CI/CD DevOPs Tools

CI – Continuous Integration

CD – Continuous Delivery/Deployment

SonarQube – Code Quality Testing Tool (SonarLint – plugin for Eclipse) Sonar Cloud

Maven LifeCycle

Pom.xml is the backbone for the Maven project

Package.json is the backbone of JS Framework (Angular/React/Vue/Ember….)

Maven is a Build, deployment, test, package, dependency & project management tool.

Build – Is a process of converting source to byte code [.java -> .class]

Package – Is a process of compressing multiple files to a single package (jar/war) mvn clean package

Clean – Is a process of deleting/removing the content of target folder

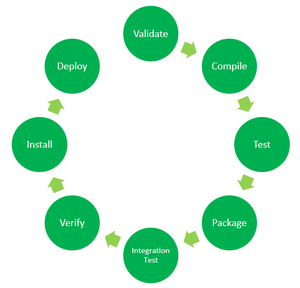
Deployment – Is a process of running the packaged code in a server (container)

|  |  |  |  |
| --- | --- | --- | --- |
|  | Source Folder | Output Folder |  |
| Core Java Project | src | bin |  |
| Dynamic Web Project | src | build |  |
| Maven Project | src | target |  |

Maven Goals – build, test, install, clean, package

* **mvn compile:** This command is used to compile the project’s source code.
* **mvn clean:** Here, the project is cleaned to remove all previous-build files generated.
* **mvn test:** With this command, one can run project testing steps.
* **mvn test-compile:** This command is used to compile the code from the test source.
* **mvn install:** This command helps deploys the packaged WAR/JAR files storing them as classes in the local repository.
* **mvn package:** With this Maven lifecycle command, one packages or creates a project WAR or JAR file to be able to use a distributable format.
* **mvn deploy:** The deploy command occurs after compilation, running project tests and project building. Here the packaged WAR/JAR files are copied to the remote repository for use by other developers.

**Maven Lifecycle:**Below is a representation of the default Maven lifecycle and its 8 steps: Validate, Compile, Test, Package, Integration test, Verify, Install and Deploy.



Project Development (dev, test, UAT )

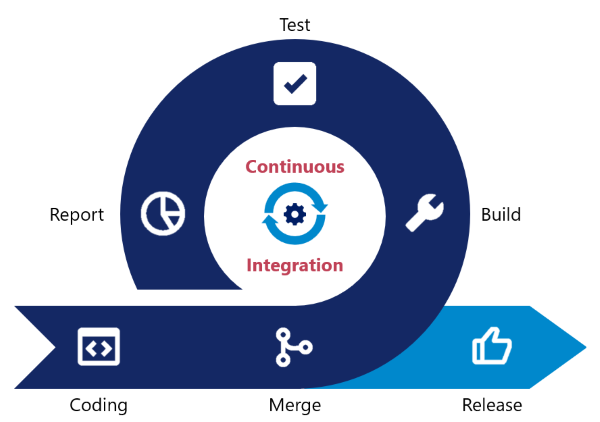
1. Design Team
2. Coding Team
3. Testing Team
4. Quality Control Team

Project Operation (pre-prod, prod)

1. Operations Team
2. Support Team
3. Maintenance Team

Dev-Ops (Development & Operations)

Jenkins, Chef, Puppet, Docker, K8S, SonarQube



**DevOps Tools - Sonar**

Sonar cloud, Sonar Qube and Sonar lint are code quality analysis tools which increase the readability, security and matainability of code. Utilizing code quality analysis tools can help development teams produce higher quality code, and provide standardization between developers, which can play a vital role in the integration of code.

**References**

* [Sonar Cloud - Documentation](https://sonarcloud.io/documentation)
* [Sonar Qube - Documentation](https://docs.sonarqube.org/latest/)
* [Sonar Lint - Github](https://github.com/SonarSource/sonarlint-eclipse)

**Code Quality Analysis and Code Smells**

Code Quality Analysis tools are programs specifically designed to expose errors as well as 'code smells'.

Code Smells

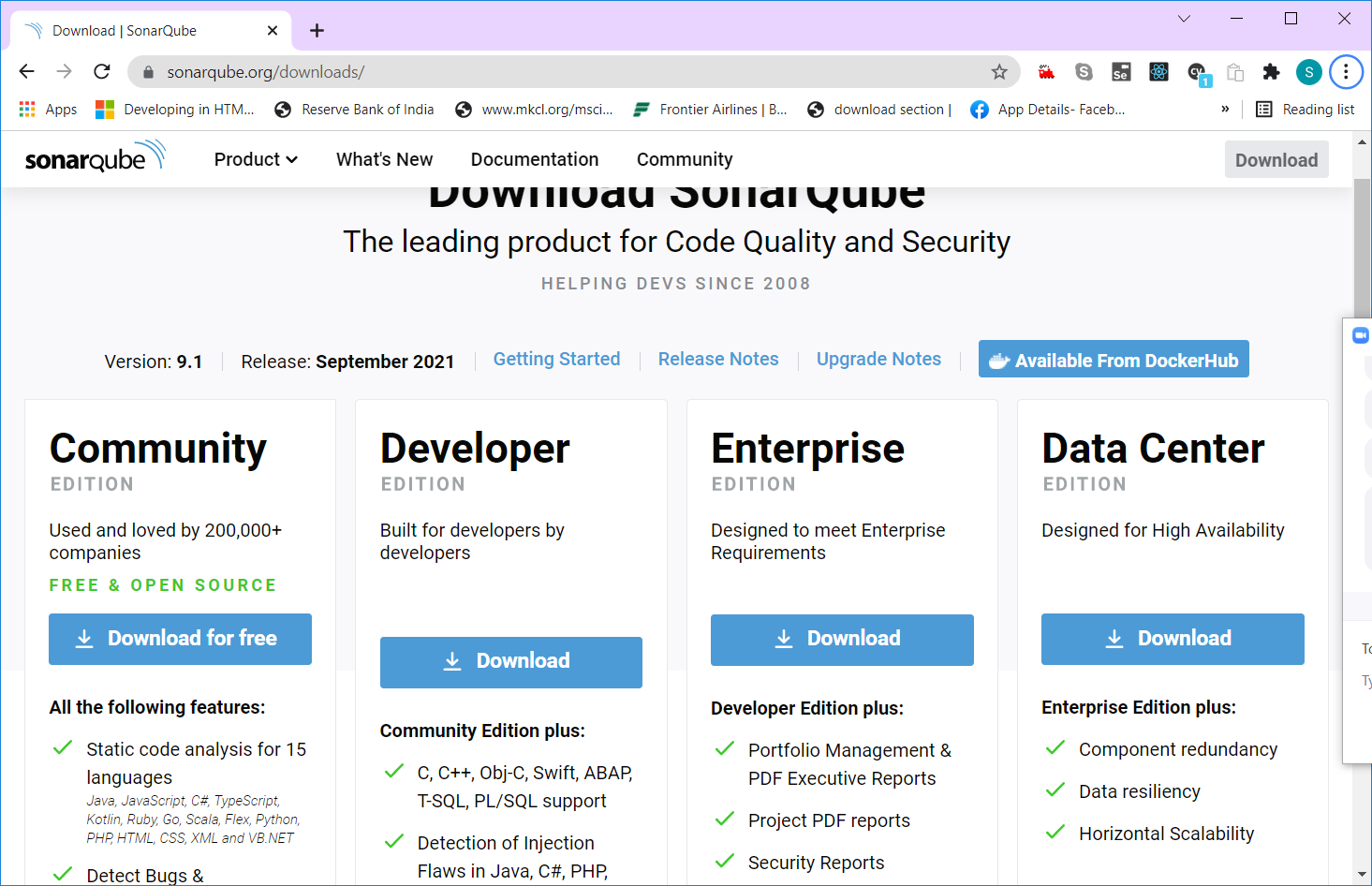
* Vulnerabilities
  + Data security issues
* Bugs
  + Issues with functionality of code
* Maintainability issues
  + Confusing or hard to maintain code
  + Repeated instances of code
  + Unused imports
  + Empty code blocks
  + Unaddressed automated code comments

**Sonar Cloud**

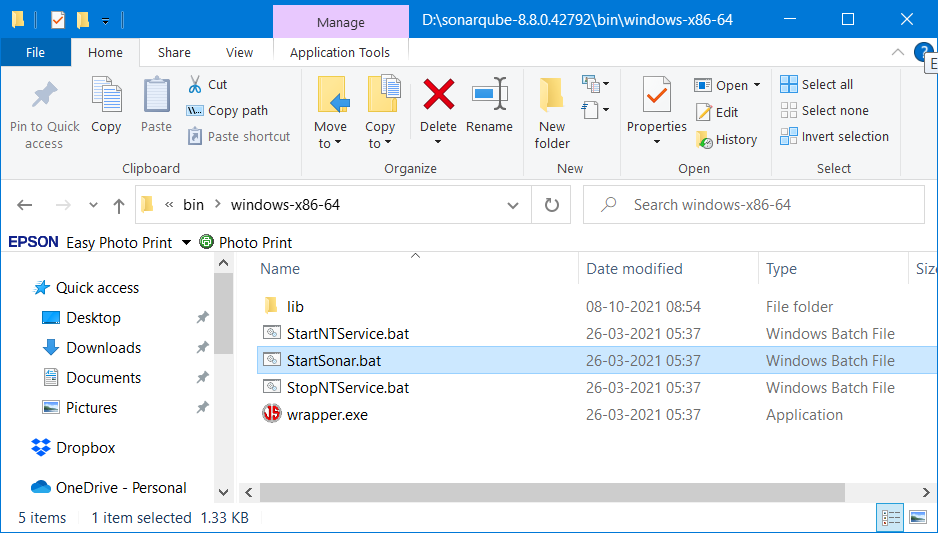
A cloud based code review solution which can be configured to review code within a cloud repositoy, such as Github.

SonarQube Download Community Edition

<https://www.sonarqube.org/downloads/>

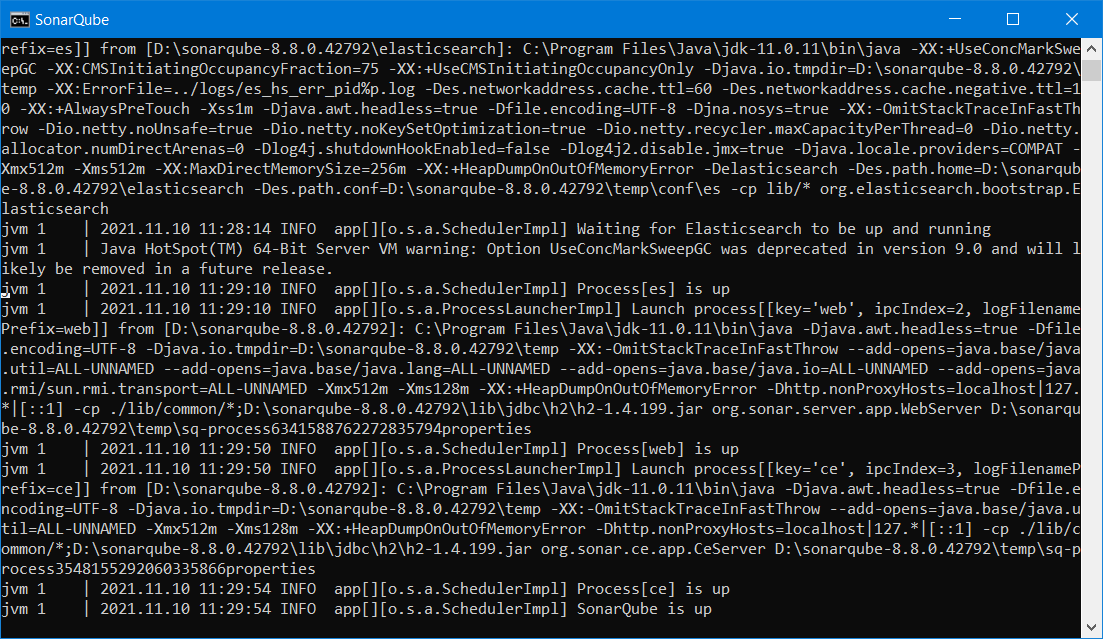


Extract the zip file and open bin folder



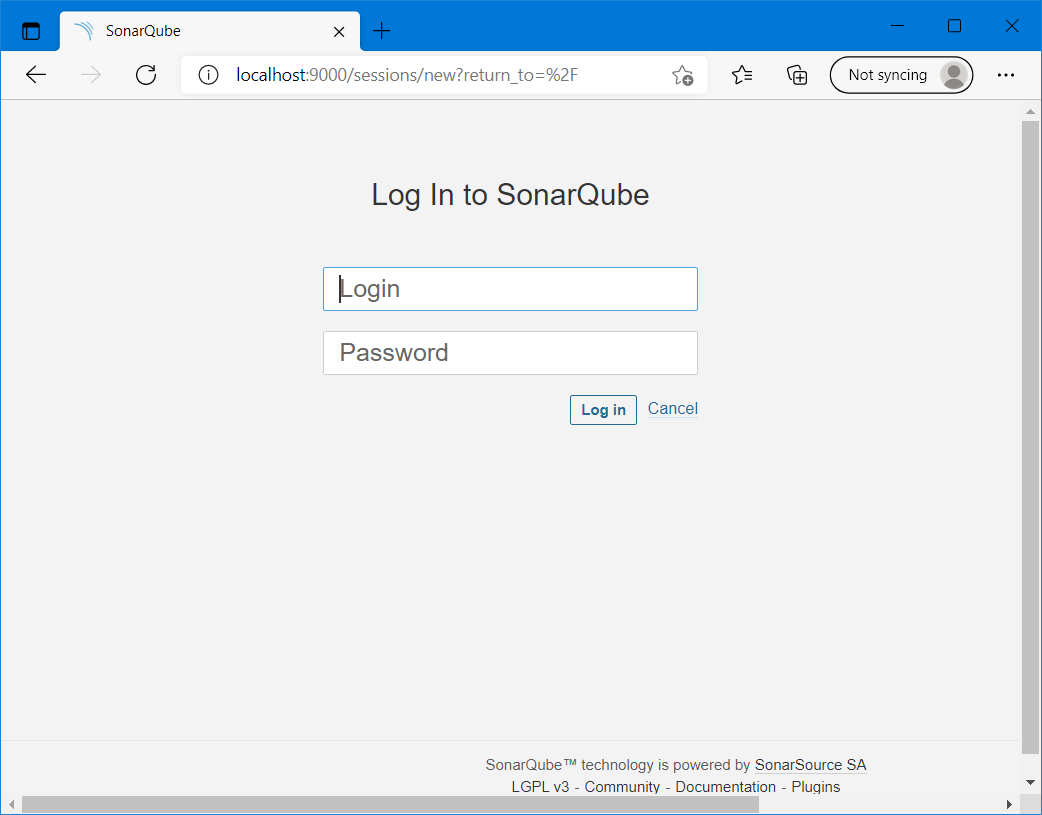
Select the folder based on the OS and double click the StartSonar.bat file

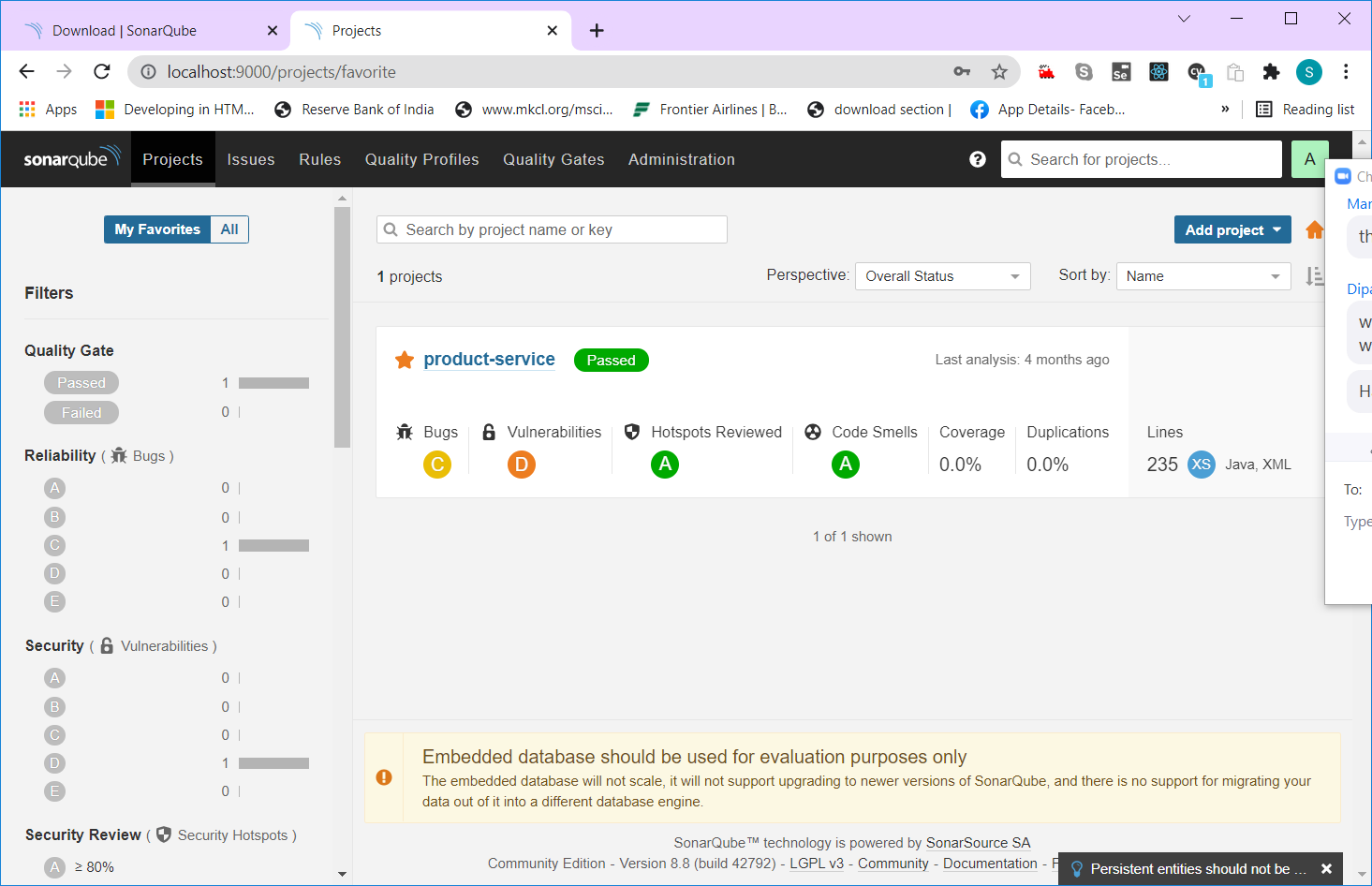
Note: SonarQube expects jdk11 in path. (Environment Variable)



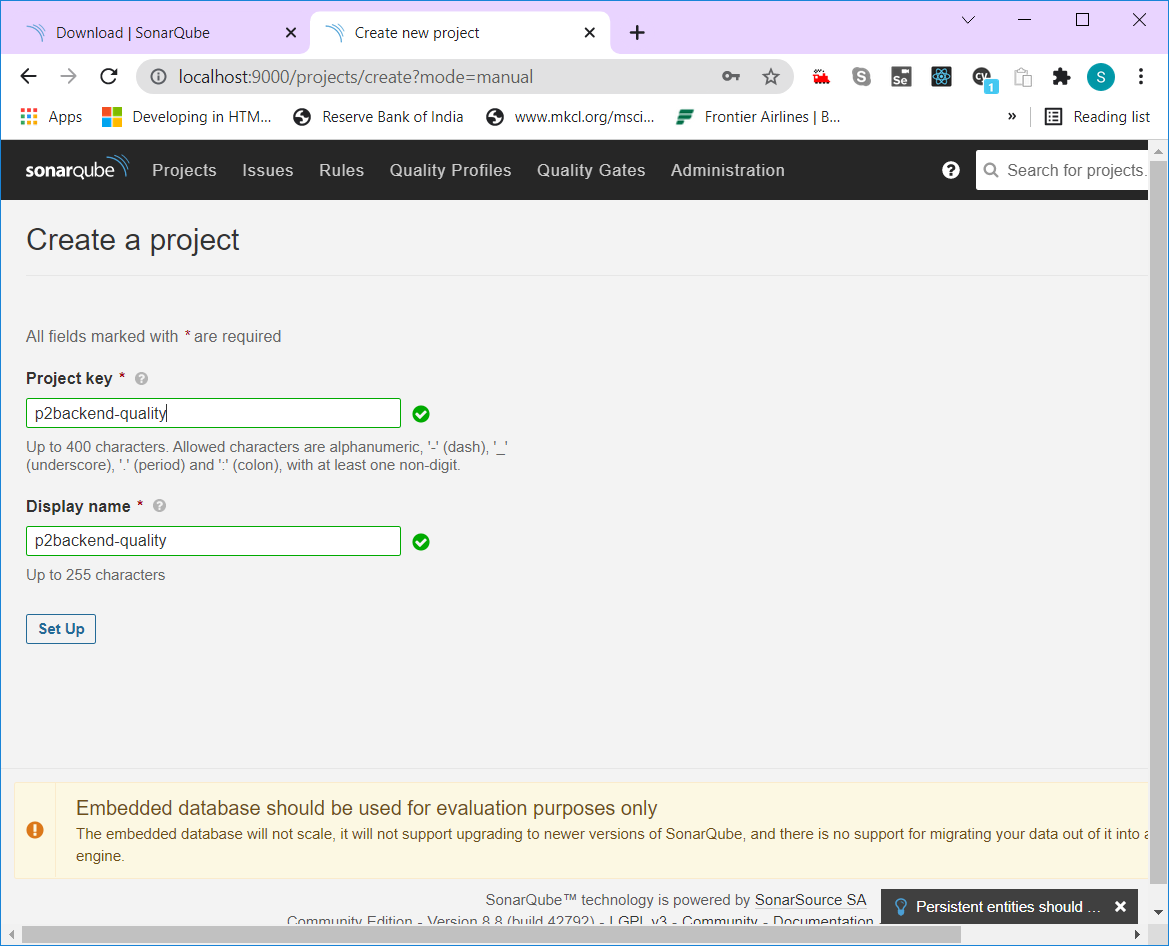
Open browser and navigate to localhost:9000

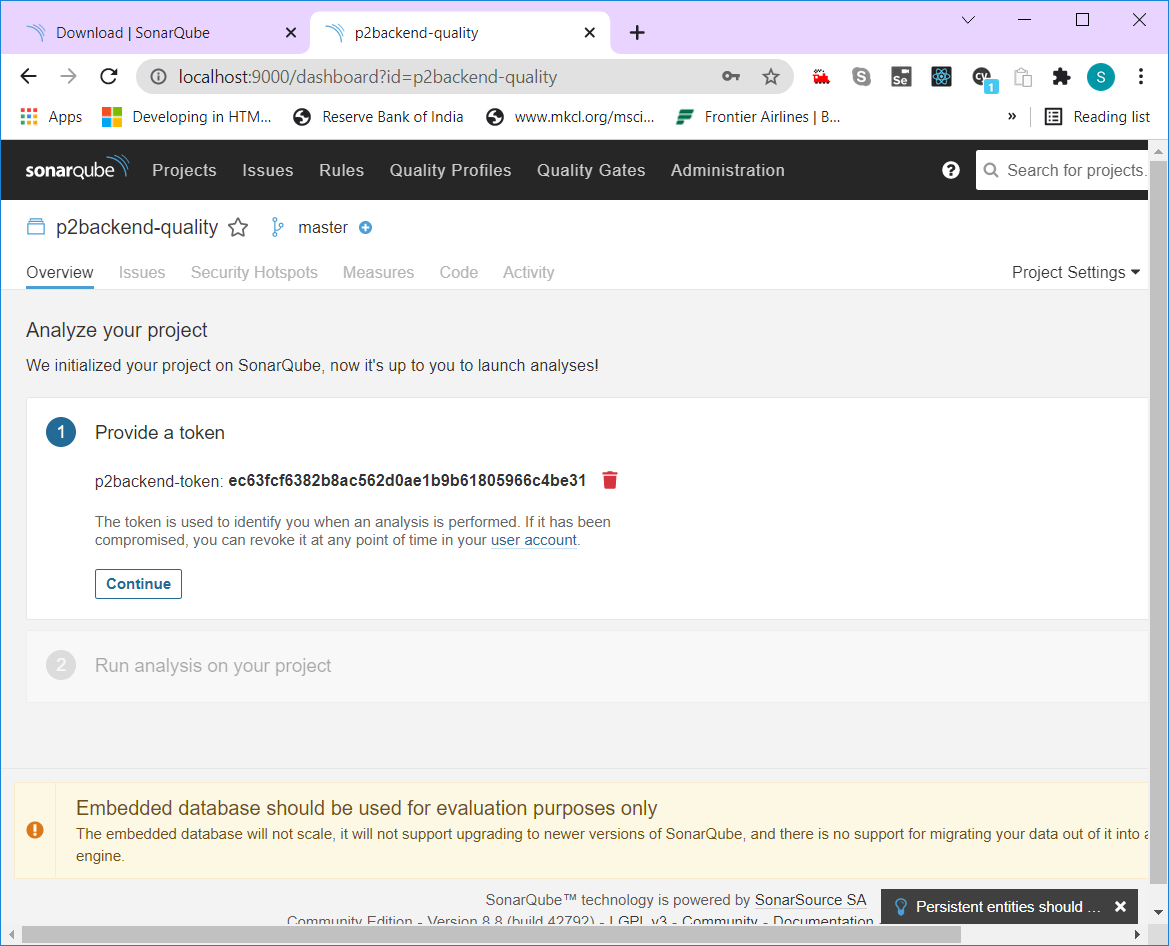
Default username and password for the sonarqube is admin/admin

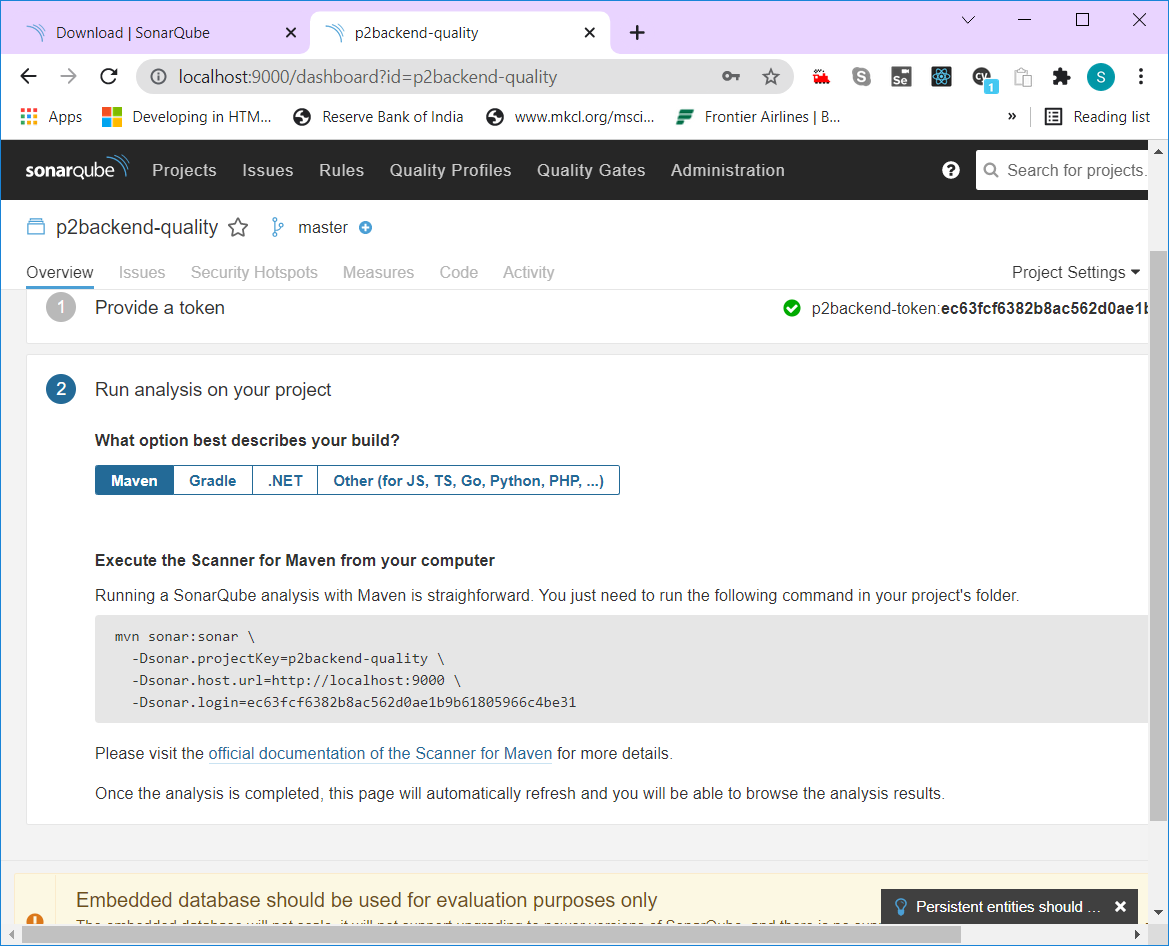




Click “Add Project” button on top right hand side and select “manully” option

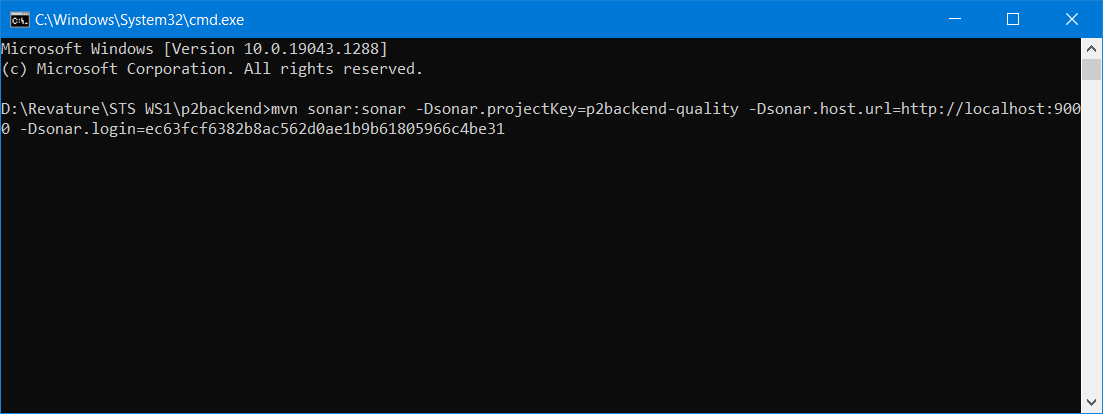






mvn sonar:sonar -Dsonar.projectKey=p2backend-quality -Dsonar.host.url=http://localhost:9000 -Dsonar.login=ec63fcf6382b8ac562d0ae1b9b61805966c4be31

Open the command prompt in the location where the p2backend project is available and paste the above maven command

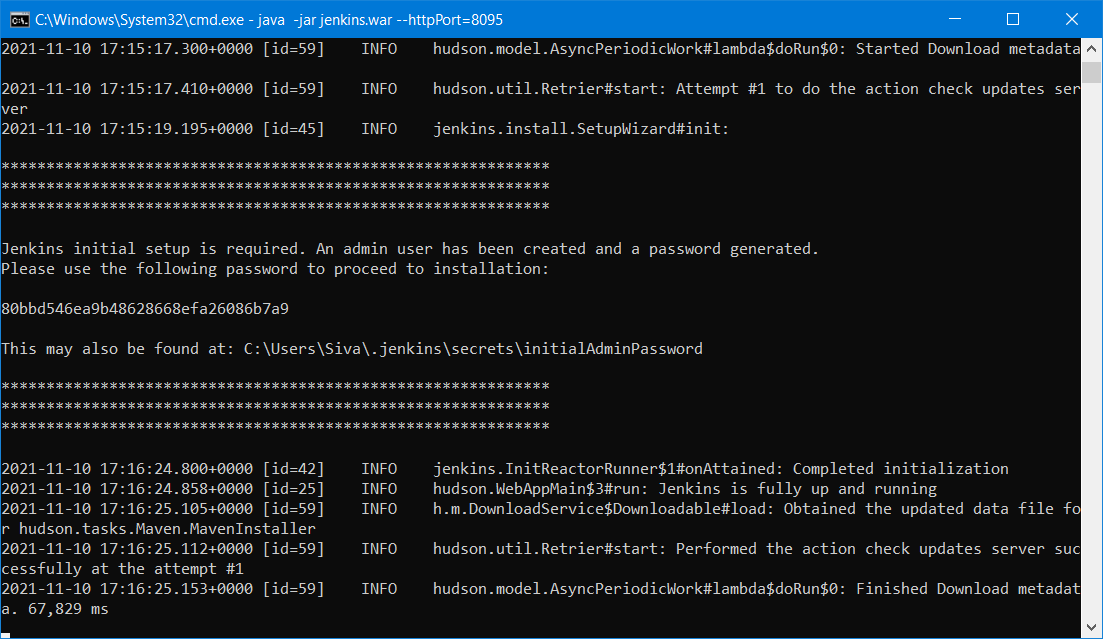


Jenkins download

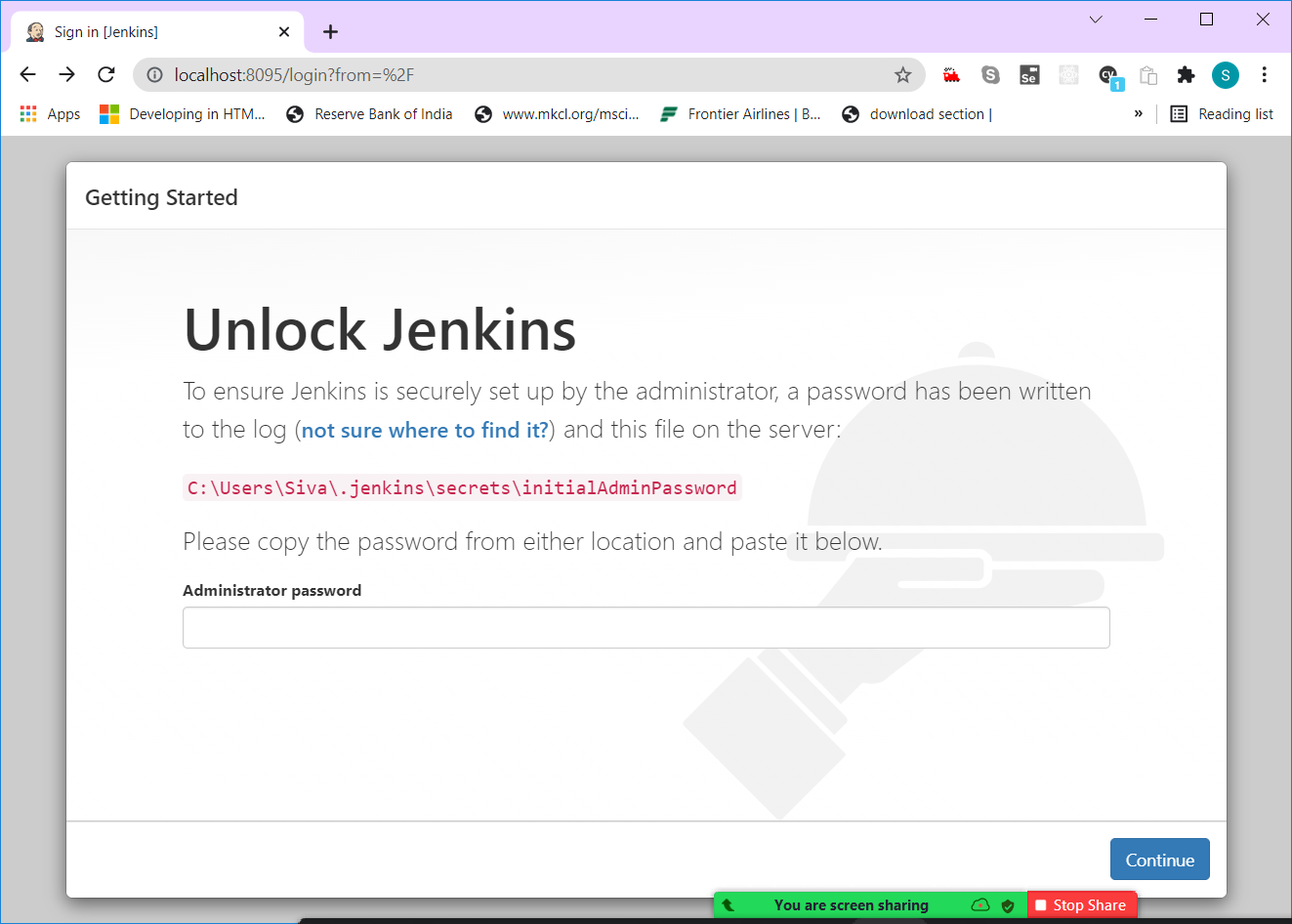
<https://www.jenkins.io/download/>

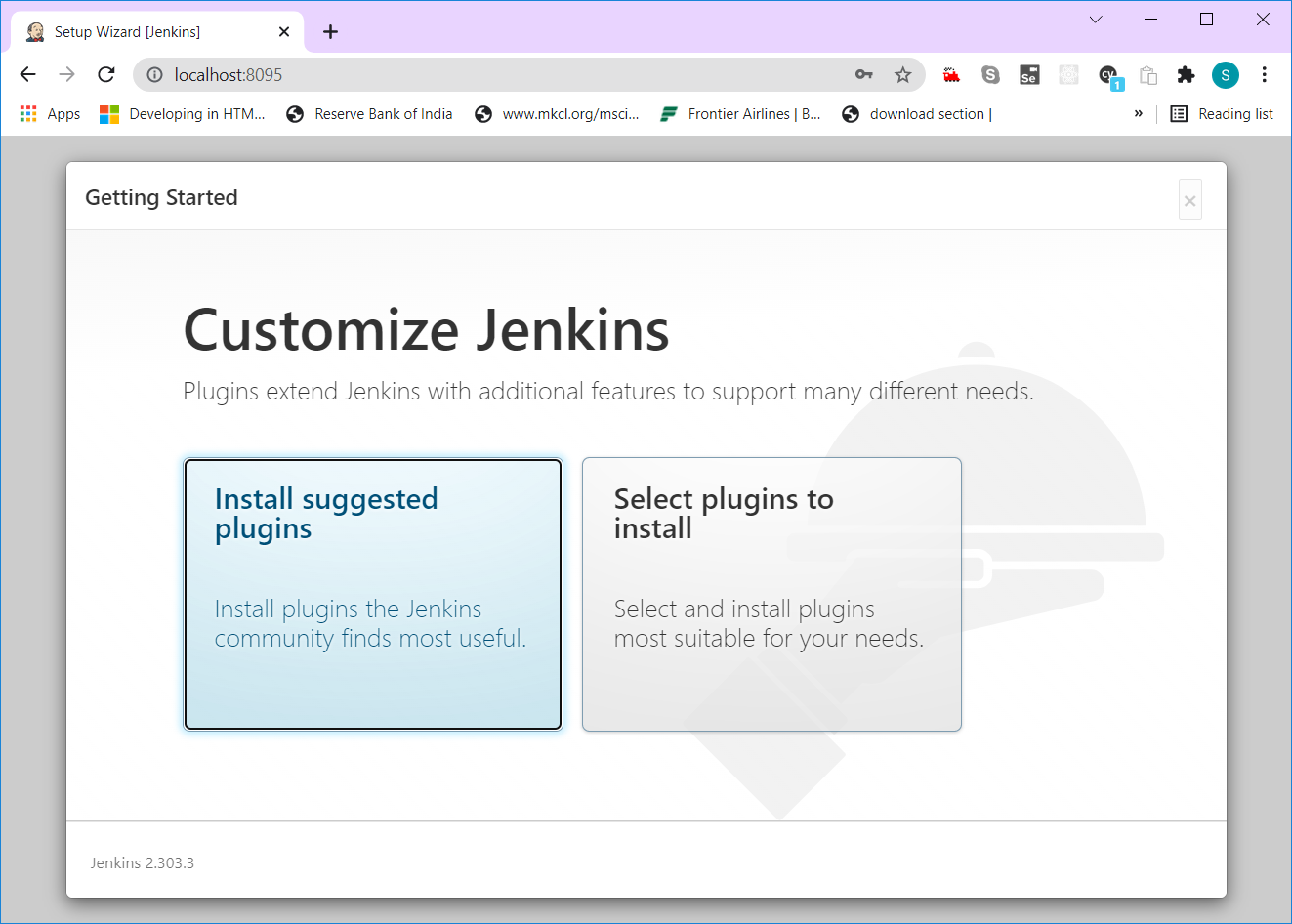
Download war & msi (Microsoft installer)

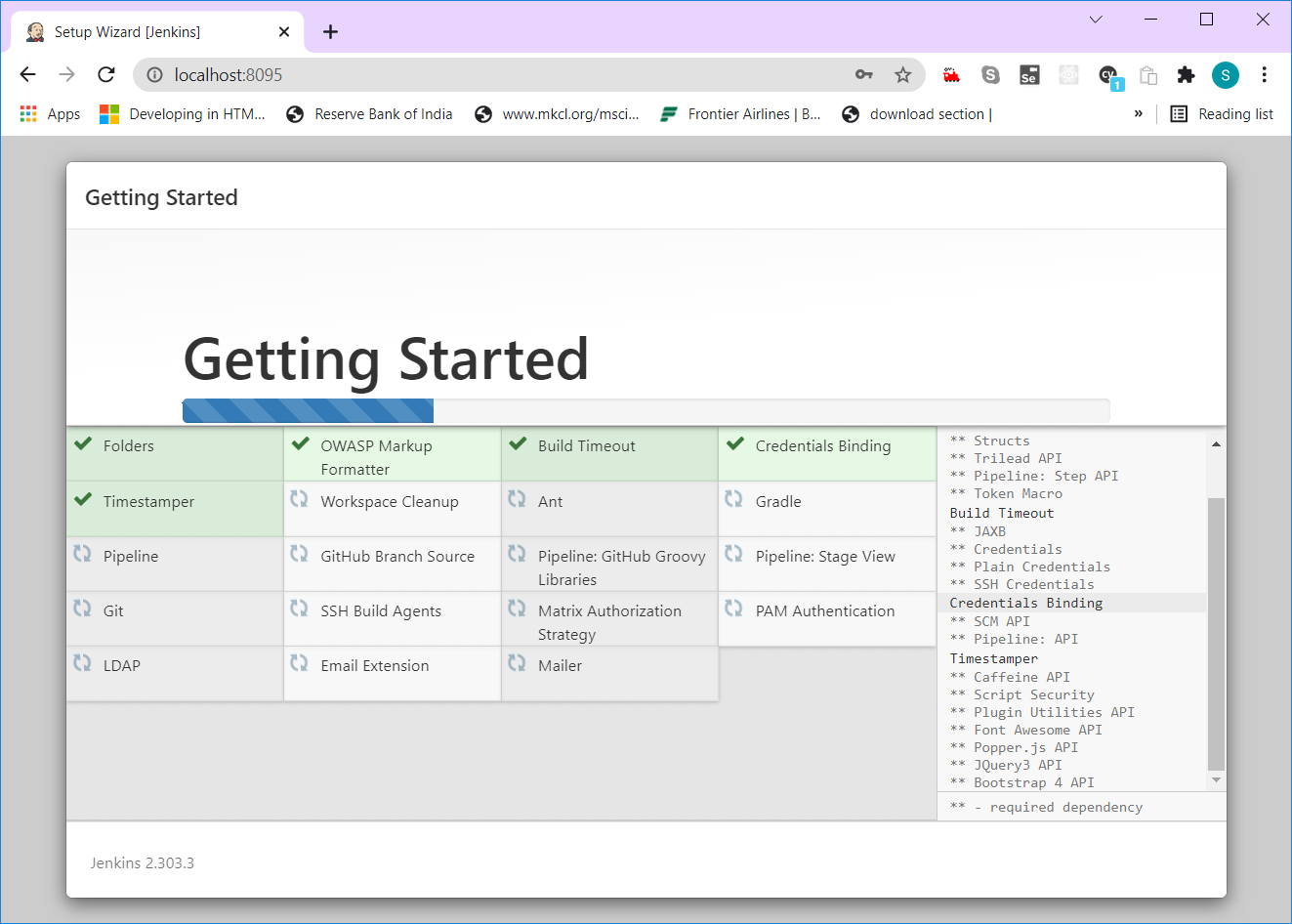
java -jar jenkins.war --httpPort=8080

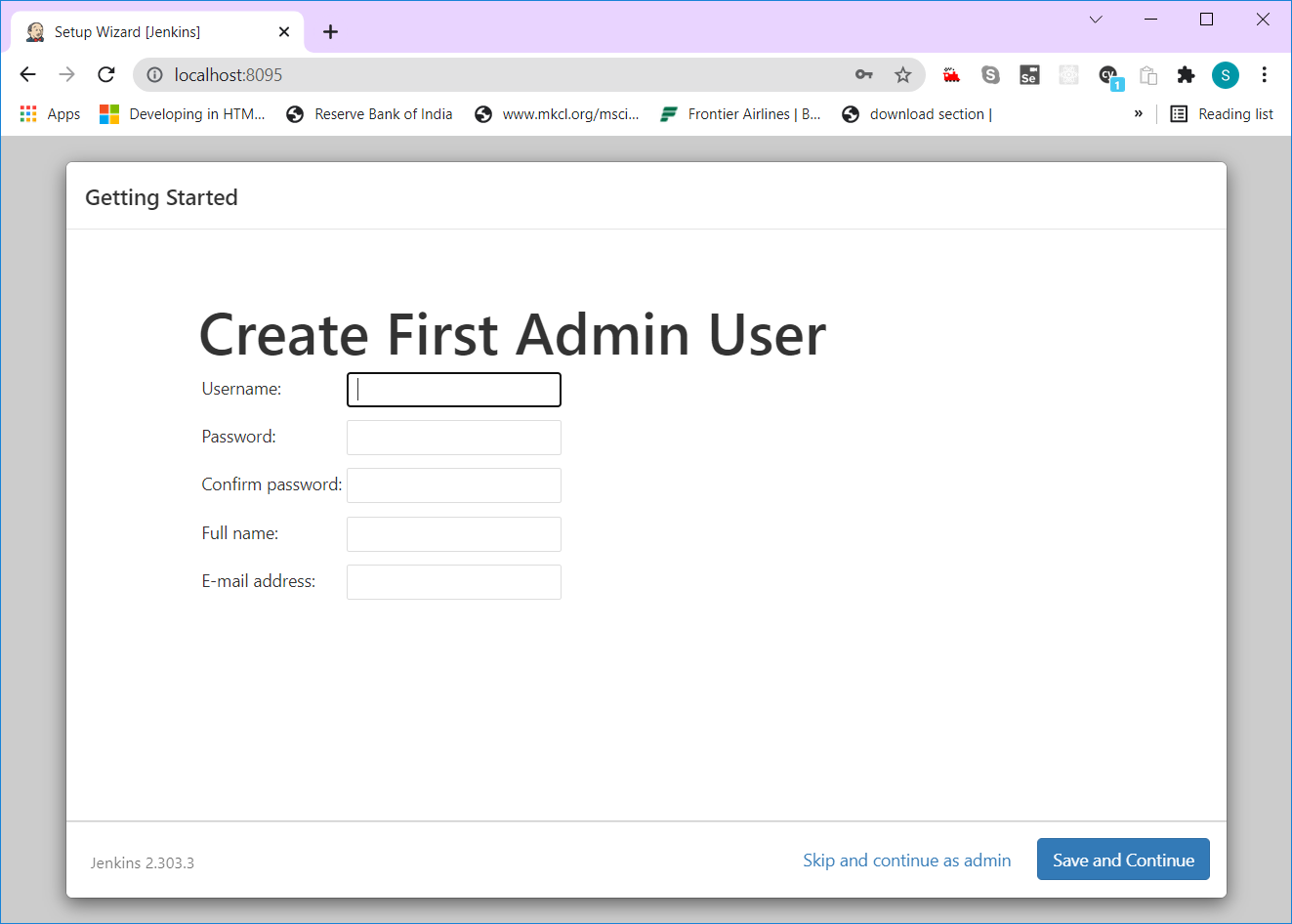


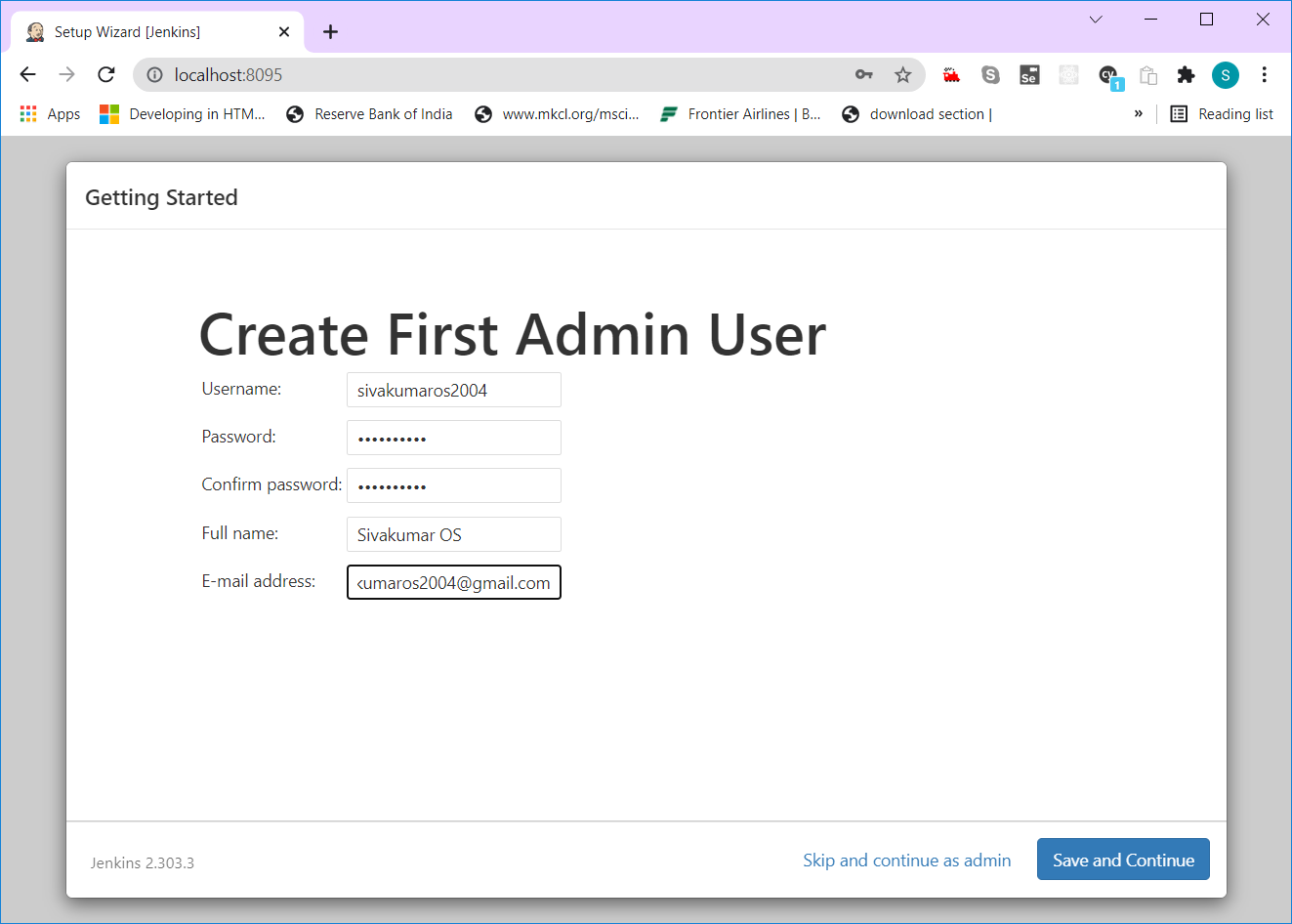
80bbd546ea9b48628668efa26086b7a9

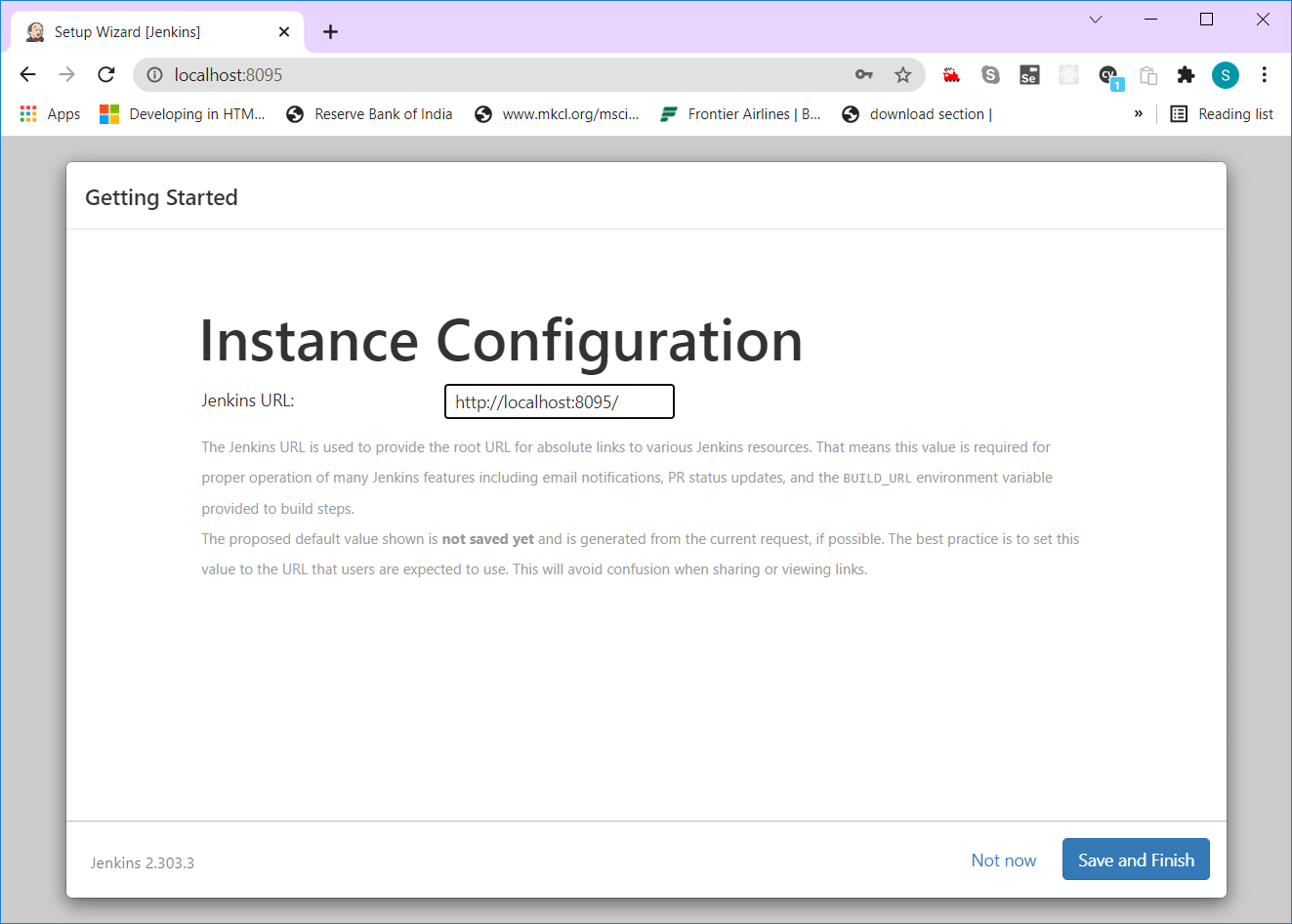


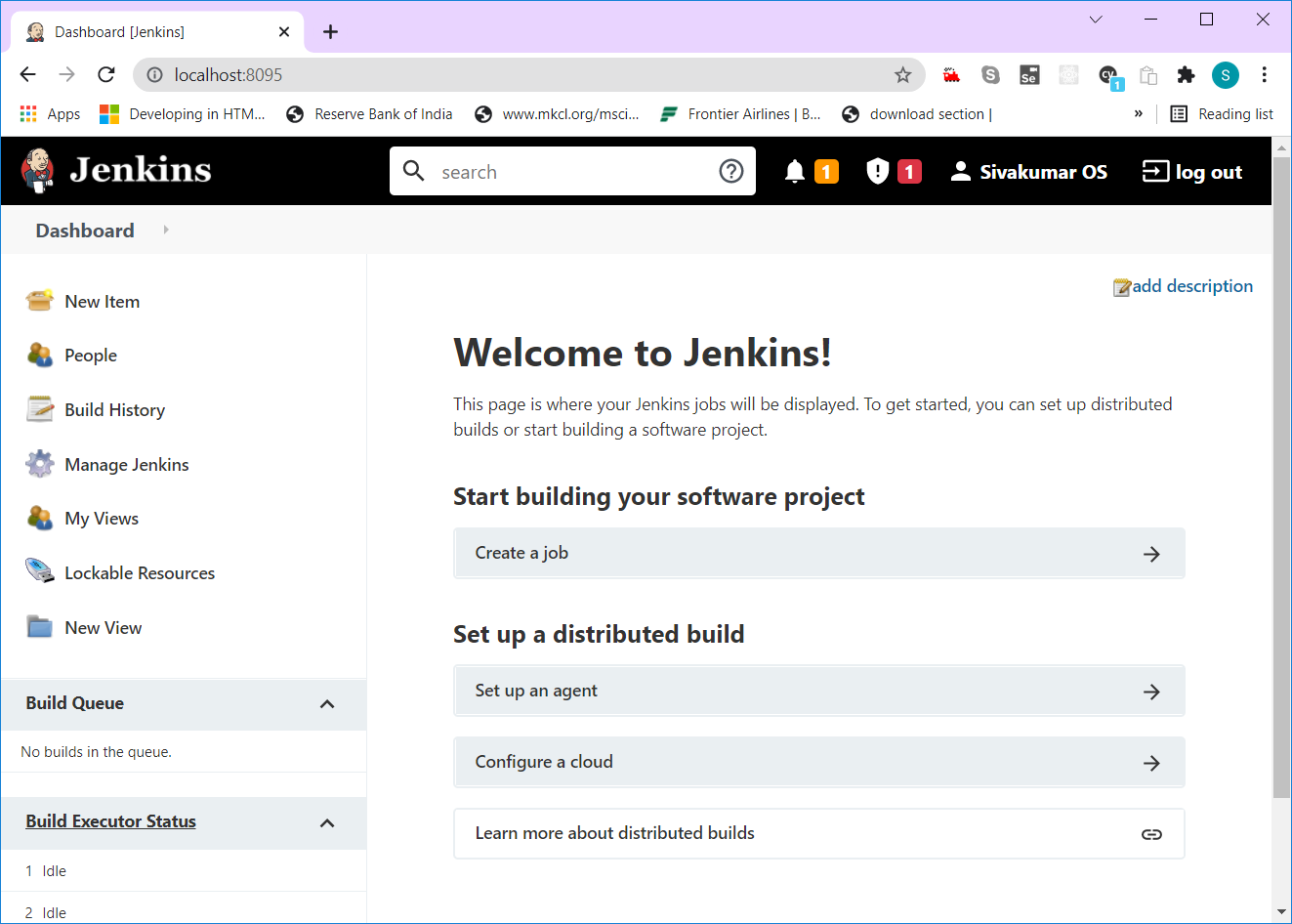


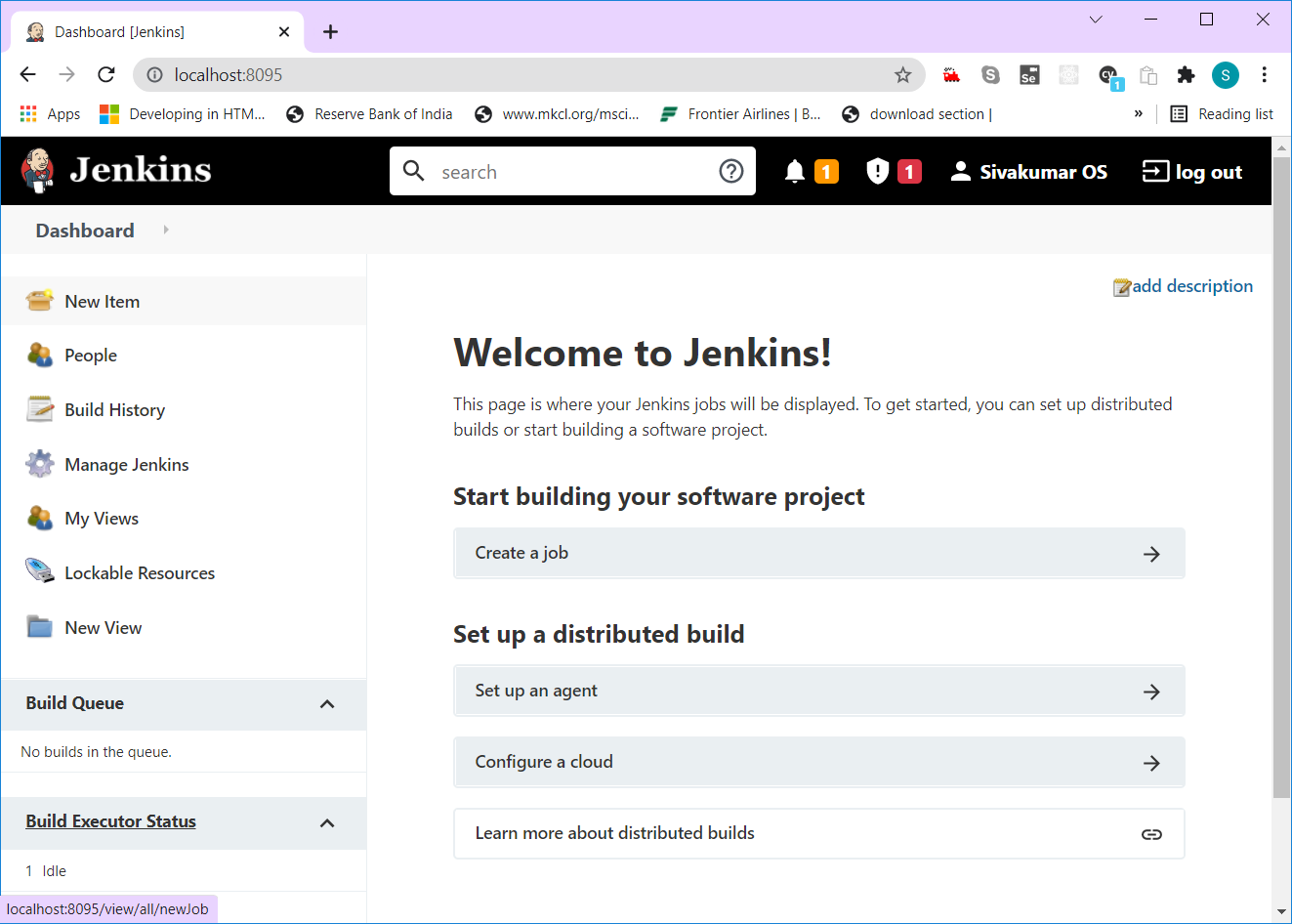


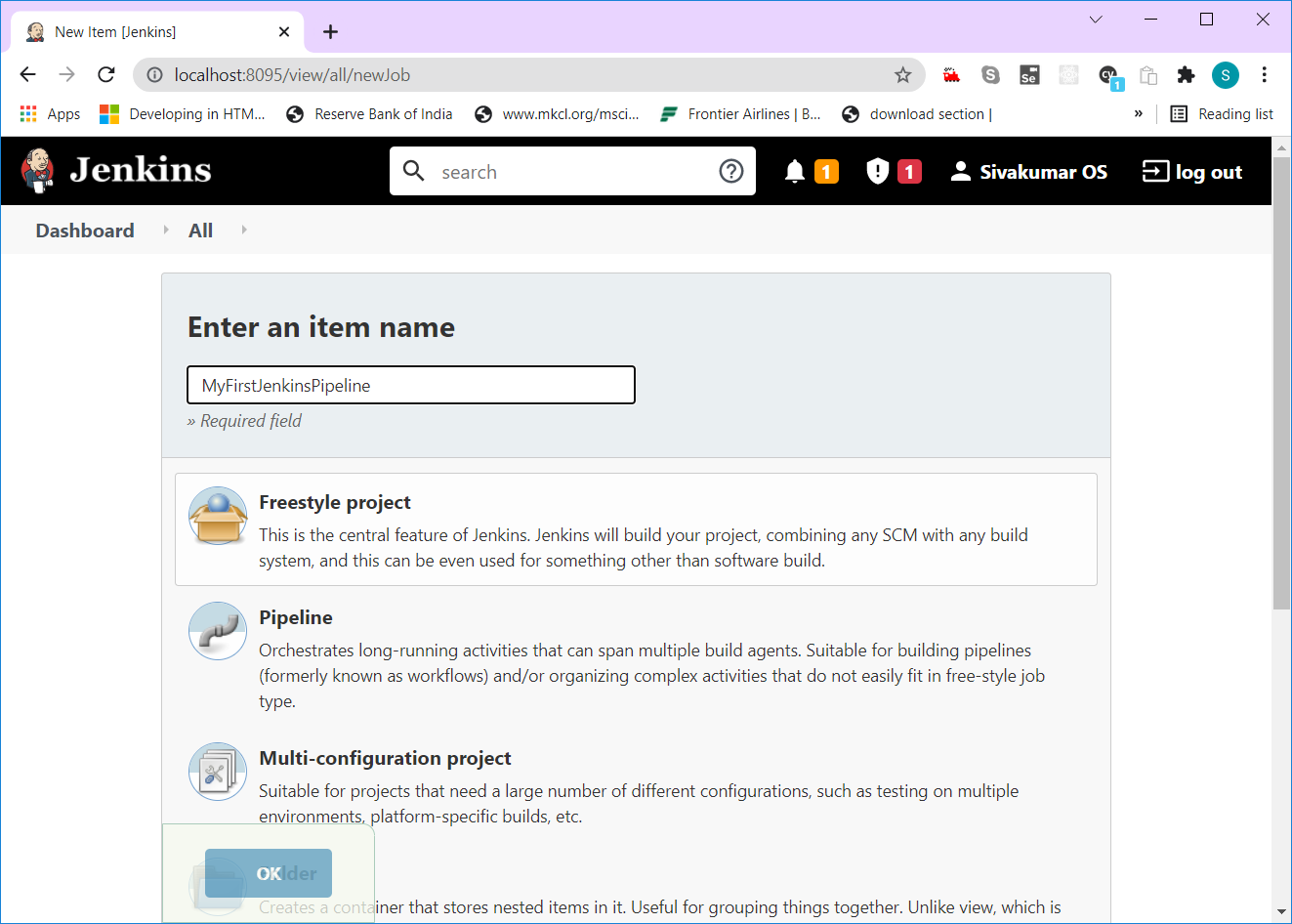




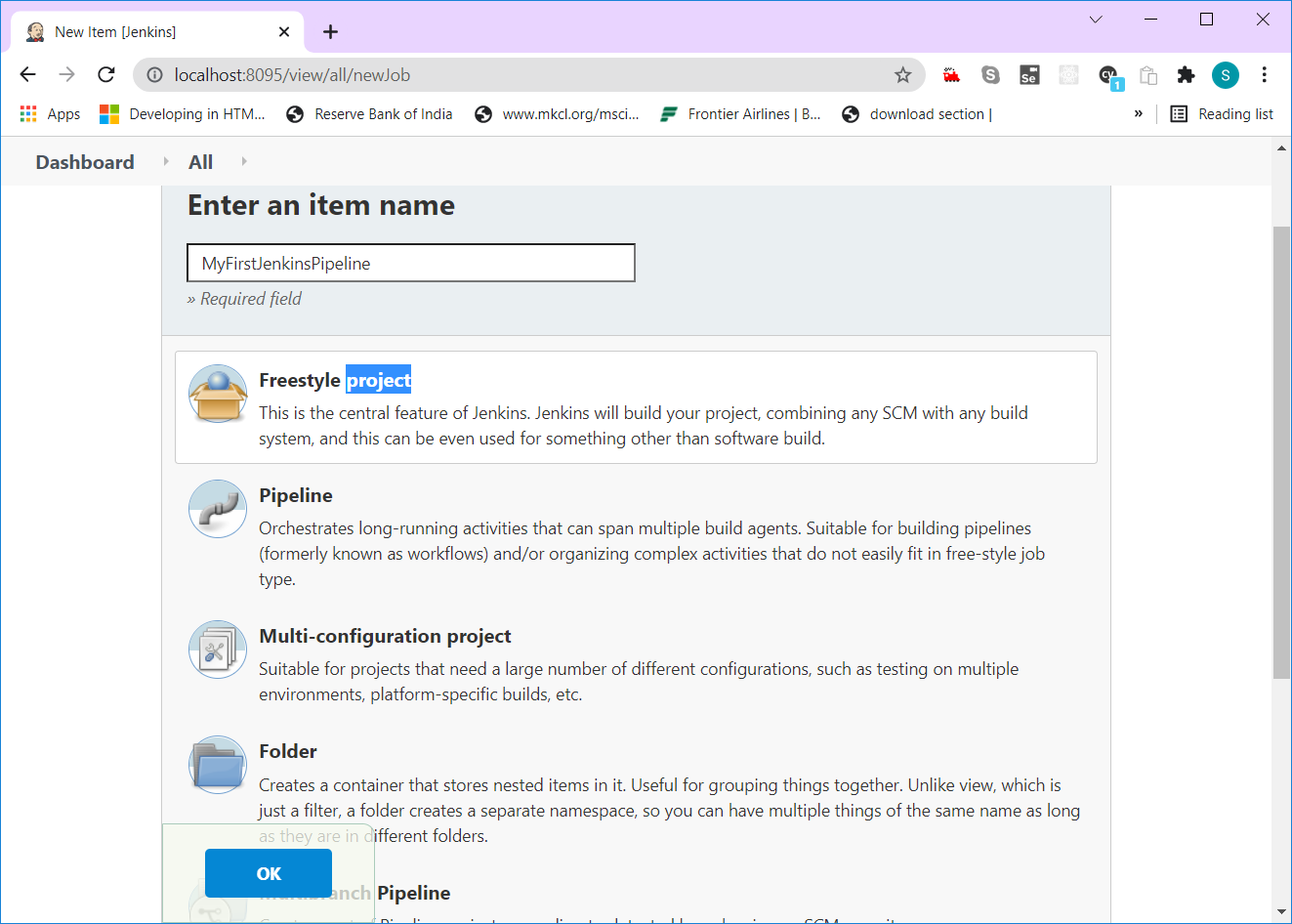


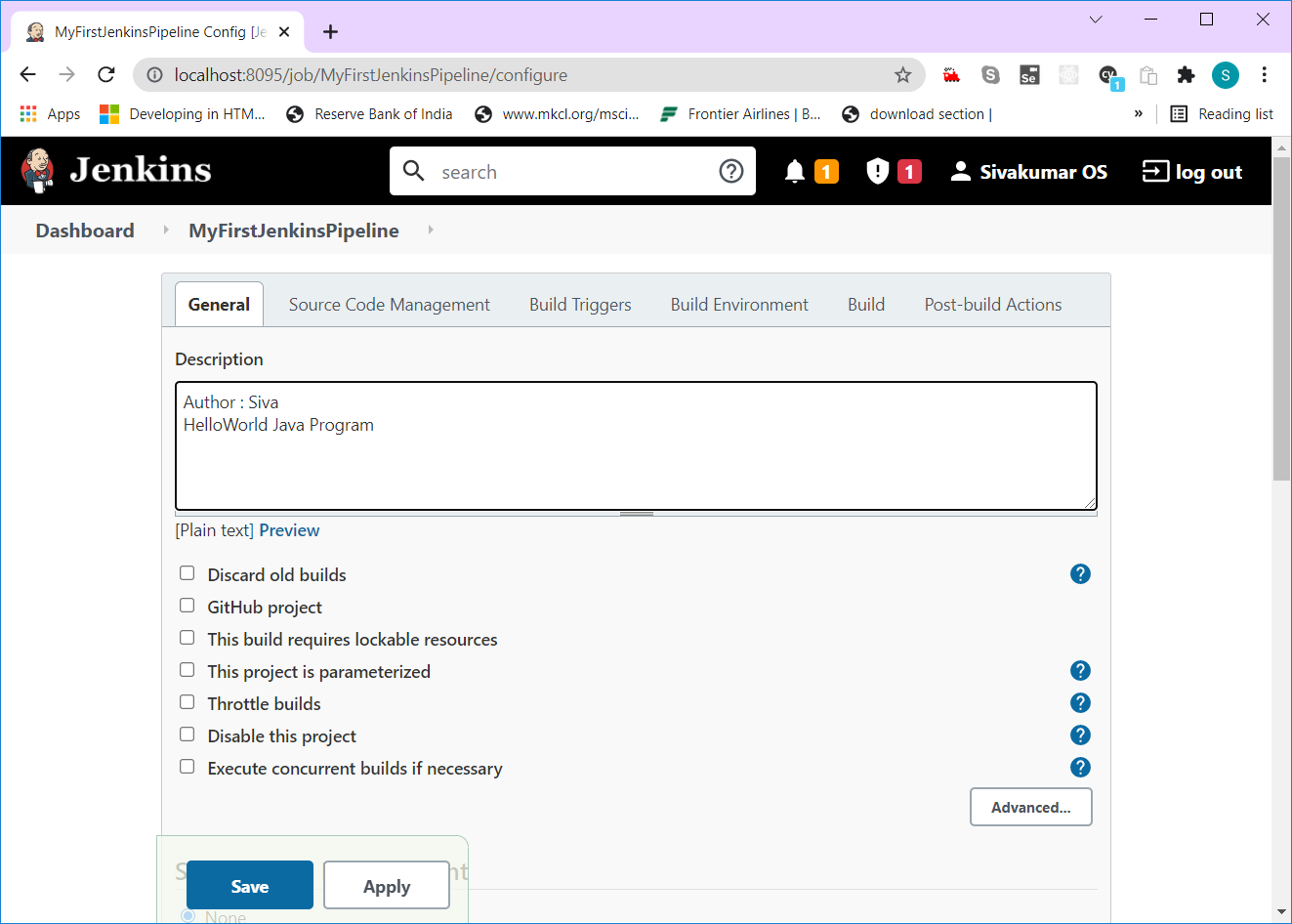




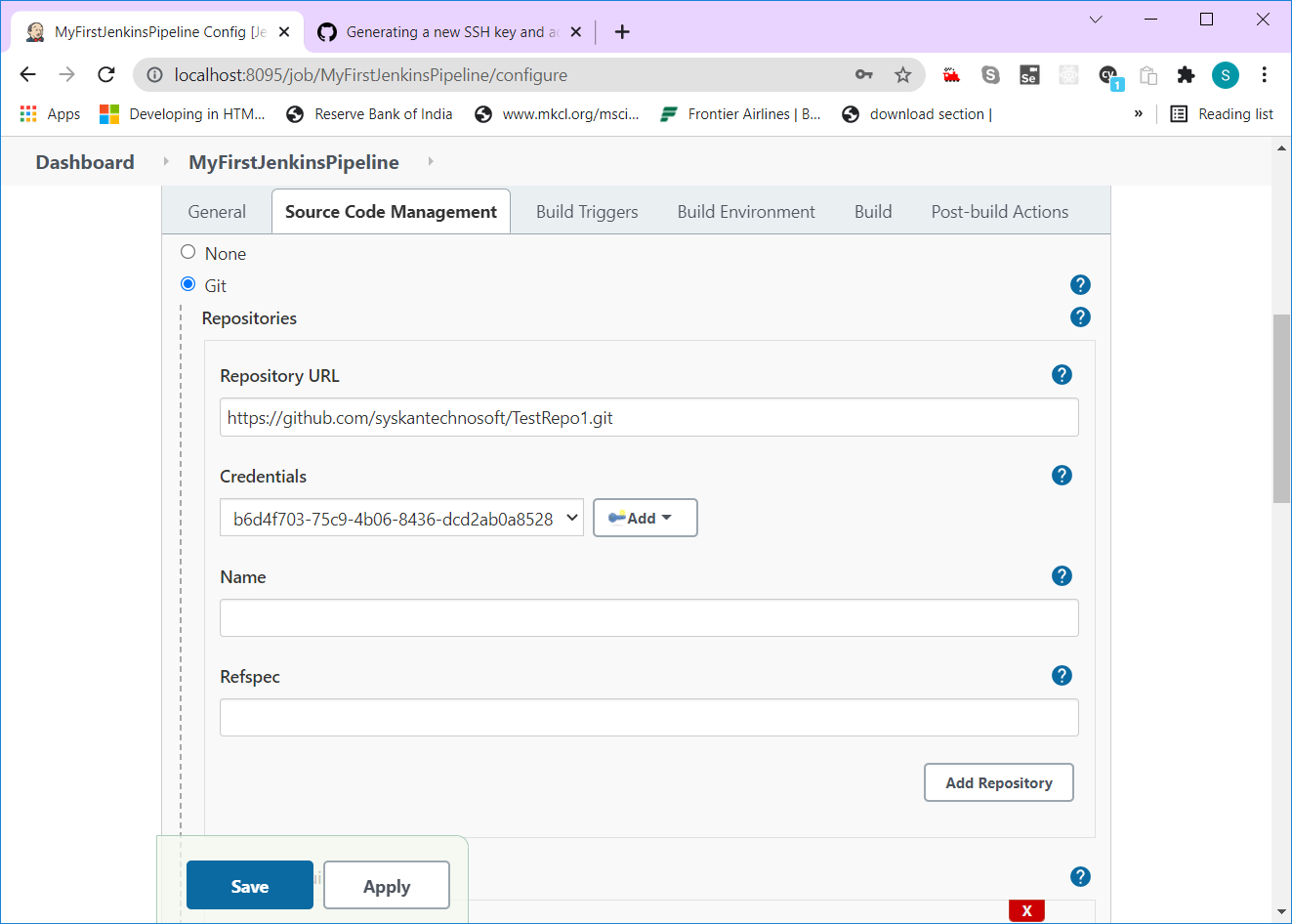


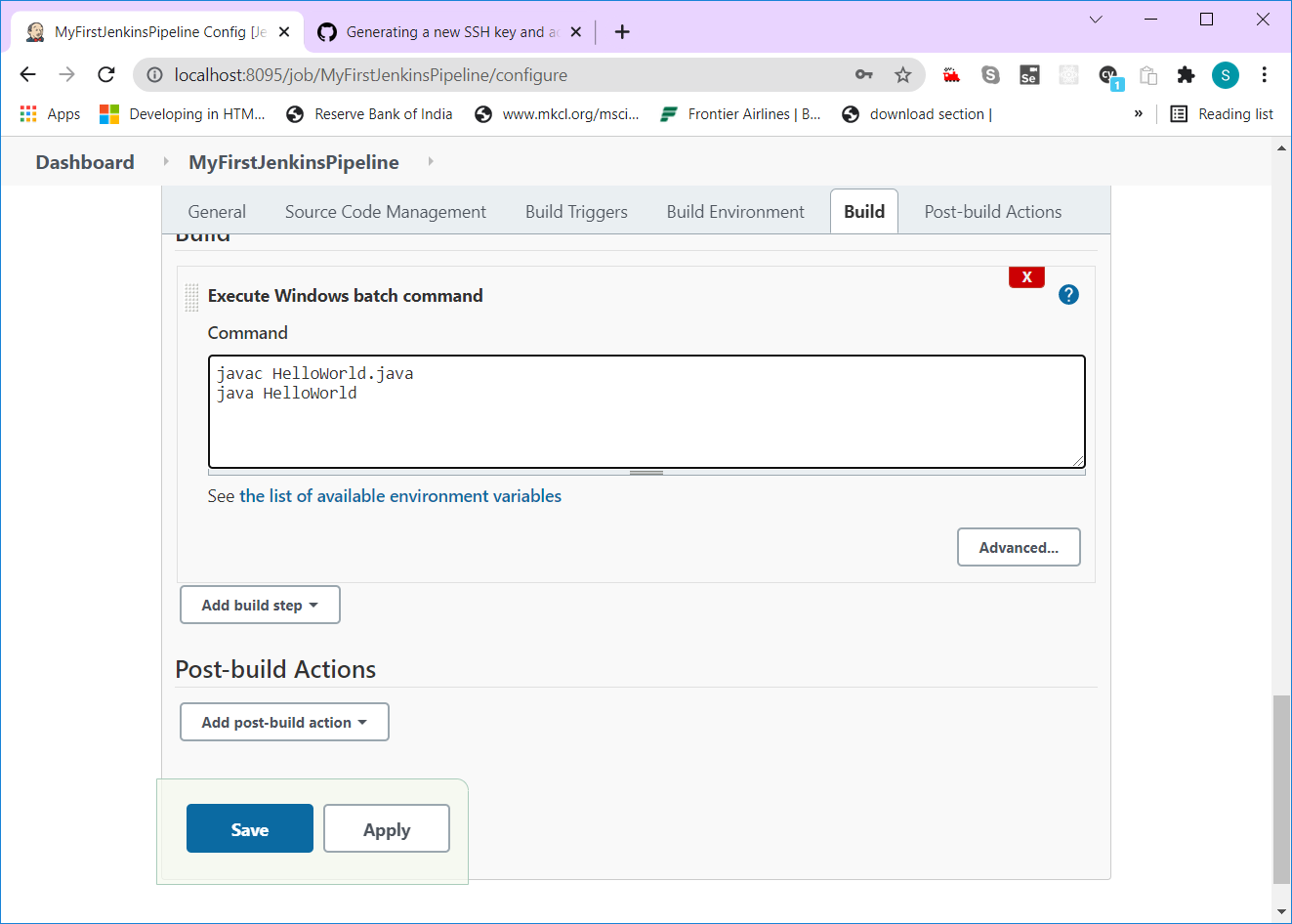
Select “Free Style Project” and click Ok,





$ ssh-keygen -t ed25519 -C "syskantechnosoft@gmail.com"





Users/{username}

getMapping

User findByUsername(String username)

be\_username = user.getUsername(); if (be\_password == null) “Invalid UserName or User not registered”

be\_password = user.getPassword();

fe\_username

fe\_password

<nav>

        <a routerLink="/customers">Customers</a>

        <a routerLink="/accounts">Accounts</a>

    </nav>