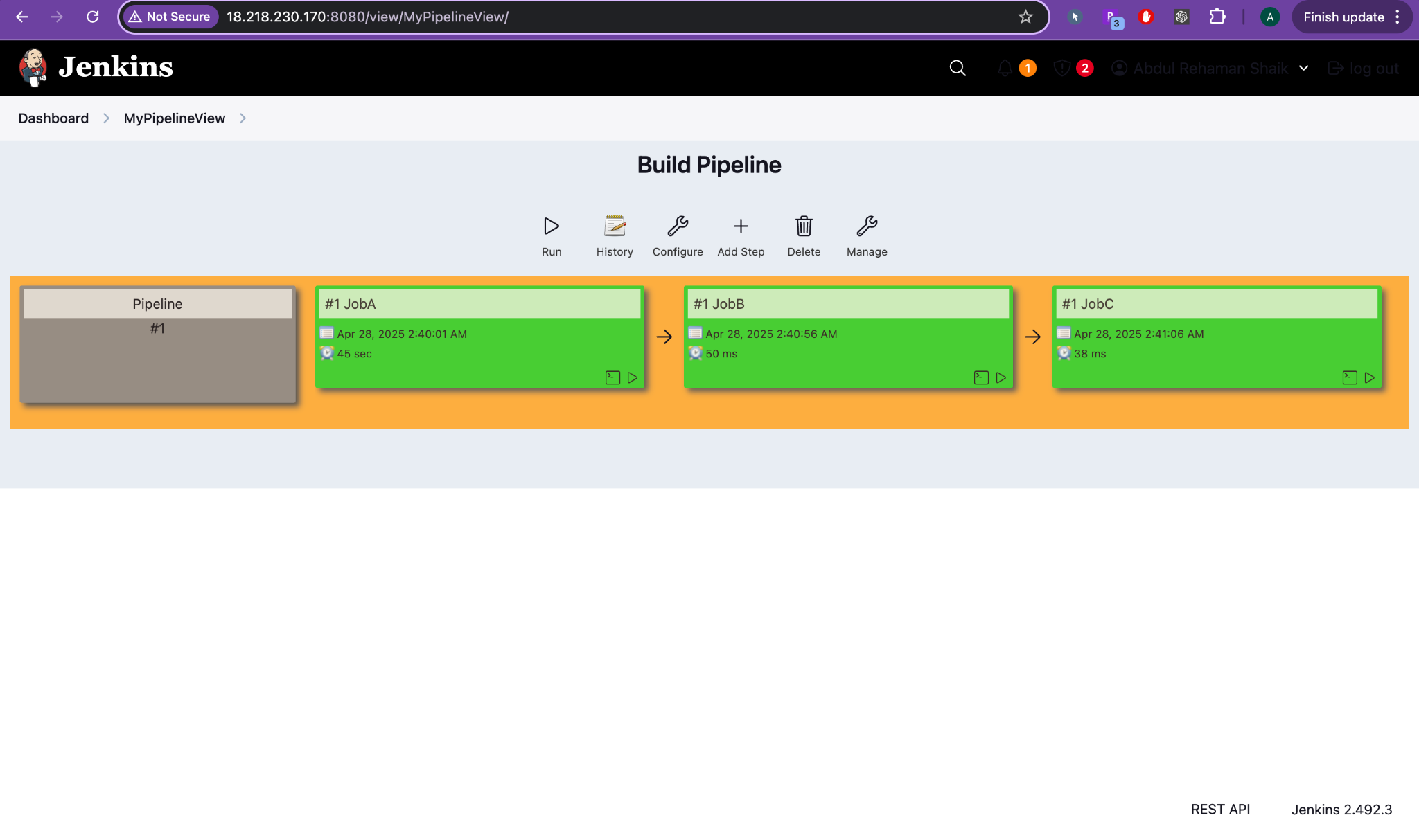
### **2. Visualizing Jobs with Graphical Representation**

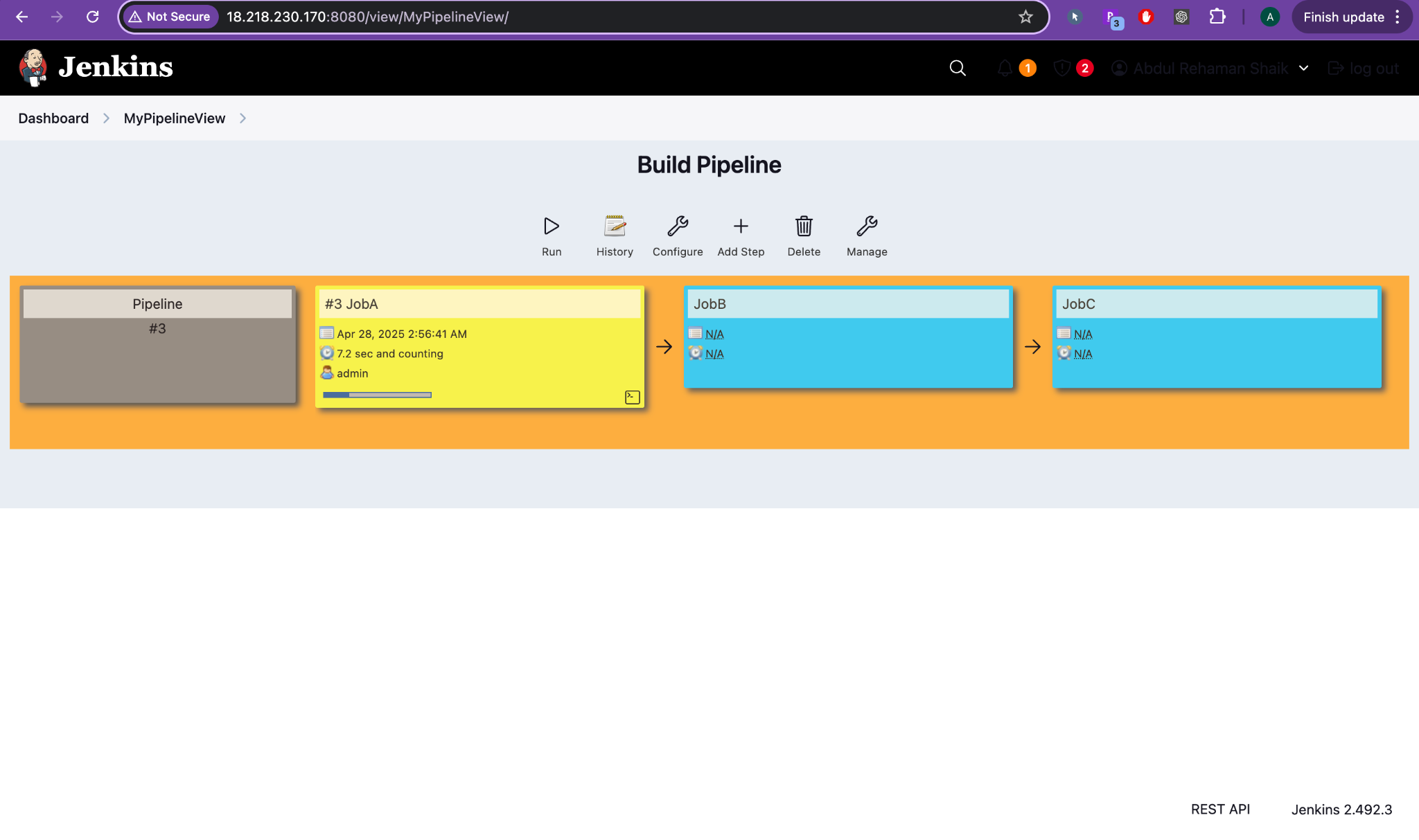
To see the pipeline visually (graphical format), use the **Build Pipeline View**.

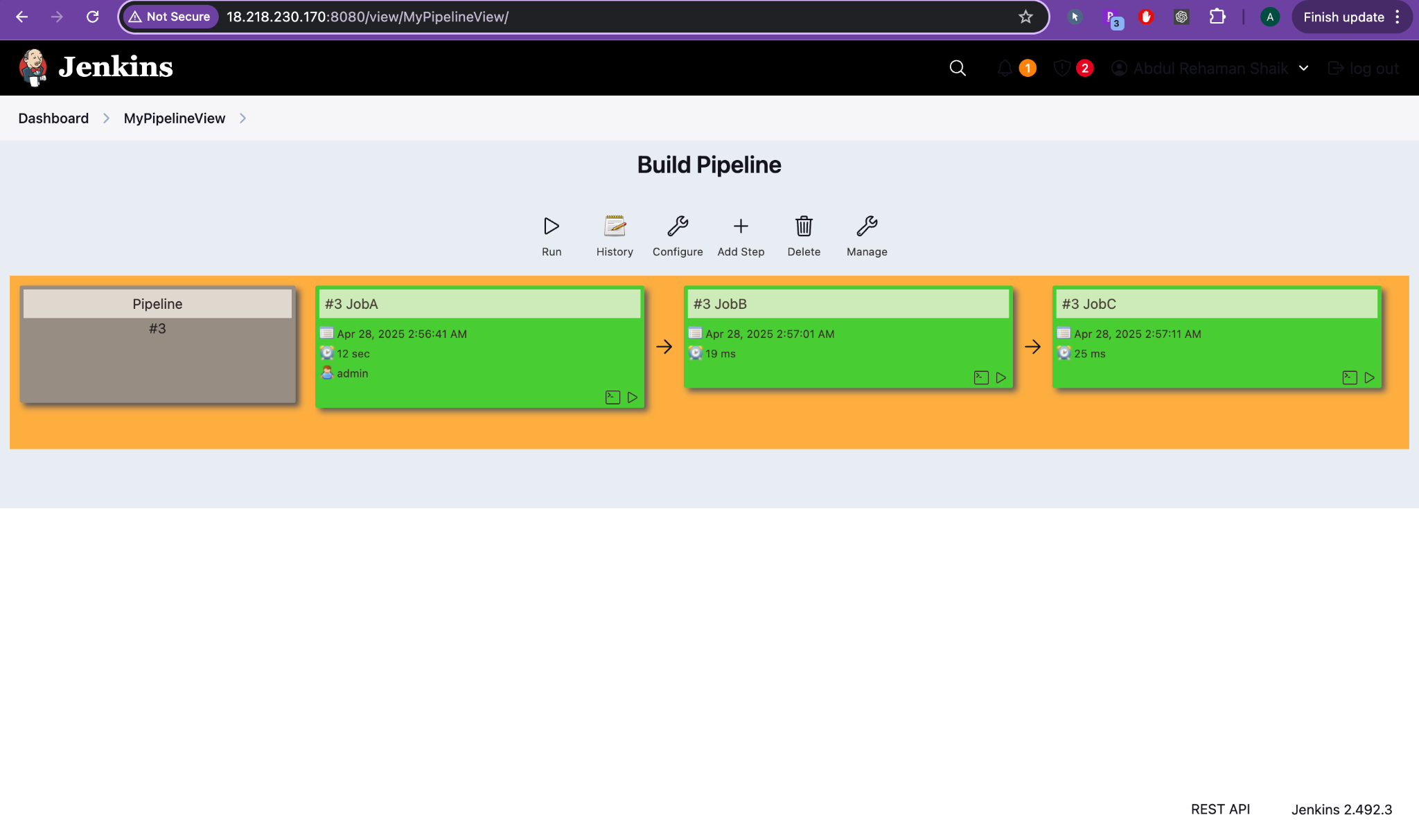
#### **Steps to Create a Build Pipeline View:**

1. In Jenkins dashboard, click the **“+”** icon (to create a new view)
2. Select **Build Pipeline View**
3. Give your view a name (e.g., MyPipelineView)









1. Click **OK**
2. In configuration:  
   * Set **Initial Job** (starting point, e.g., myfirstjob)
   * Enable **Refresh Frequency** (e.g., 5 seconds to auto-refresh view)
3. Save the view

Note: Only *List View* and *My View* are default Jenkins views; Build Pipeline View needs a plugin.

### **3. Installing Required Plugin (Build Pipeline Plugin)**

#### **Steps:**

1. Go to: **Manage Jenkins** → **Manage Plugins**
2. Click the **Available** tab
3. Search for **Build Pipeline Plugin**
4. Select it and choose:  
   * **Install without restart** OR
   * **Download now and install after restart** (if restart is required)
5. After installation, restart Jenkins if necessary

Use **“Check now”** to refresh plugin list  
 Always ensure no jobs are running before restart

### **4. Chaining Jobs (Configuring Downstream Jobs)**

#### **Scenario:**

Trigger mysecondjob automatically when myfirstjob completes successfully.

#### **Steps:**

1. Go to **Configure** of the downstream job (mysecondjob)
2. Scroll to **Build Triggers**
3. Enable **“Build after other projects are built”**
4. Enter myfirstjob in the text box
5. Click **Apply** → **Save**

Now Jenkins understands the job dependency

### **5. Add Script to First Job (Optional)**

To add a shell script to the first job:

* Go to: myfirstjob → **Configure** → **Build** section
* Add a **Build step**: Select **Execute shell**
* Example: echo "This is first build"

After saving, triggering myfirstjob will execute the shell script and then automatically trigger the downstream mysecondjob.

### **6. Colors in Pipeline View (Build Status Indicators)**

In the **Build Pipeline View**, you'll notice:

* **Green**: Successful build
* **Red**: Build failed
* **Blue (or yellow depending on theme)**: Unstable build

These visual cues help track the status of each job in the pipeline.

### **7. Bonus: Triggering Build Through Scripting**

You can trigger Jenkins jobs via command-line or script using:

#### **a. Curl with Jenkins API:** curl -X POST http://<jenkins-url>/job/<job-name>/build \

#### --user username:api\_token

#### **b. Jenkins CLI Tool:** java -jar jenkins-cli.jar -s http://<jenkins-url> build <job-name>

#### You must generate an API token from your Jenkins account for authentication.

#### 

### 

### 

### Real-World Example:

#### Use Case: You have a 3-stage build pipeline:

#### Code-Checkout (Git clone)

#### Build-and-Test (compile + run unit tests)

#### Deploy-to-Dev

#### This can be modeled in Jenkins as:

#### Set Code-Checkout as the initial job in Build Pipeline View

#### Set Build-and-Test → build after Code-Checkout

#### Set Deploy-to-Dev → build after Build-and-Test

#### Each job is triggered automatically after the previous one completes successfully.

#### 

# **STEP-BY-STEP GUIDE (GITHUB ➔ JENKINS ➔ JOBA ➔ JOBB ➔ JOBC)**

## **STEP 1: Correct Webhook URL**

✅ Use this URL: http://18.218.183.190:8080/github-webhook/

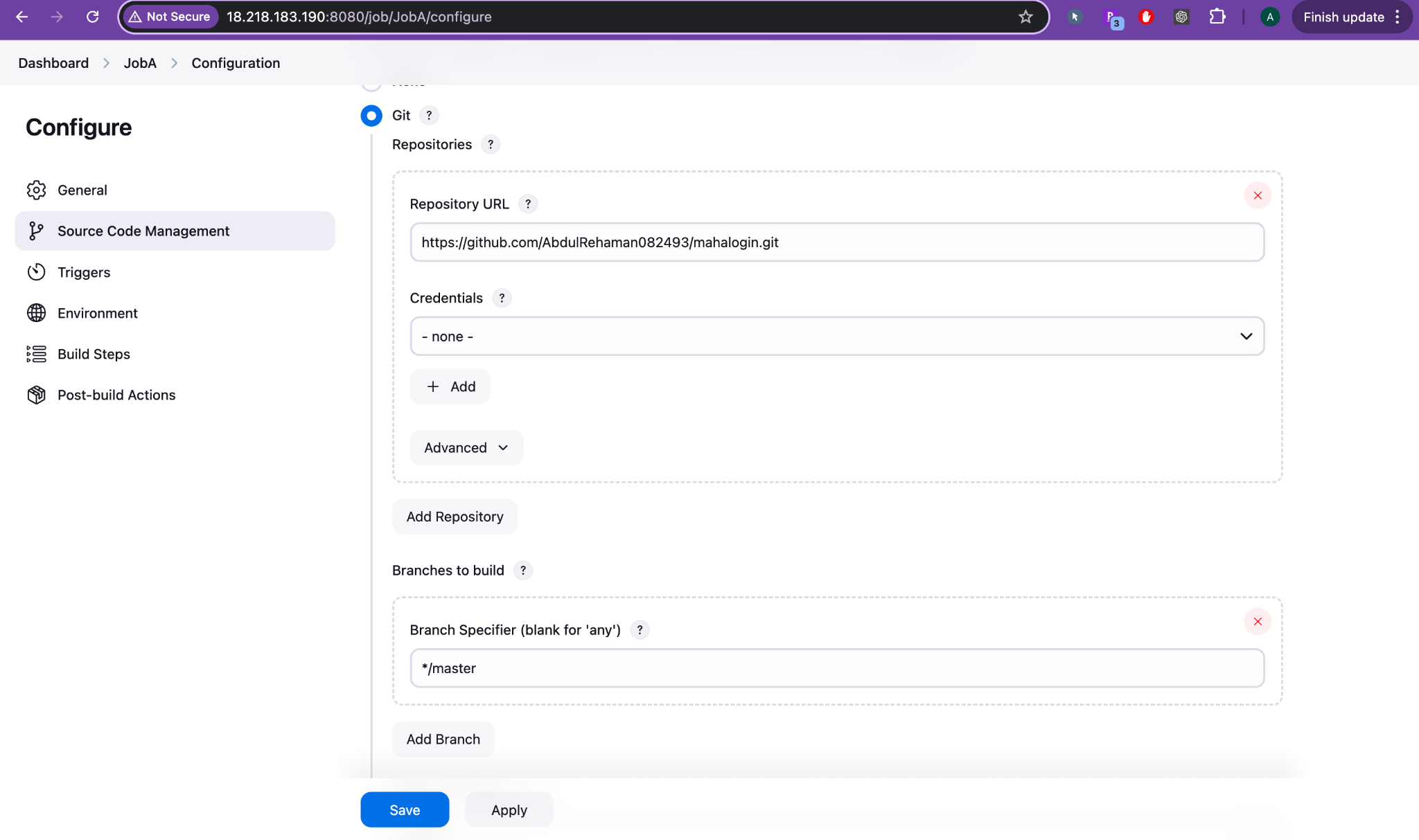
## 

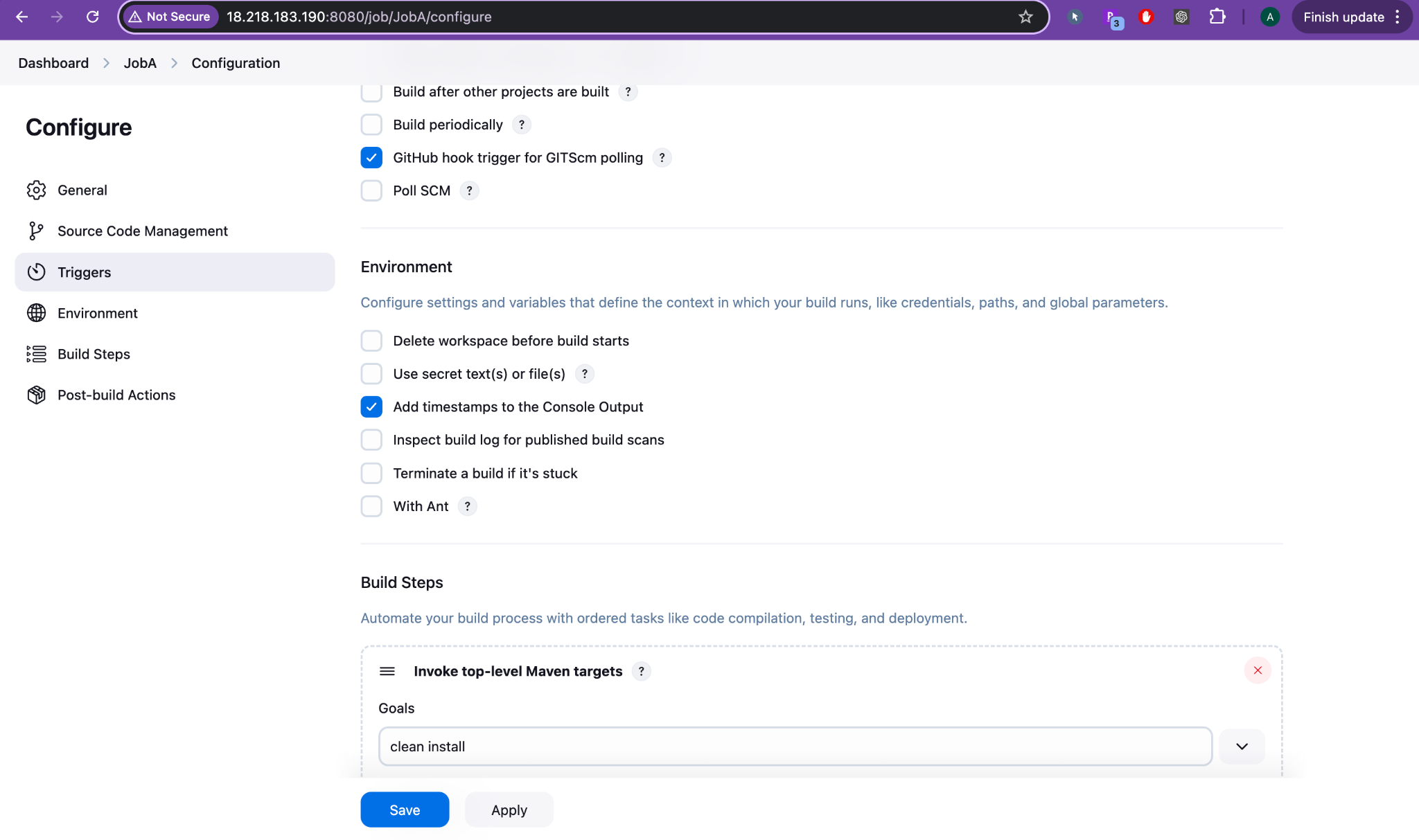
## 

## 

## **STEP 2: Fix Jenkins JobA Settings**

Inside **JobA**:





### **2.1 Source Code Management**

* Git → Repository URL →  
   https://github.com/AbdulRehaman082493/mahalogin.git
* Credentials:  
  + **If your repo is Private**, create credentials.  
    - Go to Jenkins → Credentials → Add:  
      * Kind: **Username and Password** (or Personal Access Token).
      * Username: GitHub username
      * Password: GitHub token
    - Then select it here.
  + **If Public Repo**, keep Credentials: - none -.
* Branches to build:  
   ✅ \*/master (Correct if you are using master branch.)

### **2.2 Triggers**

✅ Tick the checkbox:

* **GitHub hook trigger for GITScm polling** ✔️
* **Poll SCM** (❌ Don't check. Not needed if webhook is coming.)

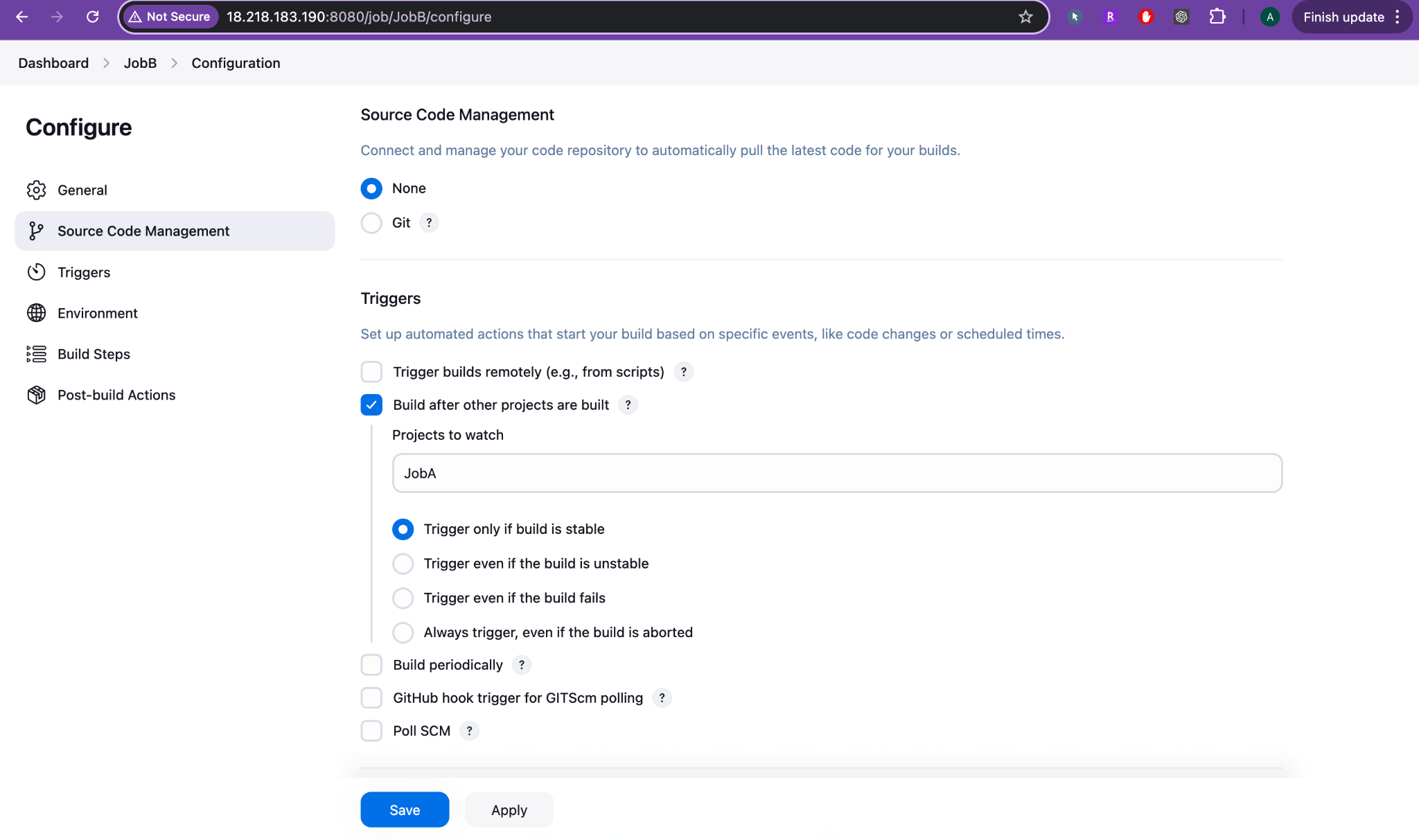
### **2.3 Build Steps**

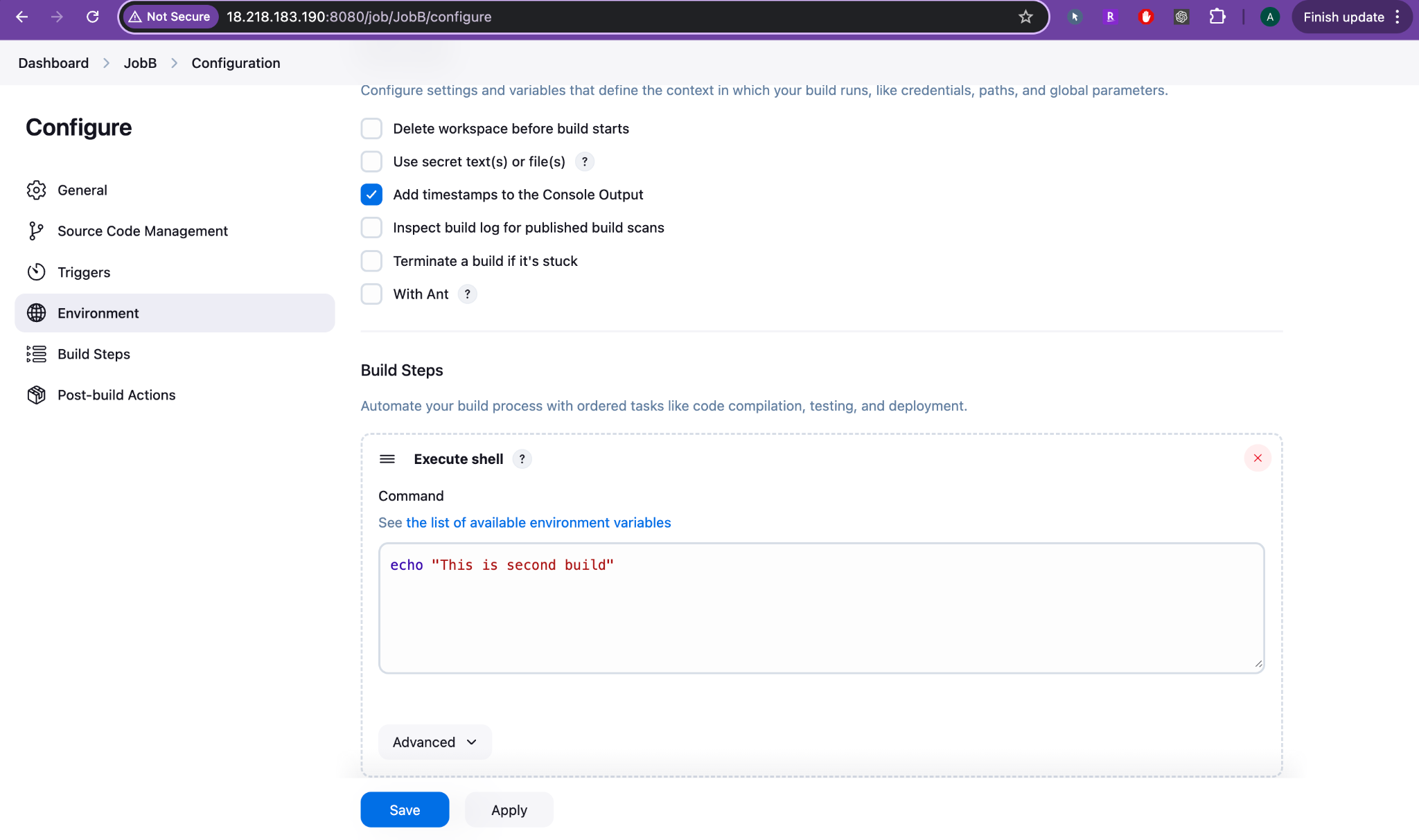
✅ Build step:

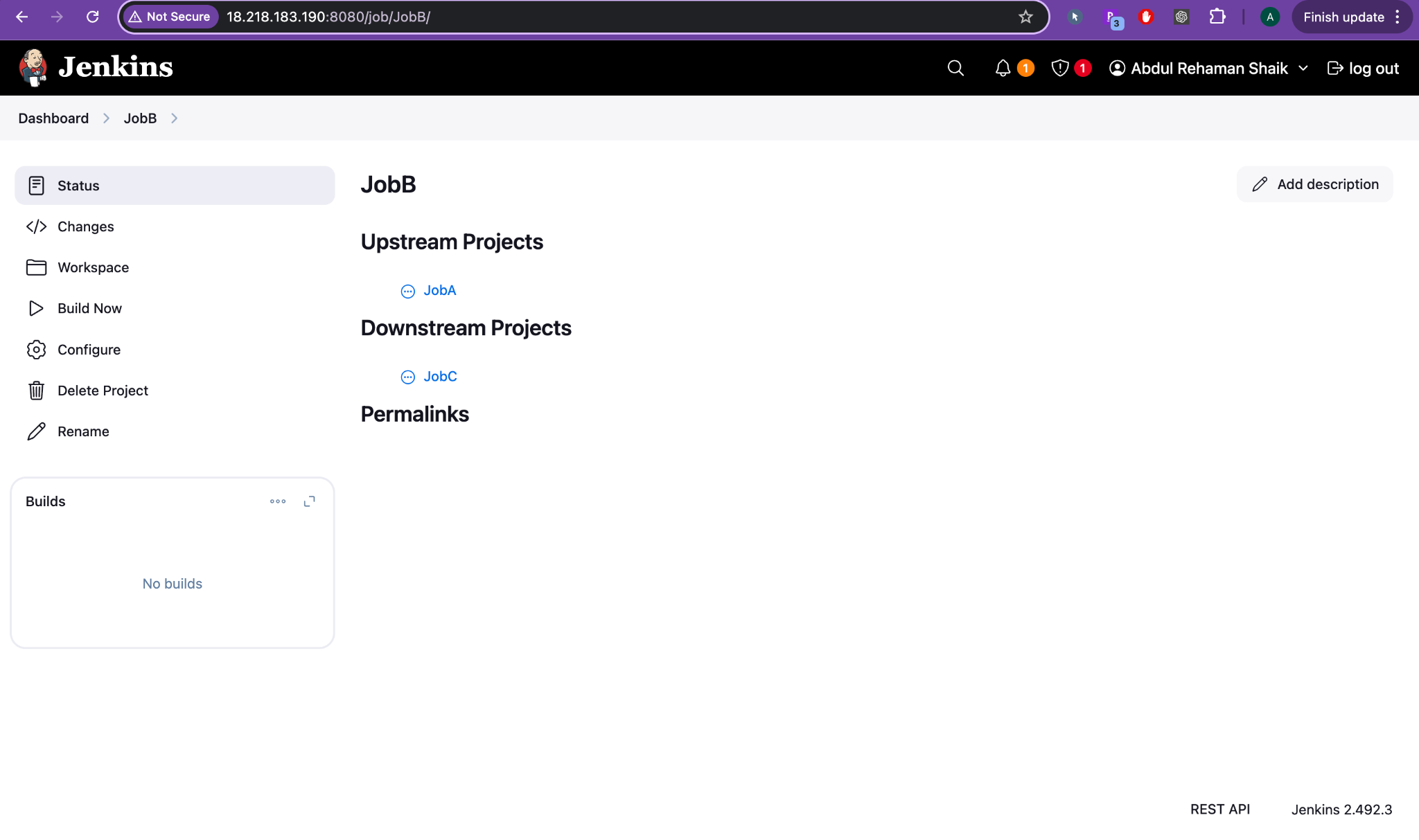
* Invoke top-level Maven targets.
* Goals:  
   clean install

## **STEP 3: Fix Jenkins JobB Settings**

Inside **JobB**:



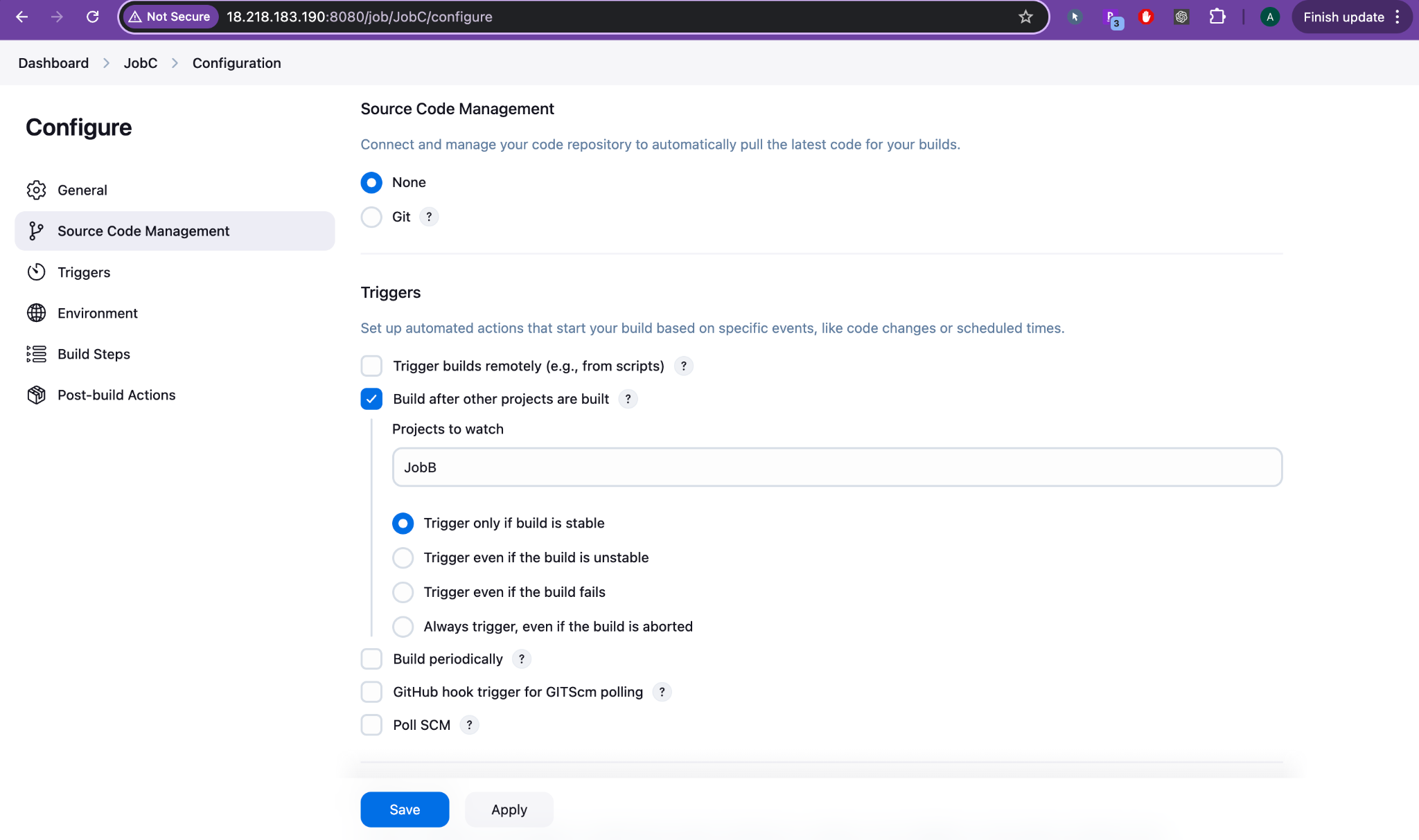


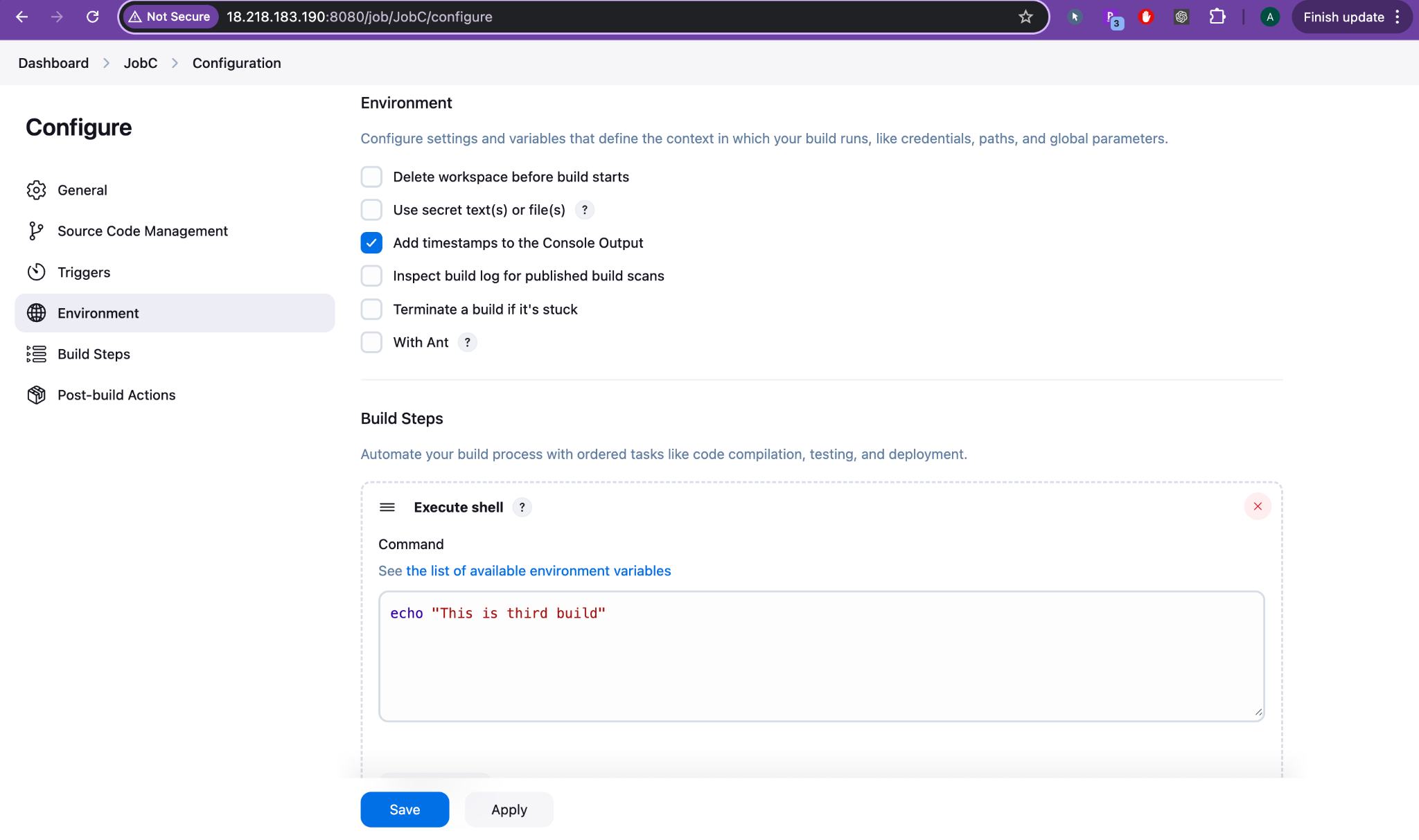


* Source Code Management → None (because no git clone needed here).
* Triggers:  
  + Tick **Build after other projects are built** ✔️
  + Projects to watch:  
     JobA
  + "Trigger only if build is stable" (default and good).
* Build Steps:  
  + Execute Shell →  
     echo "This is second build"

## **STEP 4: Fix Jenkins JobC Settings**

Inside **JobC**:





* Source Code Management → None.
* Triggers:  
  + Tick **Build after other projects are built** ✔️
  + Projects to watch:  
     JobB
* Build Steps:  
  + Execute Shell →  
     echo "This is third build"

## **STEP 5: Configure GitHub Webhook**

In GitHub:

* Open your repo: mahalogin
* Go to: Settings → Webhooks → Add Webhook
* Fill:  
  + **Payload URL**:  
     http://18.218.183.190:8080/github-webhook/
  + **Content type**:  
     application/json
  + **Event**:  
     "Just the push event"
* Click **Add webhook** ✅

## **STEP 6: Test the Webhook**

* Go to GitHub Repo → Make any small change (example: update README.md) and push.
* GitHub should send payload (green tick if successful).

✅ In Jenkins:

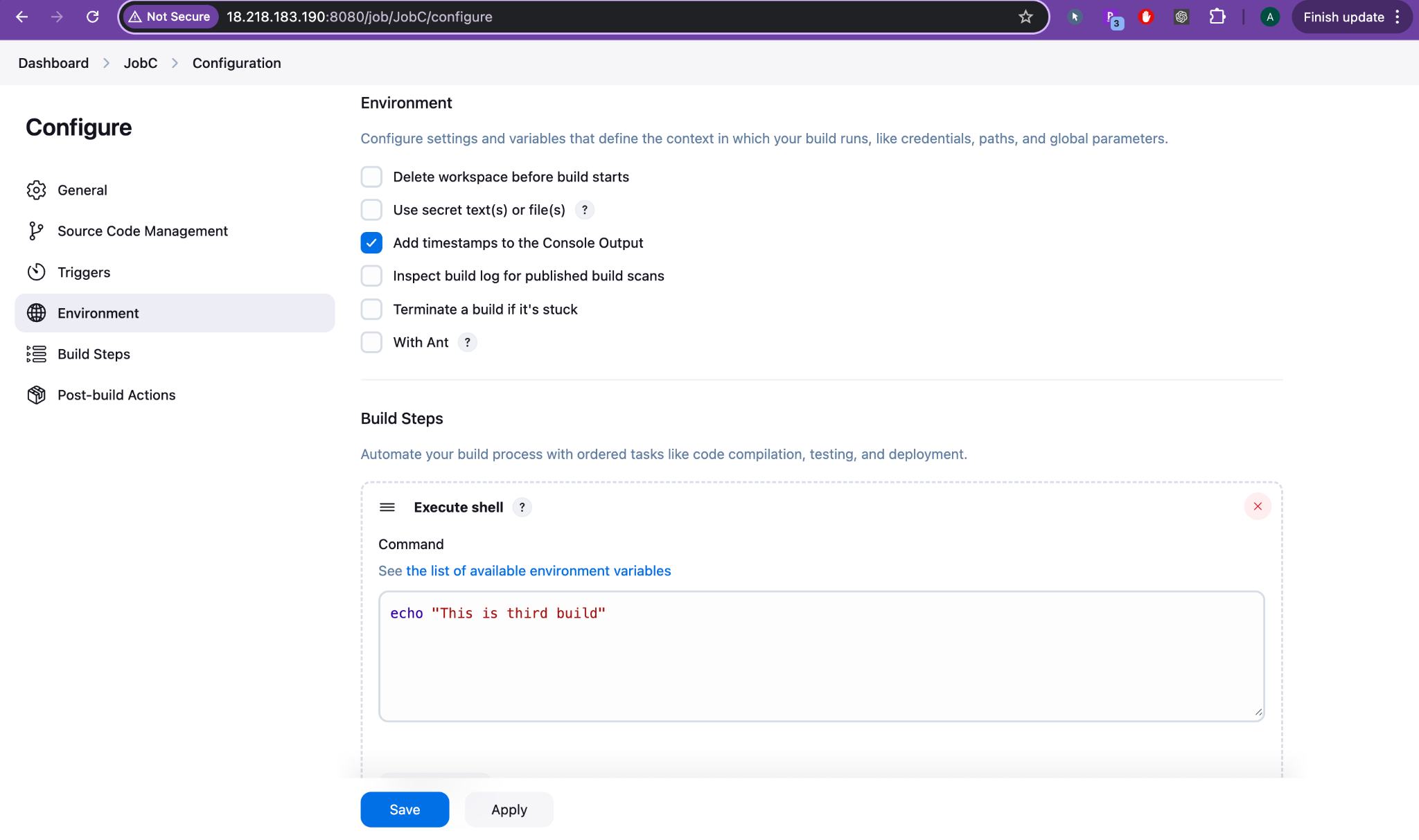
* JobA will start automatically.
* JobB will start after JobA success.
* JobC will start after JobB success.

🎯 FINAL CHAIN

Git Push ➔ GitHub Webhook ➔ Jenkins JobA ➔ Jenkins JobB ➔ Jenkins JobC

📢 IMPORTANT NOTES:

| **Topic** | **Details** |
| --- | --- |
| Plugin | Make sure **GitHub plugin** is installed in Jenkins |
| Jenkins IP | IP must be Public and Port 8080 open (Security Group if AWS) |
| Credentials | Only if repo is Private |
| Webhook testing | Webhook → "Recent Deliveries" → Green tick ✅ means success |



http://18.218.230.170/