In Jenkins, **Upstream** and **Downstream** refer to the relationship between different jobs (projects), especially when they are configured to **trigger each other** as part of a pipeline or build chain.

### **🔁 Definitions:**

#### **🟩 Upstream Job:**

* A job that **triggers another job**.
* Think of it as the **source** job in a pipeline.
* Example: If **Job A** triggers **Job B**, then **Job A is the upstream of Job B**.

#### **🟦 Downstream Job:**

* A job that is **triggered by another job**.
* It executes **after** the upstream job finishes.
* In the same example, **Job B is the downstream of Job A**.

### **🔧 Real-World Example:**

You have a CI/CD pipeline with the following jobs:

Job A: Code Compile

Job B: Run Unit Tests

Job C: Deploy to Dev

#### **Relationship:**

* **Job A** triggers **Job B** → A is upstream of B, B is downstream of A
* **Job B** triggers **Job C** → B is upstream of C, C is downstream of B

### 

### 

📦 Use Cases:

| **Use Case** | **Upstream Job Role** | **Downstream Job Role** |
| --- | --- | --- |
| CI Pipeline | Compile source code | Run tests, create artifacts |
| CD Pipeline | Package application | Deploy to servers |
| Automated Testing | Trigger test data generation | Run automation scripts |
| Build Promotion | Trigger after successful build | Promote to staging/prod |

### **⚙️ How to Configure:**

#### **1. From Jenkins Job Configuration:**

* Open a job (e.g., **Job A**)
* Go to **Post-build Actions**
* Choose **"Build other projects"**
* Enter downstream job names (e.g., **Job B**)

#### **2. Or from Downstream Job:**

* Open **Job B**
* In **"Build Triggers"** section, check:  
  + ✅ **Build after other projects are built**
  + Set **Job A** as the upstream

🧠 Summary:

| **Term** | **Triggers** | **Triggered By** |
| --- | --- | --- |
| Upstream | Others | No |
| Downstream | No | Others |

This relationship helps **create automated build chains**, ensuring that jobs are executed in the right sequence based on dependencies.

✅ Jenkins Job Chain: **JobA → JobB → JobC**

## **🔹 Step-by-Step Execution Flow**

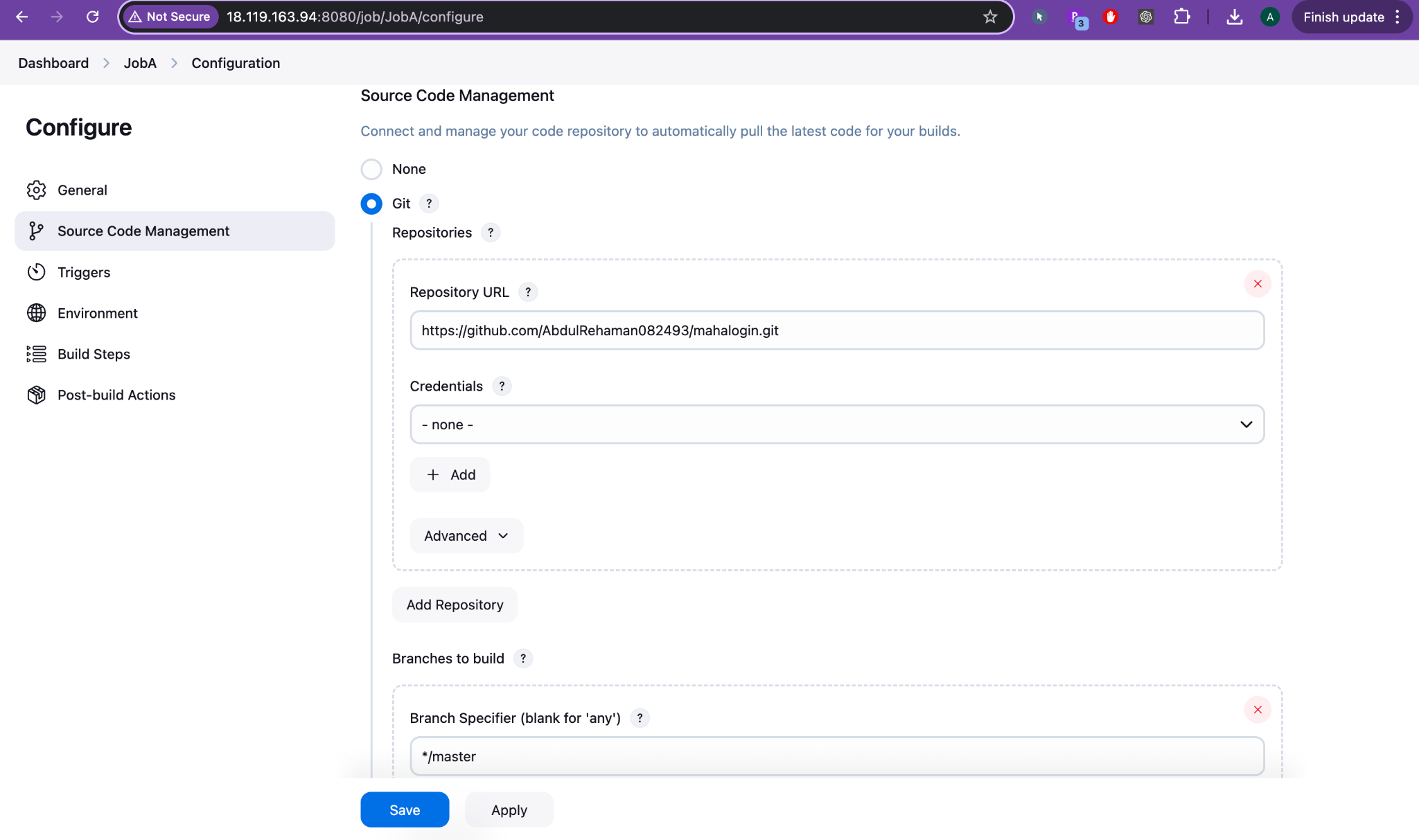
### **🔸 Step 1: GitHub Webhook or Manual Trigger Starts JobA**

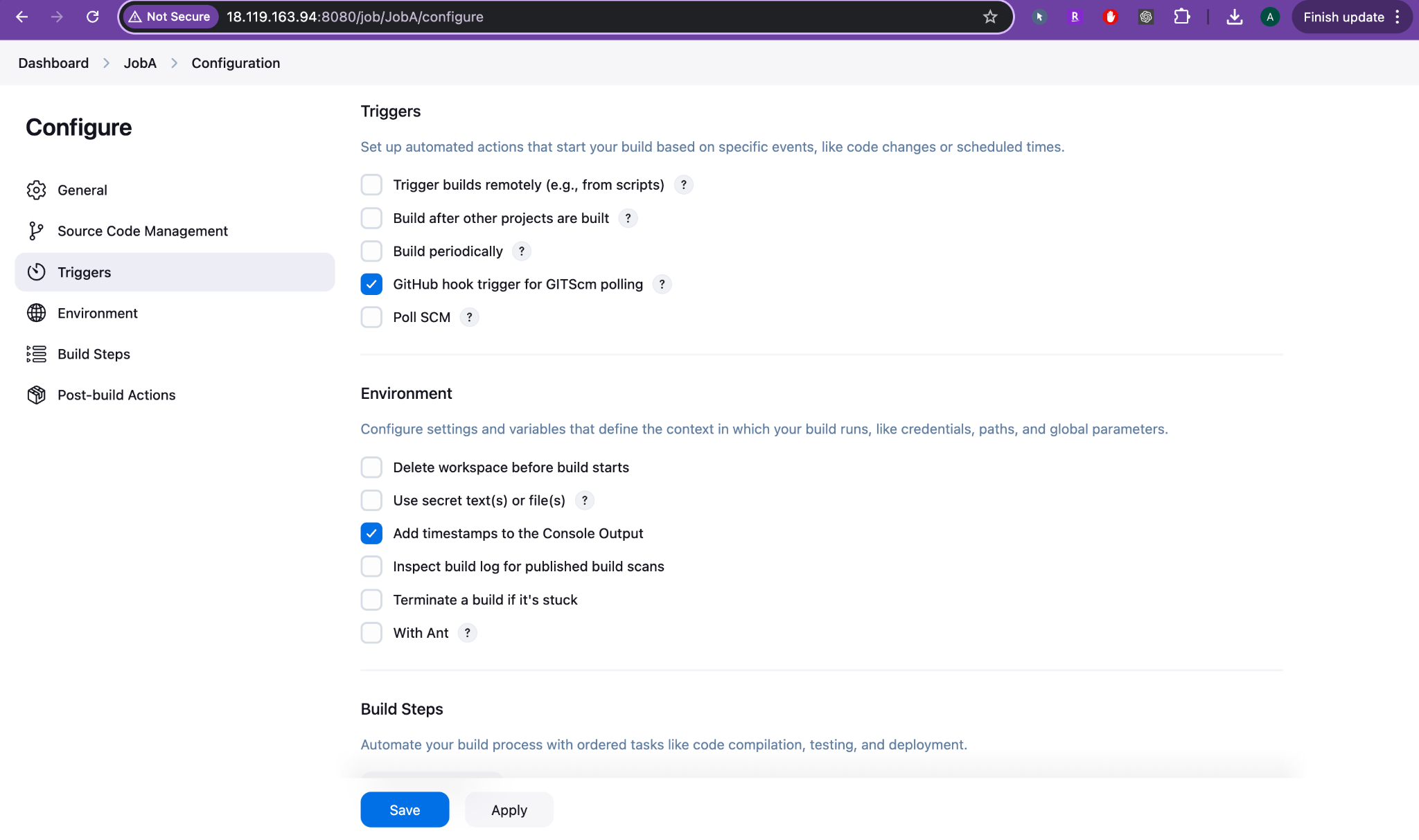
#### **🔧 What Happens:**

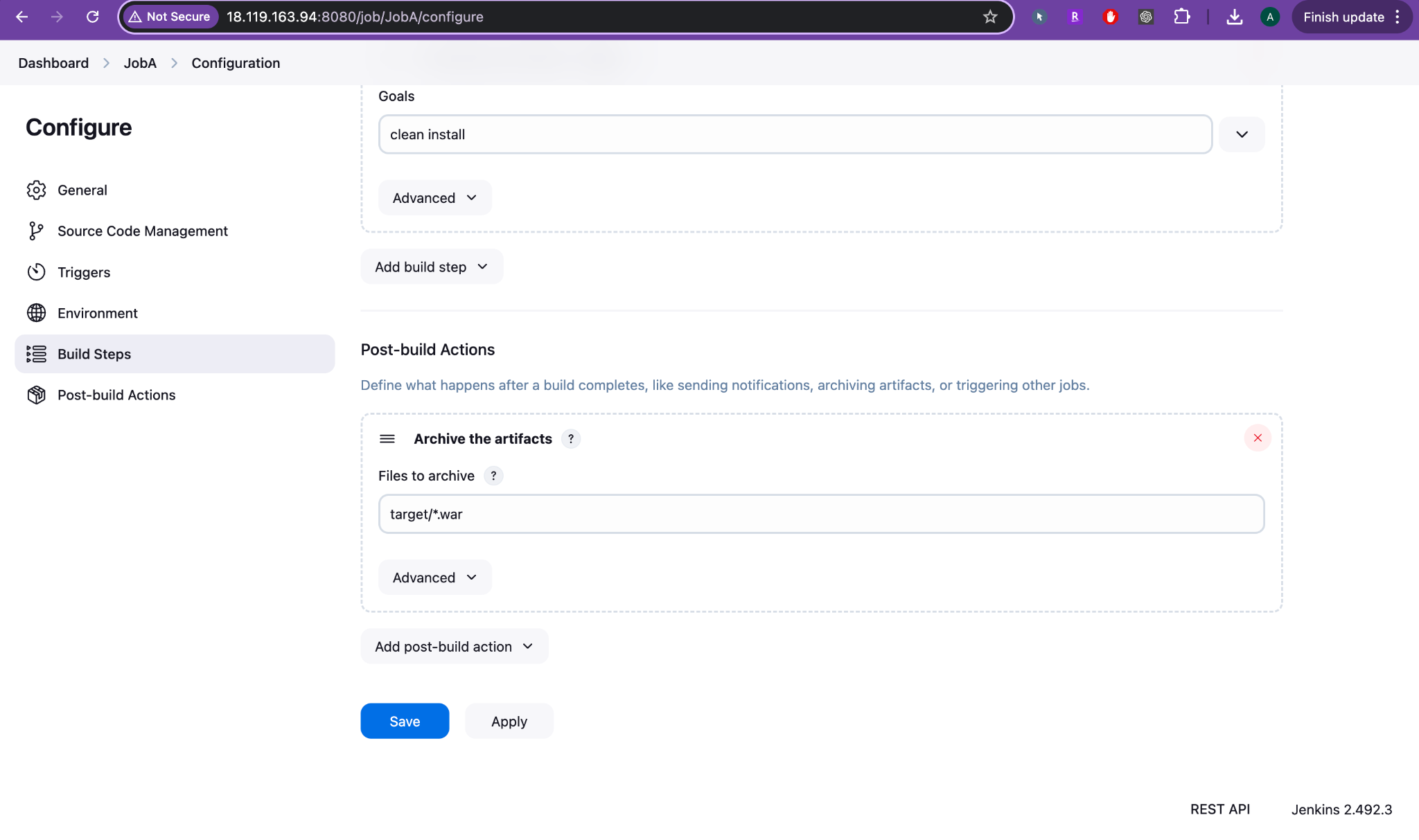
* **Jenkins detects a GitHub push event or you manually click “Build Now” for JobA.**
* **The GitHub hook trigger for GITScm polling is enabled, so:**
  + **Jenkins verifies that the webhook came from the correct repository.**
  + **Git plugin polls the repository and finds a new commit.**

#### **✅ Configuration:**

* **SCM Git URL: https://github.com/AbdulRehaman082493/mahalogin.git**
* **Branch: \*/master**

****

****

****

### **🔸 Step 2: JobA Runs Build & Archives Artifact**

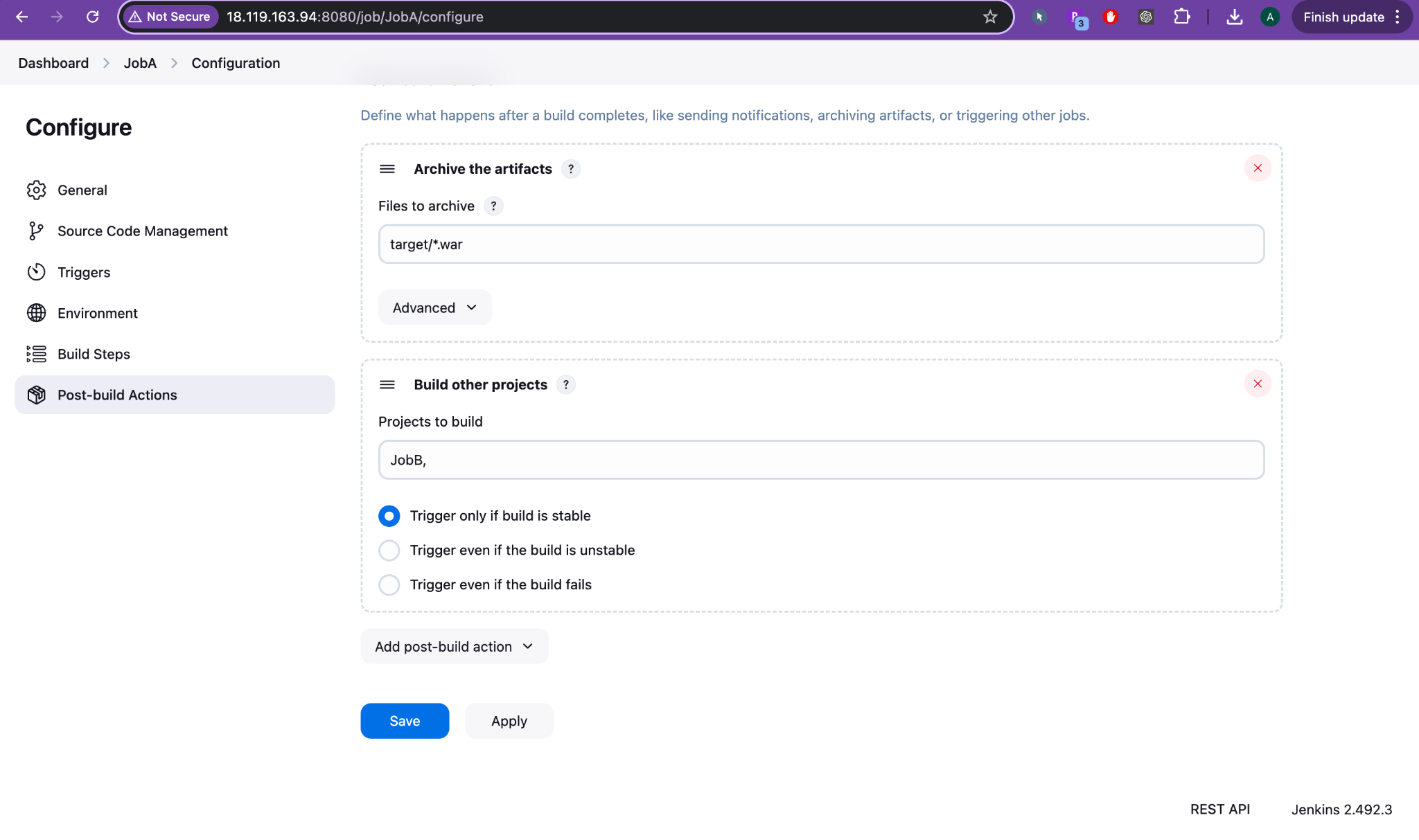
#### **🔧** What Happens:

1. Jenkins clones the GitHub repo (checked out at master).
2. Maven build runs: mvn clean install

This compiles your code and produces a .war file.

1. Shell command: echo "This is first build"
2. Jenkins archives the output: target/mahaLogin-2.0.war is saved as a build artifact.
3. Post-build Action:

**JobA triggers JobB** (✅ as downstream) **only if the build is stable.**

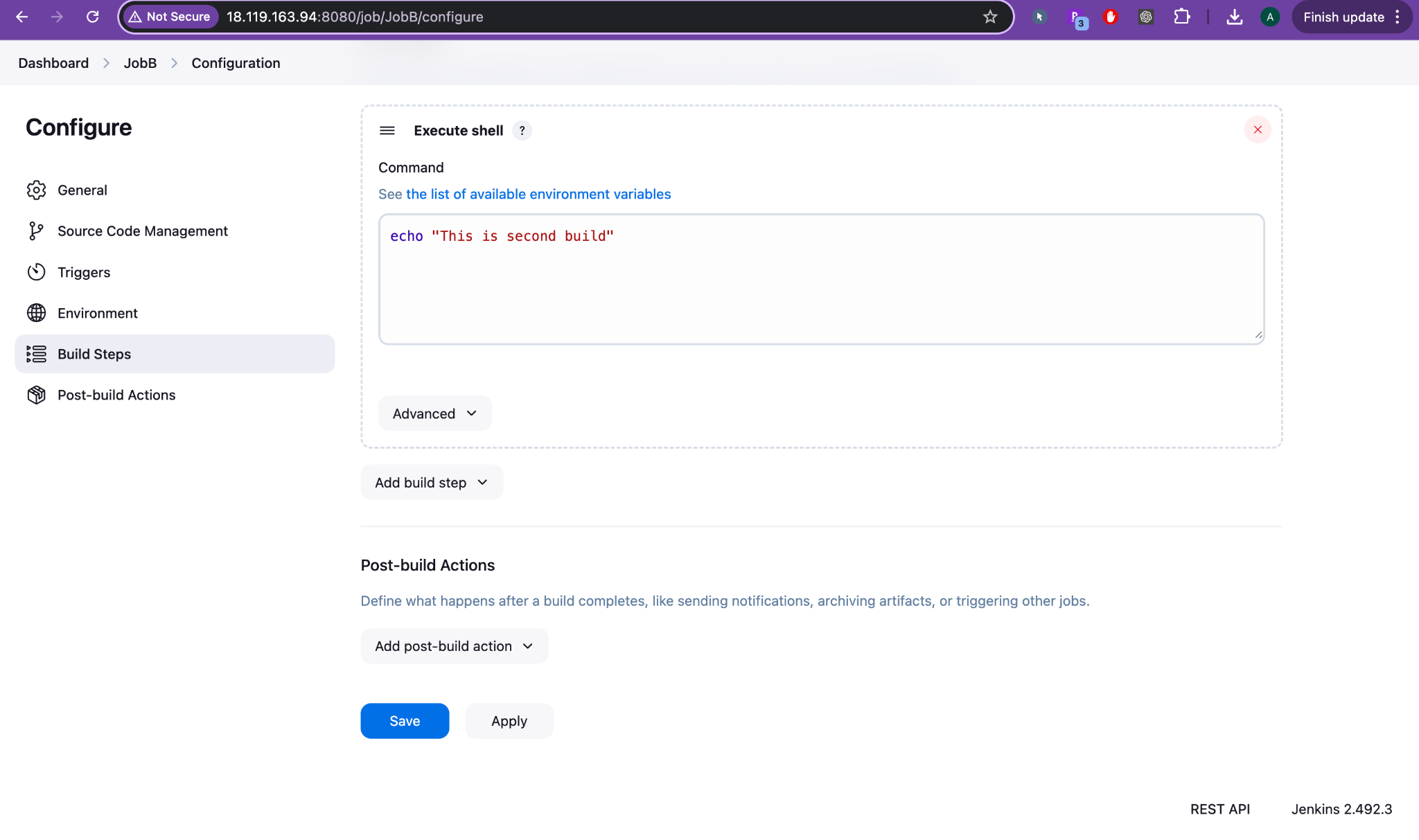


#### **✅ You See:**

* Last Successful Artifacts: mahaLogin-2.0.war
* Downstream Projects: JobB

### **🔸 Step 3: JobB Starts Automatically After JobA Success**

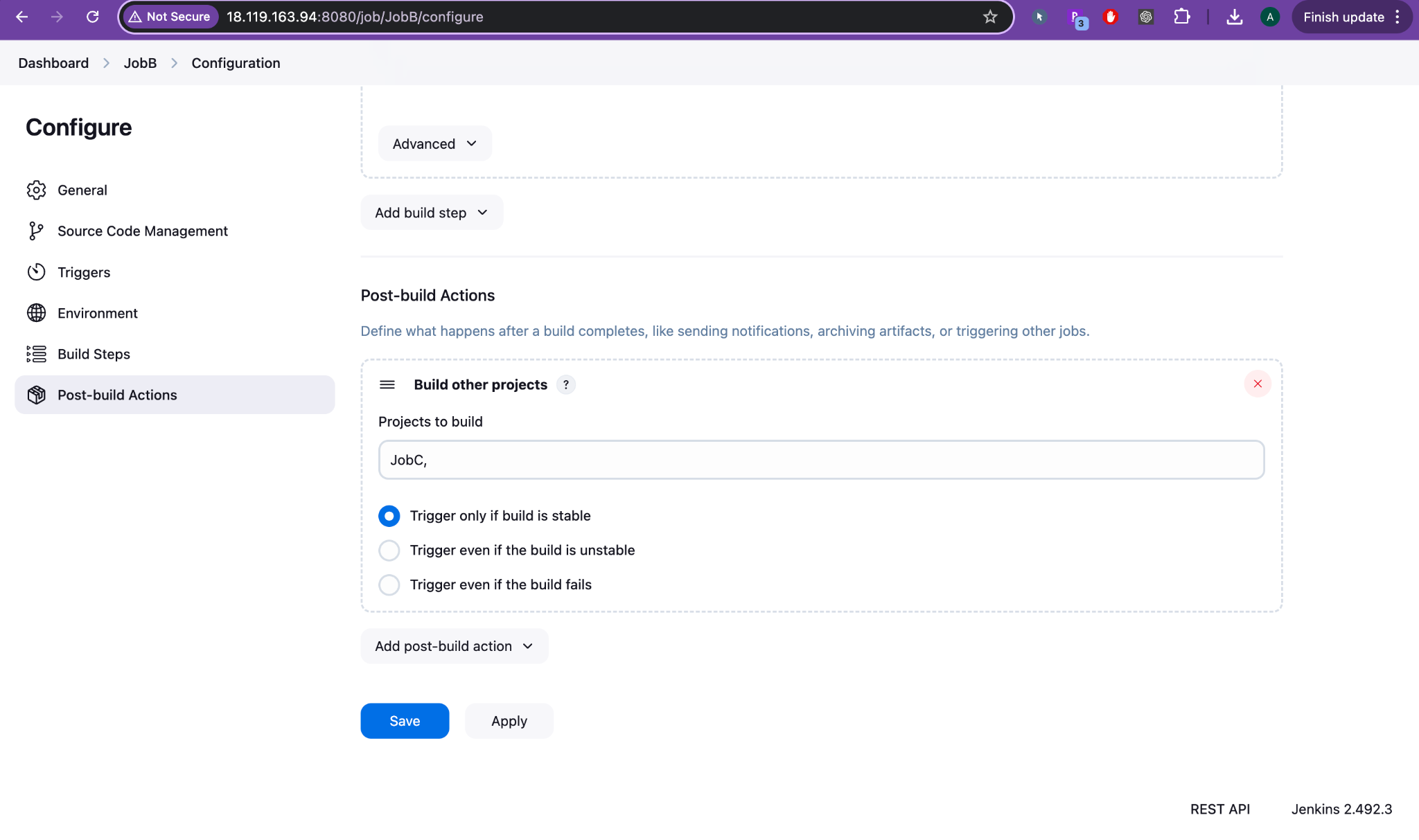
#### **🔧 What Happens:**



* JobB is configured as a **downstream job** in JobA’s "Post-build Actions."
* Once JobA finishes, Jenkins **automatically triggers JobB**.

#### **JobB Configuration:**

* Build Step: echo "This is second build"
* Post-build Action: Trigger **JobC** (next downstream job)

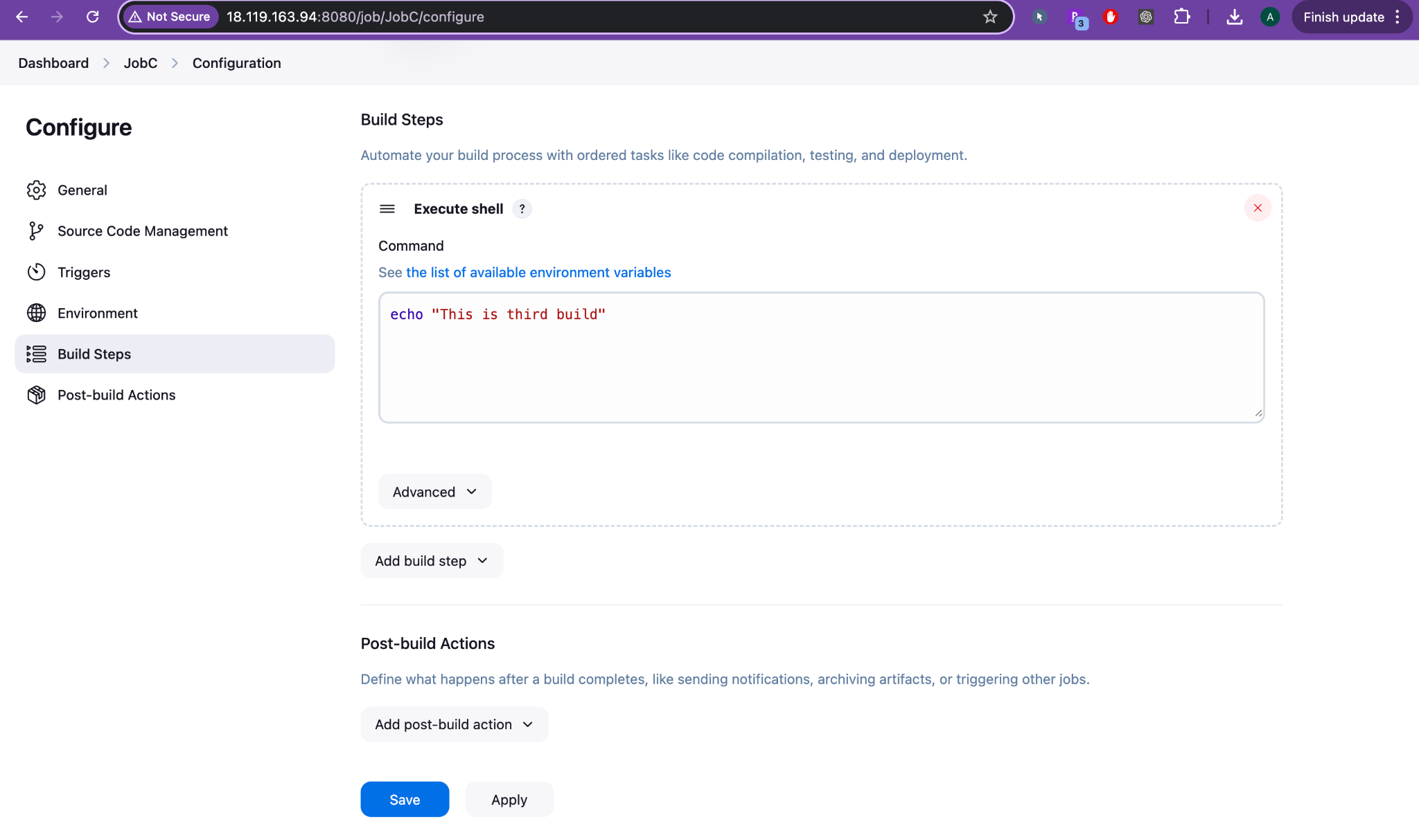


#### **✅ You See:**

* **Upstream Project**: JobA
* **Downstream Project**: JobC

### **🔸 Step 4: JobC Starts Automatically After JobB Success**

#### **🔧 What Happens:**



* Jenkins automatically triggers **JobC** after JobB is complete and stable.

#### **JobC Configuration:**

* Build Step: echo "This is third build"

#### **✅ You See:**

* **Upstream Project**: JobB
* No downstream (final job in the chain)

📊 Timeline View (Example from Console Logs)

| **Time** | **Job** | **Action** |
| --- | --- | --- |
| 05:31 AM | JobA | Triggered (via GitHub or manual) |
| 05:31 AM | JobA | Build Success + Archived .war |
| 05:32 AM | JobB | Auto-triggered by JobA |
| 05:32 AM | JobB | Executed echo "This is second build" |
| 05:32 AM | JobC | Auto-triggered by JobB |
| 05:32 AM | JobC | Executed echo "This is third build" |

## **🔗 Job Relationship Diagram**

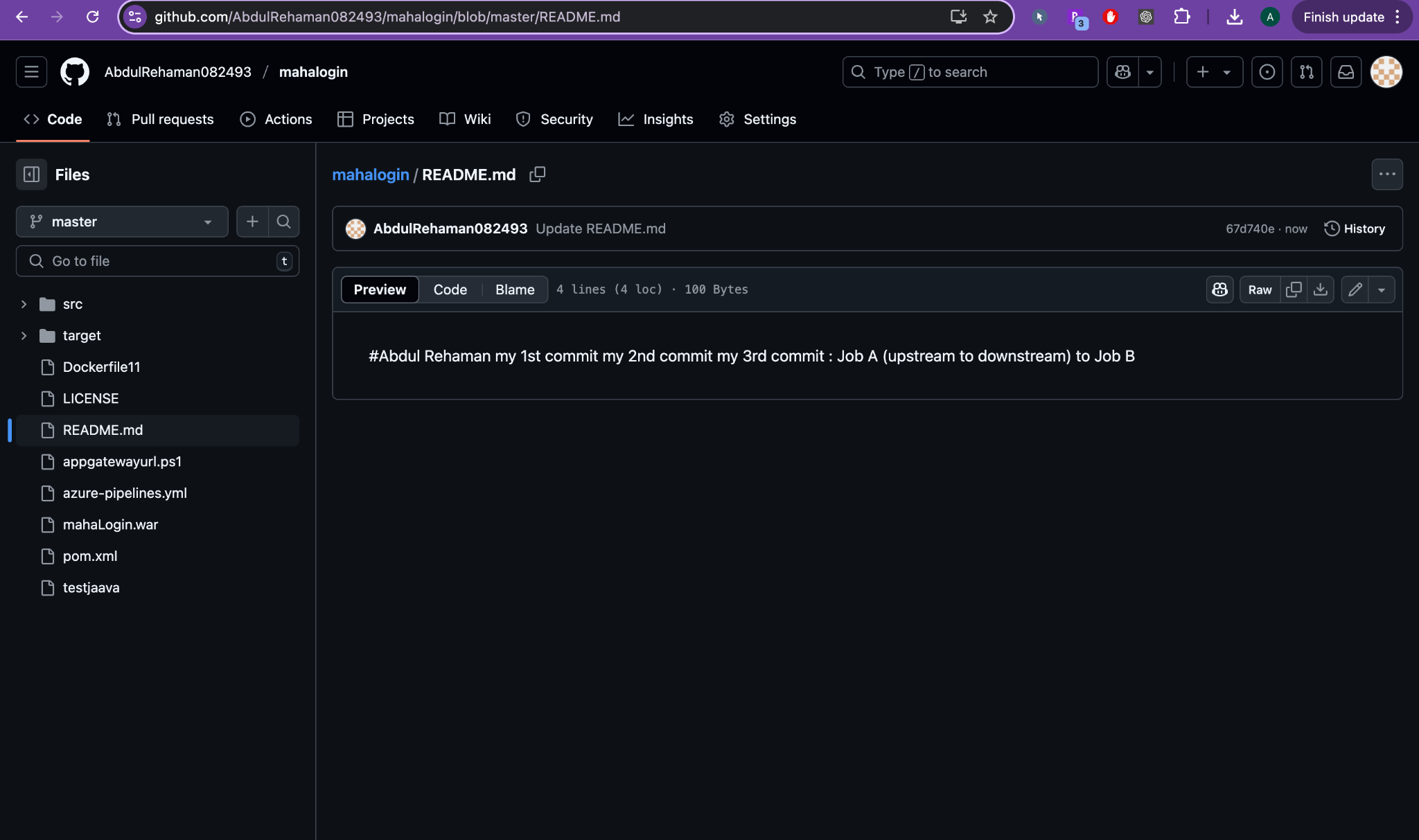
graph TD

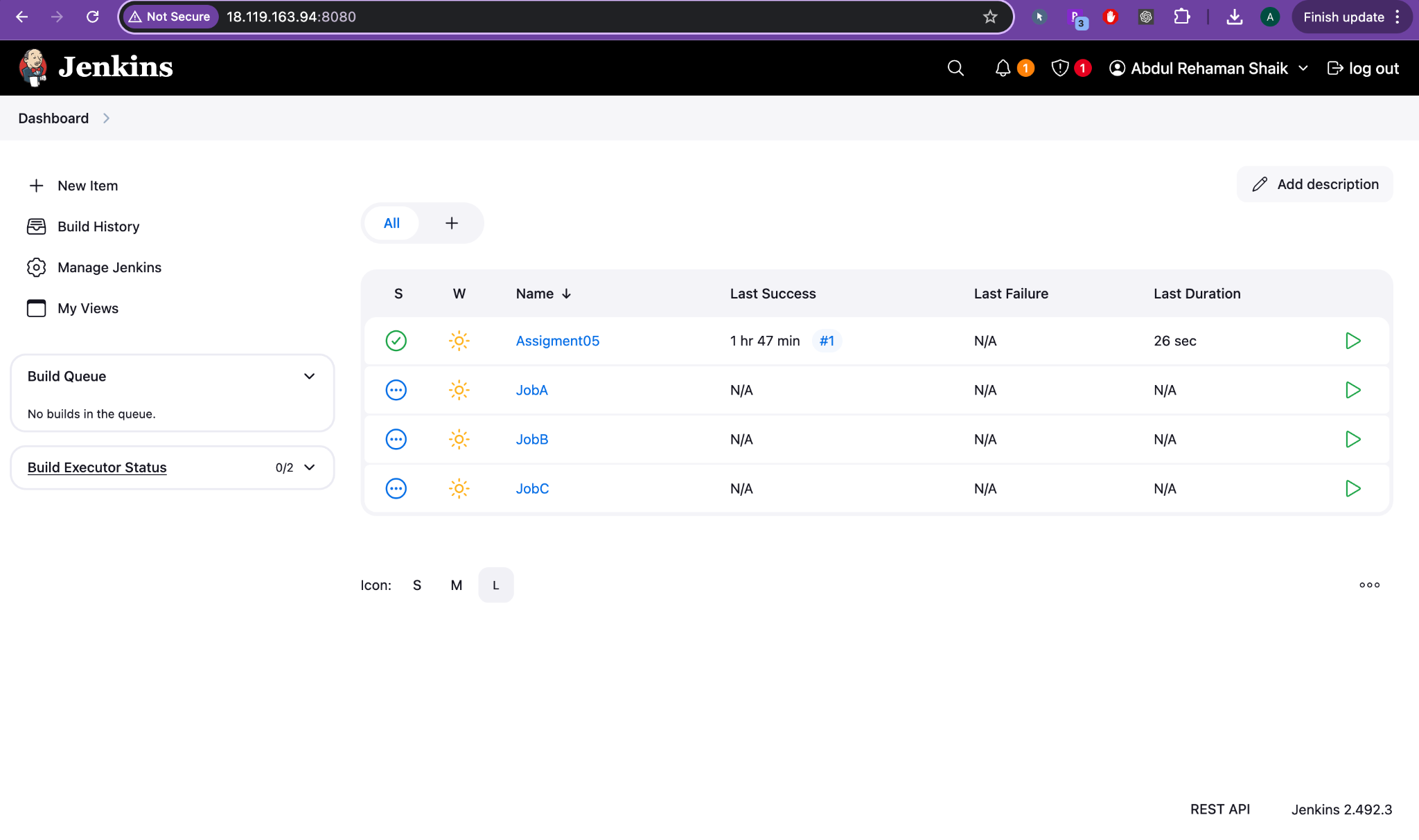
A[JobA: Git Checkout + Build + Archive] --> B[JobB: echo 2nd build]

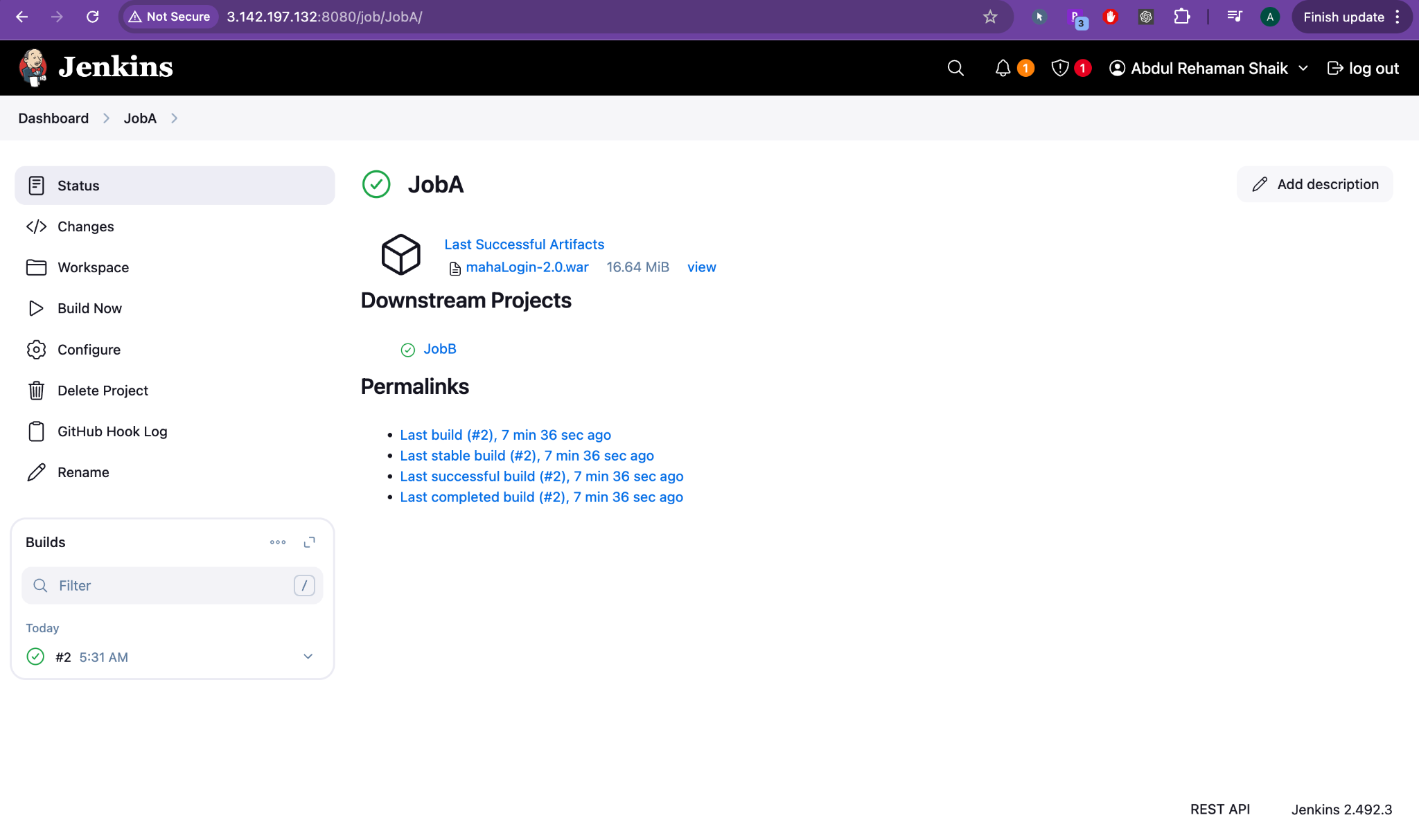
B --> C[JobC: echo 3rd build]

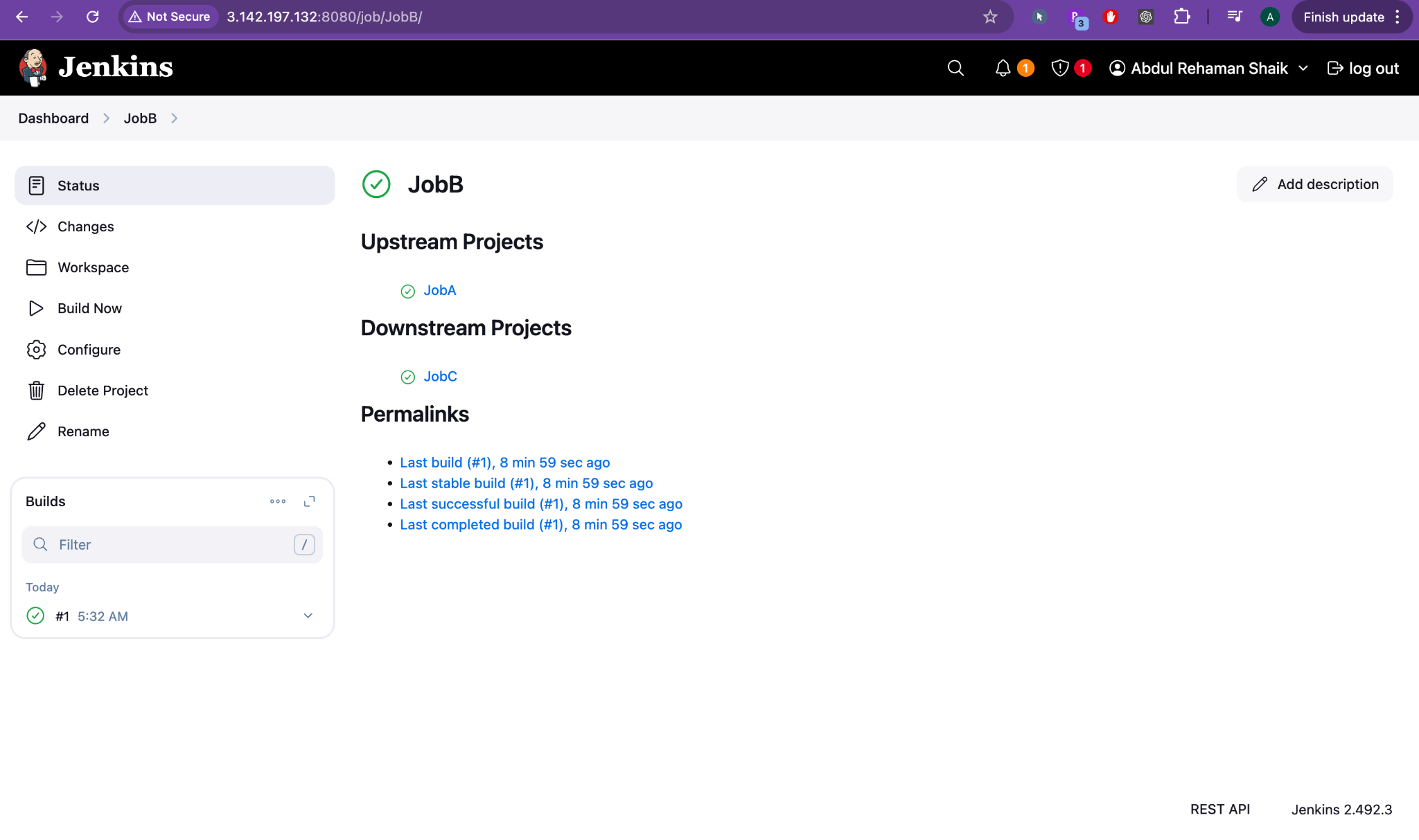
🎯 Summary of Purpose

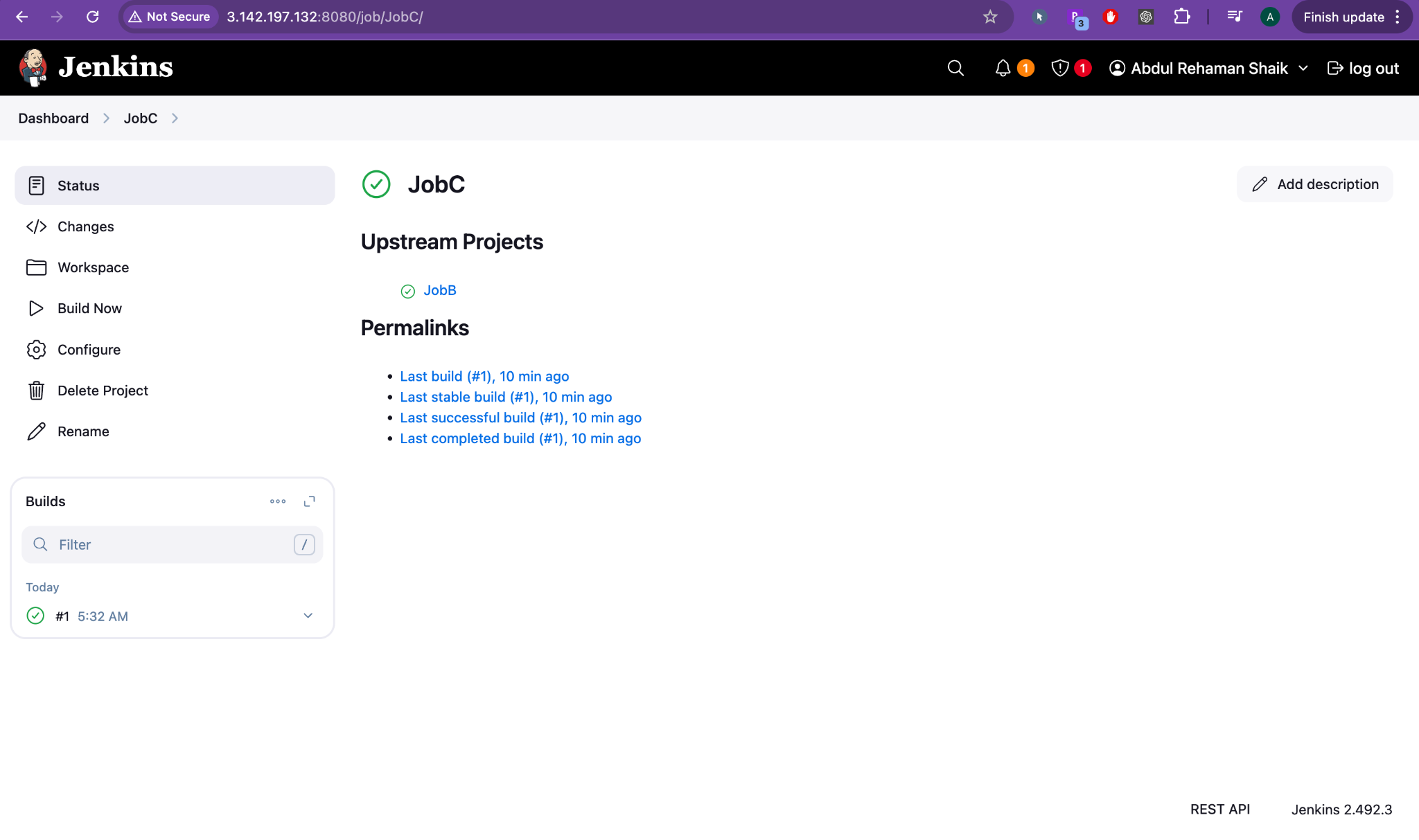
| **Job** | **Function** | **Triggers** |
| --- | --- | --- |
| **JobA** | Pull from GitHub, build .war file | GitHub webhook/manual |
| **JobB** | Confirm 2nd step in chain (echo) | Triggered by JobA |
| **JobC** | Final confirmation (echo) | Triggered by JobB |

****

****

****

****

****