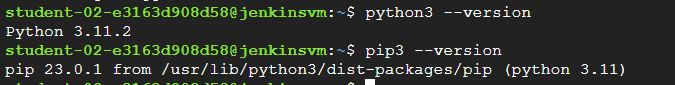
**Experiment 5b**

Part 1 – VM Setup & Selenium Installation

sudo apt update && sudo apt upgrade -y

### **3. Install Python & Pip**

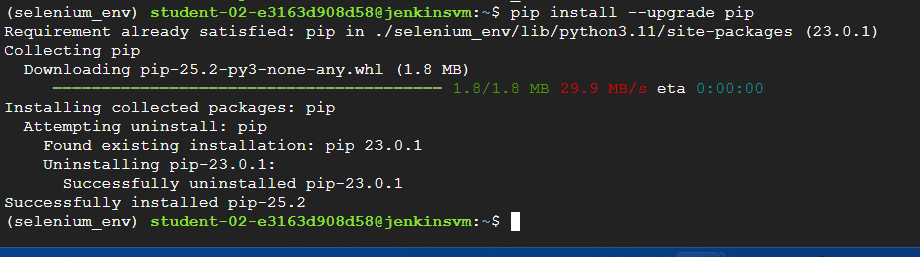
sudo apt install -y python3 python3-pip python3-venv wget unzip curl  
python3 --version  
pip3 --version



### **4. Create virtual environment**

python3 -m venv selenium\_env  
source selenium\_env/bin/activate  
pip install --upgrade pip





### **5. Install Selenium & WebDriver Manager**

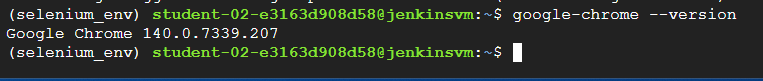
pip install selenium webdriver-manager

This avoids manual ChromeDriver downloads.

### **6. Install Google Chrome**

wget -q <https://dl.google.com/linux/direct/google-chrome-stable_current_amd64.deb>sudo apt install -y ./google-chrome-stable\_current\_amd64.deb || sudo apt --fix-broken install   
google-chrome --version





Part 2 – Write Selenium Scripts

### **A) Google Search Automation**

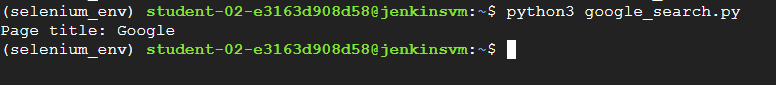
sudo vim google\_search.py

Write following code in it

from selenium import webdriver  
from selenium.webdriver.chrome.service import Service  
from selenium.webdriver.common.by import By  
from selenium.webdriver.common.keys import Keys  
from selenium.webdriver.chrome.options import Options  
from webdriver\_manager.chrome import ChromeDriverManager  
import time  
  
options = Options()  
options.add\_argument("--headless")  
options.add\_argument("--no-sandbox")  
options.add\_argument("--disable-dev-shm-usage")  
  
driver = webdriver.Chrome(service=Service(ChromeDriverManager().install()), options=options)  
driver.get("https://www.google.com")  
  
print("Page title:", driver.title)  
  
search = driver.find\_element(By.NAME, "q")  
search.send\_keys("Python Selenium tutorial")  
search.send\_keys(Keys.RETURN)  
  
time.sleep(2)  
results = driver.find\_elements(By.CSS\_SELECTOR, "h3")  
  
for i, r in enumerate(results[:5]):  
 print(f"{i+1}. {r.text}")  
  
driver.quit()

Run:

python3 google\_search.py



1. **Create a form file in your VM**  
    In your home directory, create form.html:

<!DOCTYPE html>  
<html>  
<head>  
 <title>Selenium Demo Form</title>  
</head>  
<body>  
 <h2>Selenium Practice Form</h2>  
 <form>  
 First name: <input type="text" name="firstname" /><br><br>  
 Last name: <input type="text" name="lastname" /><br><br>  
 Gender:  
 <input type="radio" id="sex-1" name="sex" value="female"> Female  
 <input type="radio" id="sex-2" name="sex" value="male"> Male<br><br>  
 Experience:  
 <input type="radio" id="exp-3" name="exp" value="3"> 3 years<br><br>  
 <input type="submit" id="submit" value="Submit" />  
 </form>  
</body>  
</html>

Save it as:

sudo vim form.html

### **B) Automate Form Submission -Save Screenshot with Explicit Path**

Update your screenshot\_test.py:

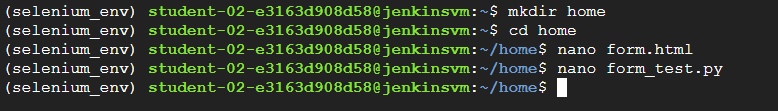
# **Part 1 – Passing Test (No Screenshot)**

pass\_test.py

# **Part 1 – Passing Test (No Screenshot)**

**Update your Selenium script (form\_test.py)**

from selenium import webdriver  
from selenium.webdriver.chrome.service import Service  
from selenium.webdriver.common.by import By  
from selenium.webdriver.chrome.options import Options  
from webdriver\_manager.chrome import ChromeDriverManager  
import time  
import os  
  
options = Options()  
options.add\_argument("--headless")  
options.add\_argument("--no-sandbox")  
options.add\_argument("--disable-dev-shm-usage")  
  
driver = webdriver.Chrome(service=Service(ChromeDriverManager().install()), options=options)  
  
# Open local file (file:// protocol)  
file\_path = f"file://{os.path.abspath('form.html')}"  
driver.get(file\_path)  
  
try:  
 driver.find\_element(By.NAME, "firstname").send\_keys("Aditi")  
 driver.find\_element(By.NAME, "lastname").send\_keys("Raut")  
 driver.find\_element(By.ID, "sex-1").click()  
 driver.find\_element(By.ID, "exp-3").click()  
 driver.find\_element(By.ID, "submit").click()  
  
 print("Form submitted successfully!")  
except Exception as e:  
 print("Test failed:", e)  
finally:  
 time.sleep(2)  
 driver.quit()





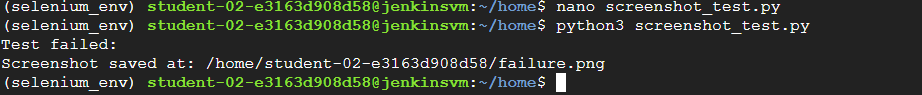
## **What “Capture Screenshots on Failure” Means**

* You **wrap your Selenium code in try/except**.
* If something goes wrong (for example, an assertion fails, or an element is missing), the script **saves a screenshot of the browser window** into a .png file.
* Since you’re running in a **headless Chrome inside your GCP VM**, you won’t “see” the browser, but the screenshot proves what the browser actually saw.

## **Example Demo Script (screenshot\_test.py)**

from selenium import webdriver  
from selenium.webdriver.chrome.service import Service  
from selenium.webdriver.chrome.options import Options  
from webdriver\_manager.chrome import ChromeDriverManager  
import os  
  
options = Options()  
options.add\_argument("--headless")  
options.add\_argument("--no-sandbox")  
options.add\_argument("--disable-dev-shm-usage")  
  
driver = webdriver.Chrome(service=Service(ChromeDriverManager().install()), options=options)  
  
try:  
 driver.get("https://www.google.com")  
 # Force a failure  
 assert "Bing" in driver.title  
 print("Test passed")  
except Exception as e:  
 print("Test failed:", e)  
 screenshot\_path = os.path.expanduser("~/failure.png")  
 if driver.save\_screenshot(screenshot\_path):  
 print(f"Screenshot saved at: {screenshot\_path}")  
 else:  
 print("Screenshot could not be saved")  
finally:  
 driver.quit()

## **Run it**

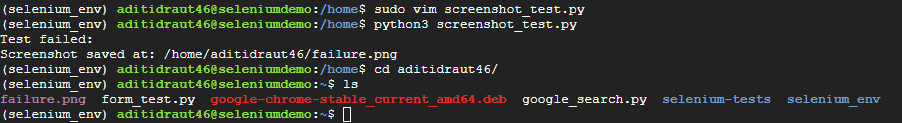
python3 screenshot\_test.py  


Expected terminal output:

Test failed: AssertionError()

And inside your VM working directory (~/ or wherever you ran it), a new file will appear:

failure.png



## **Part 3 – Run on Different Browsers**

### **Selenium Setup Guide: Chrome & Firefox (Ubuntu 22.04)**

# **Step 1: Update System**

sudo apt update  
sudo apt upgrade -y  
**Step 2: Install Required Dependencies**

sudo apt install -y wget curl unzip \ libappindicator3-1 libasound2 libatk-bridge2.0-0 libcups2 libdbus-1-3 \ libxss1 libxtst6 libnss3 libgtk-3-0 libx11-xcb1 fonts-liberation \ python3-pip

# **Step 3: Install Google Chrome**

1. Download Chrome:

wget <https://dl.google.com/linux/direct/google-chrome-stable_current_amd64.deb>

1. Install Chrome:

sudo apt install ./google-chrome-stable\_current\_amd64.deb --allow-downgrades -y

1. Verify installation:

which google-chrome-stable  
google-chrome-stable --version

Expected output:

/usr/bin/google-chrome-stable  
Google Chrome 140.x.xxxx.xx



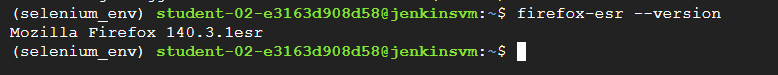
# **Step 4: Install Firefox ESR**

sudo apt install -y firefox-esr

Verify:

which firefox-esr  
firefox-esr --version

Expected output:

/usr/bin/firefox-esr  
Mozilla Firefox 140.x.xesr  
 

# **Step 5: Install Selenium and WebDriver Manager**

python3 -m pip install --upgrade pip  
python3 -m pip install selenium webdriver-manager

# **Step 6: Save the Python Selenium Script**

Save as cross\_browser\_test.py:

from selenium import webdriver  
from selenium.webdriver.chrome.service import Service as ChromeService  
from selenium.webdriver.firefox.service import Service as FirefoxService  
from selenium.webdriver.chrome.options import Options as ChromeOptions  
from selenium.webdriver.firefox.options import Options as FirefoxOptions  
from webdriver\_manager.chrome import ChromeDriverManager  
from webdriver\_manager.firefox import GeckoDriverManager  
  
URL = "<https://www.google.com>"  
  
# Chrome  
chrome\_options = ChromeOptions()  
chrome\_options.binary\_location = "/usr/bin/google-chrome-stable"  
chrome\_options.add\_argument("--headless")  
chrome\_options.add\_argument("--no-sandbox")  
chrome\_options.add\_argument("--disable-dev-shm-usage")  
chrome\_options.add\_argument("--disable-gpu")  
  
chrome\_driver = webdriver.Chrome(service=ChromeService(ChromeDriverManager().install()), options=chrome\_options)  
chrome\_driver.get(URL)  
print("Chrome Title:", chrome\_driver.title)  
chrome\_driver.quit()  
  
# Firefox  
firefox\_options = FirefoxOptions()  
firefox\_options.binary\_location = "/usr/bin/firefox-esr"  
firefox\_options.add\_argument("--headless")  
  
firefox\_driver = webdriver.Firefox(service=FirefoxService(GeckoDriverManager().install()), options=firefox\_options)  
firefox\_driver.get(URL)  
print("Firefox Title:", firefox\_driver.title)  
firefox\_driver.quit()

# **Step 7: Run the Script**

python3 cross\_browser\_test.py

Expected output:

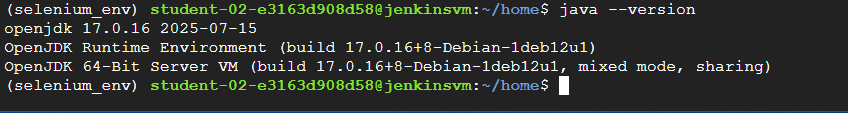
Chrome Title: Google  
Firefox Title: Google

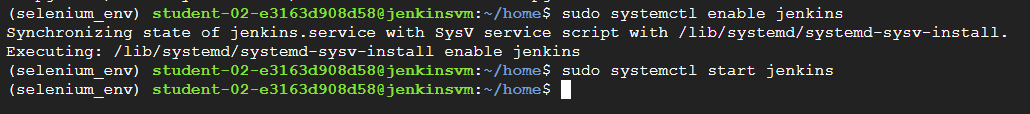


## **Part 4 – Jenkins Integration**

### **1. Install Jenkins**

sudo apt update  
sudo apt install -y openjdk-17-jdk  
java –version



wget -q -O - <https://pkg.jenkins.io/debian/jenkins.io-2023.key> | sudo tee /usr/share/keyrings/jenkins-keyring.asc  
echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] <https://pkg.jenkins.io/debian> binary/ | sudo tee /etc/apt/sources.list.d/jenkins.list  
  
sudo apt update  
sudo apt install -y jenkins  
sudo systemctl enable jenkins  
sudo systemctl start jenkins  


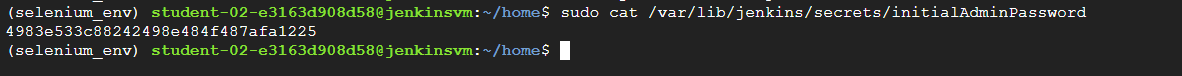
Access Jenkins: http://<VM\_IP>:8080

Create firewall rule to add TCP port no 8080

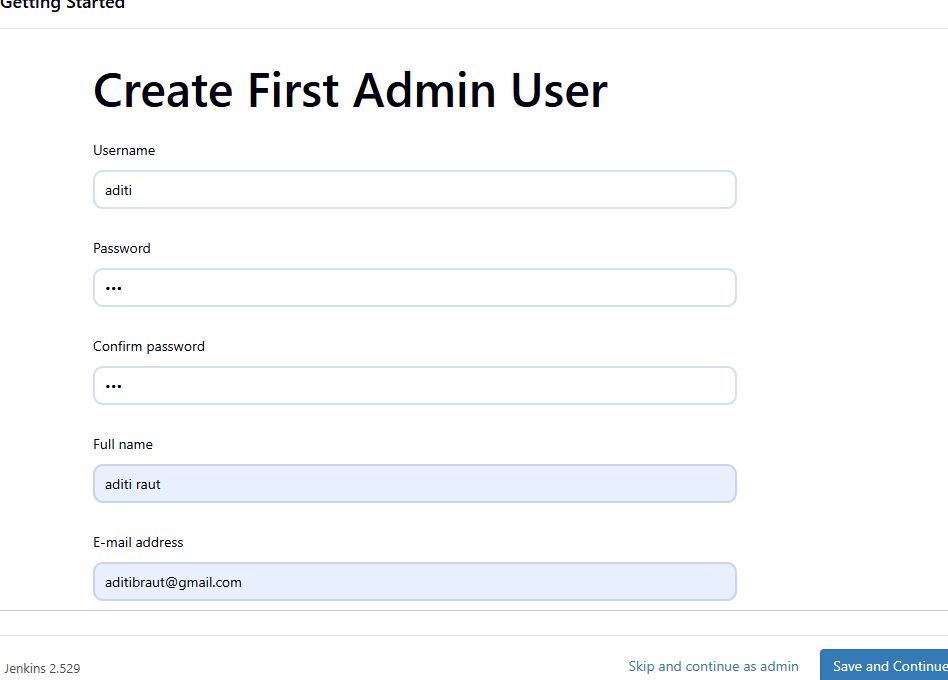
### **2. Configure Jenkins**

* Login with initial admin password:

sudo cat /var/lib/jenkins/secrets/initialAdminPassword

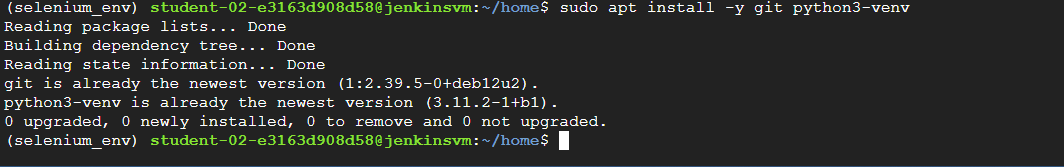


* Install **suggested plugins**.
* Create admin user.



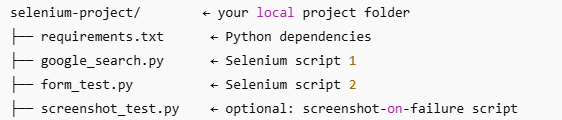
### **3. Install Python & Git in Jenkins environment**

sudo apt install -y git python3-venv

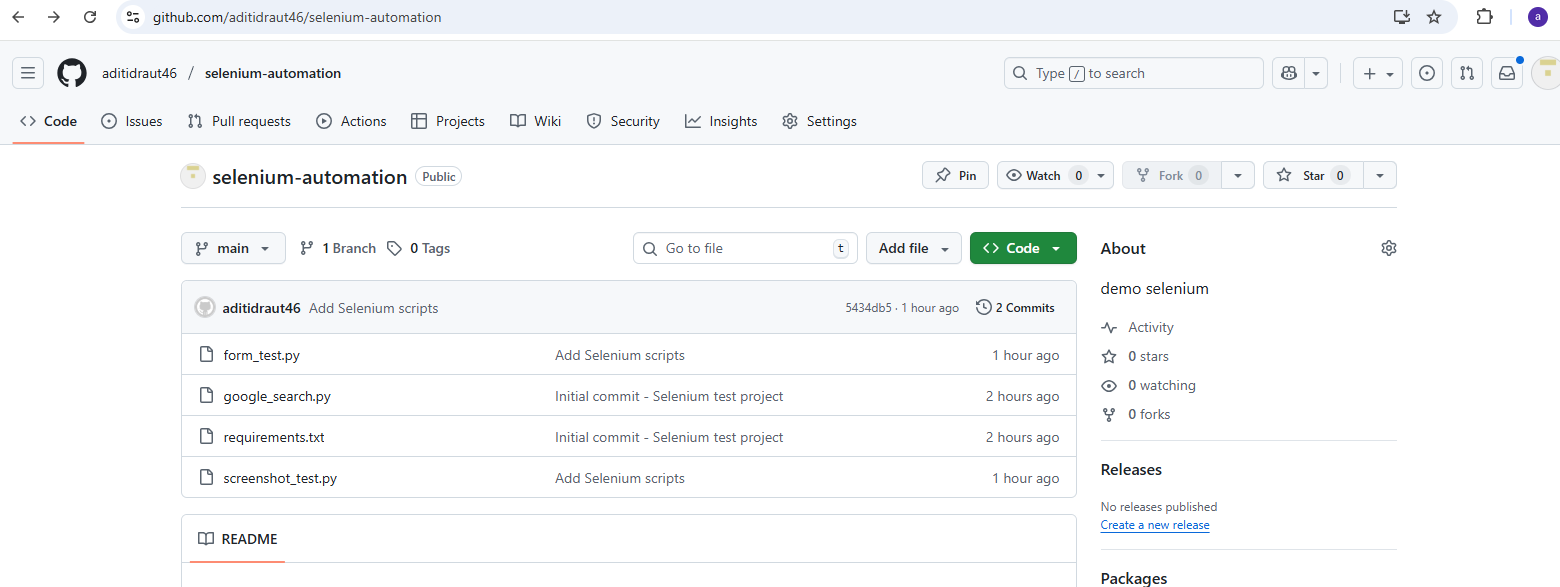


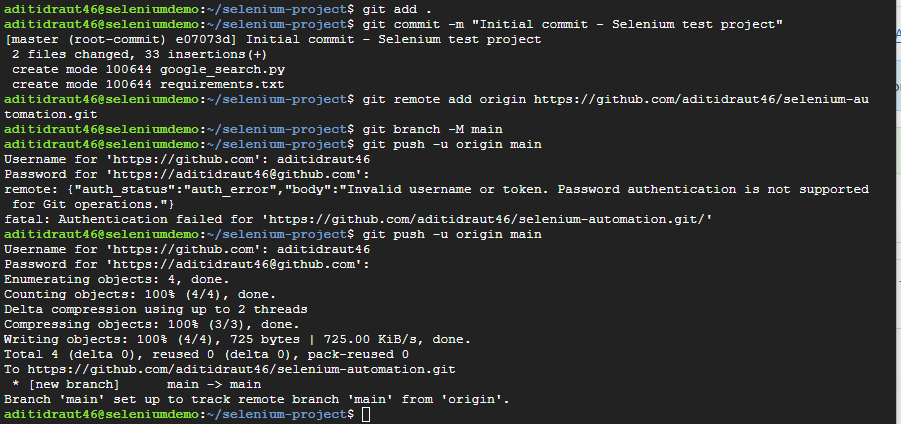
### **4. Create a Jenkins Job**

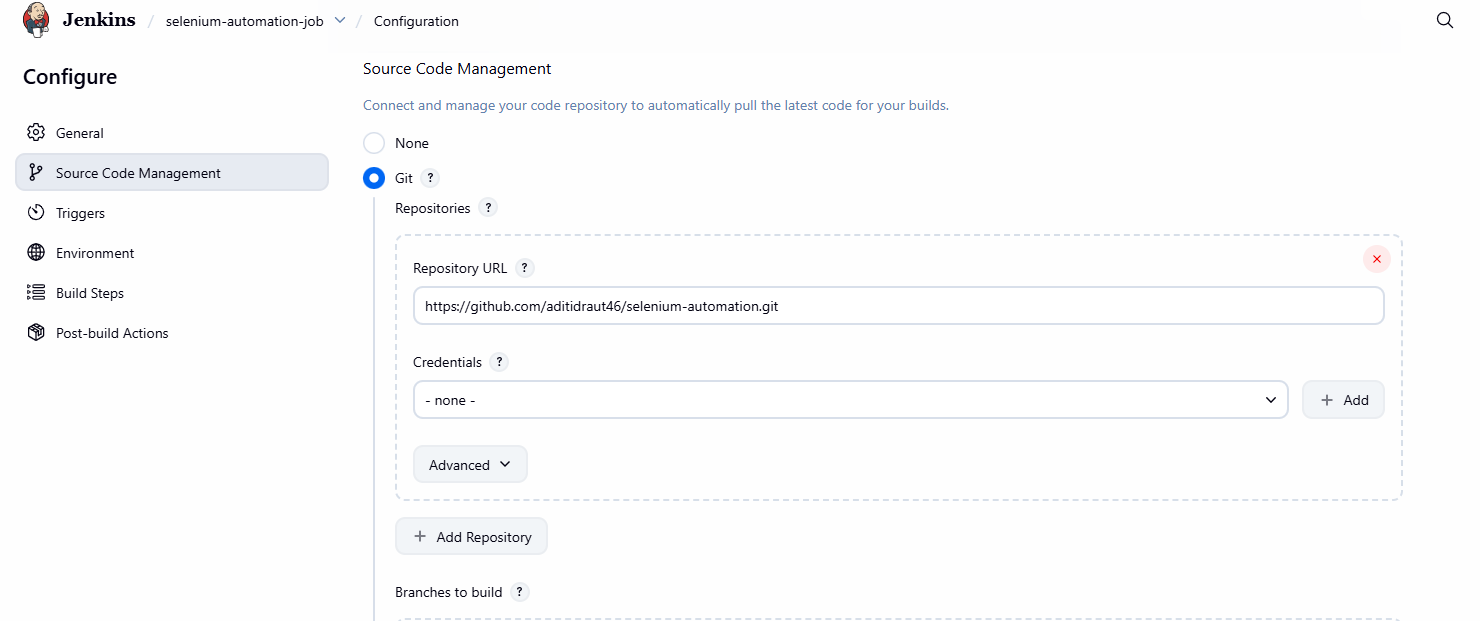
1. **Create Jenkins Job to Execute Selenium Scripts**
2. Go to Jenkins Dashboard → **New Item**
3. Enter a name: selenium-automation-job
4. Select **Freestyle project** → Click **OK**











#!/bin/bash  
cd $WORKSPACE  
python3 -m venv venv  
source venv/bin/activate  
pip install -r requirements.txt  
python3 google\_search.py  
python3 form\_test.py

1. Save and **Build Now**.

