**Why Python for Automation Testing?**

It is easy to learn and understand, simpler to code

World is moving towards artificial intelligence with machine learning, and python plays a crucial role in implementing them

More Job, Less competition

**Selenium WebDriver** – Open-source tool to automates web applications

Selenium can be run in Mutiple OS.

Selenium can be coded in multiple languages – Java, Python, JavaScript, C#, Ruby etc.

------------------------------------------------------------------------------------------------------------------------------------------

Download the Python latest version from the [Download Python | Python.org](https://www.python.org/downloads/) depending on the OS.

Make sure where the python path is created in your machine.

C:\Python311

C:\Python311\Scripts

Set python home in System Variables

Check from CMD python if it is successfully installed or not. Python --version

------------------------------------------------------------------------------------------------------------------------------------------

**What is PIP?**

Pip is a standard package manager for Python. It allows you to install and manage additional packages that are not part of Python Standard Library

Selenium needs to be downloaded and install by following command

**pip install selenium**

[selenium · PyPI](https://pypi.org/project/selenium/)

Confirm from path C:\Python311\Lib\site-packages

Verify selenium is installed or not by following command

**pip show selenium**

To upgrade the selenium

**pip install -U selenium**

Download the PyCharm community edition

[Download PyCharm: Python IDE for Professional Developers by JetBrains](https://www.jetbrains.com/pycharm/download/?section=windows)

After successful installation

Open PyCharm -> New Project -> Select the location and give a name for folder -> Select Python Base interpreter so that it creates a new python environment to work (It automatically recognizes the path if python is already installed) -> Create

Got to file -> settings -> search interpreter and check interpreter is set

Check on IDE level also for the interpreter is set

Got to file -> settings -> search interpreter -> Add Interpreter -> Add local interpreter -> Select System Interpreter (C:\Python311\python.exe)-> click ok -> click apply -> click ok

We are selecting this system interpreter because to use the interpreter which is installed in the system

------------------------------------------------------------------------------------------------------------------------------------------

Create a new file and start working on learning Python.

**FirstDemo.py**

print("Hello")  
# Here are the comments I have defined  
#Code induntation  
  
a = 3  
print(a)  
  
Str = "Hello World!"  
print(Str)  
  
b, c, d, = 5, 6.4, "Great"  
  
#print("Value is "+b)  
  
print("{} {}".format("Value is",b))  
#To cprint the string and int or other values we use the above method  
  
print(type(b))  
print(type(c))  
print(type(d))

------------------------------------------------------------------------------------------------------------------------------------------

**Datatype and its operations to manipulate**

[Python Data Types (With Complete List) | DigitalOcean](https://www.digitalocean.com/community/tutorials/python-data-types)

**Demo2.py**

values = [1, 2, "Mujeeb", 4, 5]  
# List is a data type that allows multiple values and can be different data types  
  
  
print(values[0]) # 1  
print(values[3]) # 4  
print(values[-1]) # 5 -1 gives th last index  
print(values[1:3]) # Printing from 2 index to 4 the index  
  
values.insert(3,"Rahaman") # inserting the value at 3rd index  
print(values) # [1, 2, 'Mujeeb', 'Rahaman', 4, 5]  
  
values.append("End") #This appends the value at the end  
print(values) # [1, 2, 'Mujeeb', 'Rahaman', 4, 5, 'End']  
  
# Updating the value depending on index value  
values[2] = "Mujju"  
print(values) # [1, 2, 'Mujju', 'Rahaman', 4, 5, 'End']  
  
# Deleting the value depeding on index value  
del values[0]  
print(values) # [2, 'Mujju', 'Rahaman', 4, 5, 'End']  
  
#--------------------------------------------------------------------------------------

# List is mutable but Tuple is immutable else all others are same such that updations are not possible  
  
#Tuple  
val = (1, 2, "Mujeeb", 4.5)  
print(val[1])  
  
# val[2] = "Mujju" #No Possible  
  
#---------------------------------------------------------------------------------------  
  
# Dictonary  
  
dic = {"a": 2, 4: "mujju", "c": "Hello World!"}  
  
#Prints Not on index, but on the values assigned  
print(dic[4])  
print(dic["c"])  
  
#How to create dictonary dynamically during run time {used in excell driven}  
  
dict = {}  
  
dict["firstname"] = "Mujeeb"  
  
dict["lastname"] = "Rahaman"  
  
dict["gender"] = "Male"  
  
print(dict)  
print(dict["lastname"])

------------------------------------------------------------------------------------------------------------------------------------------

**Program Flow Control in Python**

**Loops.py**

greeting = "Good Morning"  
a = 4  
  
if a > 2:  
 print("Condition Matches")  
 print("Second line")  
else:  
 print("Condition do not match")  
  
print("if else condition code is completed")  
  
  
# for loop  
  
obj = [2, 3, 5, 7, 9]  
  
for i in obj:  
 print(i\*2)  
  
# Sum of first 5 natural numbers  
# range(i,j) -> i to j-1  
summation = 0  
for j in range(1, 6):  
 summation = summation + j  
print(summation)  
  
print("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")  
for k in range(1, 10, 2):  
 print(k)  
  
print("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")  
  
for m in range(10):  
 print(m)

**WhileDemo.py**

it = 10  
  
  
# while it>1:  
# print(it)  
# it = it - 1  
# print("While loop execution is done")  
  
  
  
  
# while it>1:  
# if it != 3:  
# print(it)  
  
# it = it - 1  
# print("While loop execution is done")  
  
  
  
  
# while it>1:  
# if it == 3:  
# break  
# print(it)  
#  
# it = it - 1  
# print("While loop execution is done")  
  
  
  
  
while it>1:  
 if it ==9:  
 it = it - 1  
 continue  
 if it == 3:  
 break  
 print(it)  
  
 it = it - 1  
print("While loop execution is done")

**FuntionsDemo.py**

# In python , function is a group of related statements that perform a specific task.  
  
# Function Declaration  
def GreetMe(name):  
 print("Good Morning " + name)  
 # Function Cell  
  
  
def AddIntegers(a, b):  
 print(a+b)  
  
def AddIntegers1(a, b):  
 return a+b  
  
  
GreetMe("Mujeeb Rahaman")  
  
AddIntegers(2,3)  
  
print(AddIntegers1(5,5))

------------------------------------------------------------------------------------------------------------------------------------------

**OOPS Principles: Classes and objects in Python**

**OopsDemo.py**

# Classes are user defined blueprint or prototype  
# sum, multiplication, addition, constant  
# methods, class variables, instance variables, constructor etc..  
# Objects for your classes-  
  
  
# Class and methods and calling the methods and variables using objects  
class Calculator:  
 num = 100  
  
 def getData(self):  
 print("I am now executing as method in class")  
  
  
obj = Calculator() # Syntax to create objects in python  
obj.getData()  
print(obj.num)  
print("----------------------------------------------------------------")  
  
# Calling self constructor  
# Two Variables - 1) Class variable 2) Instance Variable  
# Class variables will be constant no matter how many objects we create  
# Variables which are defined inside constructor are called instance variables  
# The number of arguments passed in class , same number of arguments should also be passed in the constructor  
# self is an object, the obj passed as first argument to self  
# self keyword is madatory for calling variables names in methods  
# Constructor name should be \_\_init\_\_  
# new keyword is not required when you create object  
  
class Calculator1:  
 num1 = 1000 #Class variable  
 # Default constructor  
  
 def \_\_init\_\_(self, a, b):  
 self.firstNumber= a #Instance variables  
 self.secondNumber= b  
 print("I am called automatically when object is created")  
  
 def getData1(self):  
 print("I am now executing as method in class")  
  
 def Summation(self):  
 return self.firstNumber + self.secondNumber + Calculator1.num1  
 # Calling variables with self. in the methods --Mandatory in python  
  
# Parameterised constructor  
obj1 = Calculator1(2,3)  
obj1.getData1()  
print(obj1.Summation())  
print(obj1.num1)  
  
obj3 = Calculator1(4,5)  
obj3.getData1()  
print(obj3.Summation())  
print(obj3.num1)  
  
# Non-Parameterised constructor  
# obj2 = Calculator1()  
# obj2.getData1()  
# print(obj2.num1)

**ChildImp.py**

# Inheritance is child acquiring properties from parent class  
  
# 1)Calling parent in class  
# 2)Importing the parent  
# 3)If there is constructor no default, then make sure to call parent constructor  
  
from OopsDemo import Calculator1  
  
  
class ChildImp1(Calculator1):  
 num2 = 200  
  
 def \_\_init\_\_(self):  
 Calculator1.\_\_init\_\_(self,2,10)  
  
 def getCompleteData(self):  
 return self.num2 + self.num1 + self.Summation()  
  
  
obj = ChildImp1()  
print(obj.getCompleteData())  
]

**StringsDemo.py**

str = "[mujeeb.rahaman@cognine.com](mailto:mujeeb.rahaman@cognine.com)"  
str1 = "Service Based"  
str2 = "mujeeb"  
  
print(str)  
print(str[1]) # To print only some character at mentioned index  
print(str[0:6]) # To get the sub-string  
print(str+str1) #Concatenation  
print(str2 in str)# To check whether the string is present in or not  
var = str.split(".") # Splitting the string  
print(var)  
print(var[0])  
str3 = " Great "  
print(str3.strip()) #Removing white spaces  
print(str3.lstrip()) #Removing left white spaces  
print(str3.rstrip()) #Removing right white spaces

------------------------------------------------------------------------------------------------------------------------------------------

**Read & Write to Files in Python**

**Test.txt**

abc  
bjhfc  
cat  
dog  
elephant

**ReadWrite.py**

file = open('test.txt')  
# Read all the context of file  
# print(file.read())  
# print(file.read(5)) # To read n number of characters by passing paramater  
  
# To read one single line at a time readline()  
# print(file.readline())  
# print(file.readline())  
  
  
# Print line by line using readline method  
# line = file.readline()  
# while line!="":  
# print(line)  
# line = file.readline()  
  
for line in file.readlines():  
 print(line)  
  
file.close()

**Write.py**

# file = open("test.txt")  
#  
# file.close()  
  
# Read the file and store all the lines in list  
# Reverse the list  
# write the list back to the file  
with open('test.txt', 'r') as reader:  
 content = reader.readlines() #[abc,bjhfc,cat,dog,elephant]  
 reversed(content) ##[elephant,dog,cat,bjhfc,abc]  
 with open('test.txt', 'w') as writer:  
 for line in reversed(content):  
 writer.write(line)

------------------------------------------------------------------------------------------------------------------------------------------

**Exception Handling Mechanism in Python**

**ExeceptionsPythin.py**

ItemsInCart = 0  
# 2 Items will be added to cart  
  
if ItemsInCart !=2:  
 # raise Exception("Products Cart count not matching")  
 pass  
  
assert(ItemsInCart == 0)  
  
# try, except (Catch) mechanism  
  
try:  
 with open("filelog.txt",'r') as reader:  
 reader.read()  
  
except:  
 print("Some how I reached this block because there is failure in try block")  
  
  
try:  
 with open("test.txt",'r') as reader:  
 reader.read()  
  
except Exception as e:  
 print(e)  
  
  
finally:  
 print("Cleaning up records")

------------------------------------------------------------------------------------------------------------------------------------------  
  
**Selenium python package setup with different browsers execution**

demoBrowser.py

from selenium import webdriver  
from selenium.webdriver.chrome.service import Service  
  
#Chrome driver - Chrome browser  
# service\_obj = Service() #seleniumManager  
# driver = webdriver.Chrome(service=service\_obj)  
# driver.get("https://rahulshettyacademy.com")  
  
# -- Chrome  
service\_obj = Service("/Users/Mujeeb Rahaman/Downloads/PythonDocs/chromedriver-win64/chromedriver-win64/chromedriver.exe")  
driver = webdriver.Chrome(service=service\_obj)  
  
# -- Firefox  
  
# service\_obj = Service("/Users/Mujeeb Rahaman/Downloads/PythonDocs/geckodriver-v0.33.0-win64/geckodriver.exe")  
# driver = webdriver.Firefox(service=service\_obj)  
  
# -- Edge  
  
# service\_obj = Service("/Users/Mujeeb Rahaman/Downloads/PythonDocs/msedgedriver.exe")  
# driver = webdriver.Edge(service=service\_obj)  
  
driver.maximize\_window()  
driver.get("https://rahulshettyacademy.com")  
print(driver.title)  
print(driver.current\_url)  
driver.get("https://rahulshettyacademy.com/seleniumPractise/#/")  
driver.minimize\_window()  
driver.back()  
driver.refresh()  
driver.forward()  
driver.close()

------------------------------------------------------------------------------------------------------------------------------------------  
  
**Different Types of Locators in WebDriver API with Examples**

**Locators.py**

from selenium import webdriver  
from selenium.webdriver.chrome.service import Service  
from selenium.webdriver.common.by import By  
  
service\_obj = Service("/Users/Mujeeb Rahaman/Downloads/PythonDocs/chromedriver-win64/chromedriver-win64/chromedriver.exe")  
driver = webdriver.Chrome(service=service\_obj)  
  
driver.get("https://rahulshettyacademy.com/angularpractice/")  
  
# ID, Xpath, CSSSelector, Classname, name , linkText  
driver.find\_element(By.CSS\_SELECTOR,"input[name='name']").send\_keys("Mujeeb")  
[driver.find\_element(By.NAME,"email").send\_keys("mujeeb7036@gmail.com](mailto:driver.find_element(By.NAME,"email").send_keys("mujeeb7036@gmail.com)")  
driver.find\_element(By.ID,"exampleInputPassword1").send\_keys("Mujeeb@8143")  
driver.find\_element(By.CSS\_SELECTOR,"input[value='option1']").click()  
driver.find\_element(By.ID,"exampleCheck1").click()  
  
# Xpath - //tagname[@attribute='value'] -> //input[@type='submit']  
# CSS - tagname[attribute='value'] -> //input[@type='submit'], #id, .classname  
  
[driver.find\_element(By.XPATH,"//input[@type='submit']").click](mailto:driver.find_element(By.XPATH,"//input[@type='submit']").click)()  
  
message = driver.find\_element(By.CLASS\_NAME,"alert-success").text  
print(message)  
assert "Success" in message  
  
[driver.find\_element(By.XPATH,"(//input[@type='text'])[3]").send\_keys("HelloWorld](mailto:driver.find_element(By.XPATH,"(//input[@type='text'])[3]").send_keys("HelloWorld)!")  
[driver.find\_element(By.XPATH,"(//input[@type='text'])[3]").clear](mailto:driver.find_element(By.XPATH,"(//input[@type='text'])[3]").clear)()

**LocatorsExtension.py**

from selenium import webdriver  
from selenium.webdriver.chrome.service import Service  
from selenium.webdriver.common.by import By  
  
service\_obj = Service("/Users/Mujeeb Rahaman/Downloads/PythonDocs/chromedriver-win64/chromedriver-win64/chromedriver.exe")  
driver = webdriver.Chrome(service=service\_obj)  
  
driver.get("https://rahulshettyacademy.com/client/")  
  
driver.find\_element(By.LINK\_TEXT,"Forgot password?").click()  
[driver.find\_element(By.XPATH,"//form/div[1]/input").send\_keys("demo@gmail.com](mailto:driver.find_element(By.XPATH,"//form/div[1]/input").send_keys("demo@gmail.com)")  
driver.find\_element(By.CSS\_SELECTOR,"form div:nth-child(2) input").send\_keys("Hello@1234")  
driver.find\_element(By.CSS\_SELECTOR,"#confirmPassword").send\_keys("Hello@1234")  
# [driver.find\_element(By.XPATH,"//button[@type='submit']").click](mailto:driver.find_element(By.XPATH,"//button[@type='submit']").click)()  
driver.find\_element(By.XPATH,"//button[text()='Save New Password']").click()  
  
  
------------------------------------------------------------------------------------------------------------------------------------------  
  
**Python API-> Techniques to automate Web elements**

**StaticDropdowns.py**

from selenium import webdriver  
from selenium.webdriver.chrome.service import Service  
from selenium.webdriver.common.by import By  
from selenium.webdriver.support.select import Select  
  
service\_obj = Service("/Users/Mujeeb Rahaman/Downloads/PythonDocs/chromedriver-win64/chromedriver-win64/chromedriver.exe")  
driver = webdriver.Chrome(service=service\_obj)  
  
driver.get("https://rahulshettyacademy.com/angularpractice/")  
  
# ID, Xpath, CSSSelector, Classname, name , linkText  
driver.find\_element(By.CSS\_SELECTOR,"input[name='name']").send\_keys("Mujeeb")  
[driver.find\_element(By.NAME,"email").send\_keys("mujeeb7036@gmail.com](mailto:driver.find_element(By.NAME,"email").send_keys("mujeeb7036@gmail.com)")  
driver.find\_element(By.ID,"exampleInputPassword1").send\_keys("Mujeeb@8143")  
driver.find\_element(By.CSS\_SELECTOR,"input[value='option1']").click()  
driver.find\_element(By.ID,"exampleCheck1").click()  
  
# Xpath - //tagname[@attribute='value'] -> //input[@type='submit']  
# CSS - tagname[attribute='value'] -> //input[@type='submit'], #id, .classname  
driver.find\_element(By.CSS\_SELECTOR,"#inlineRadio1").click()  
  
#Static Dropdown  
dropdown = Select(driver.find\_element(By.ID,"exampleFormControlSelect1"))  
dropdown.select\_by\_visible\_text("Female")  
dropdown.select\_by\_index(0)  
# dropdown.select\_by\_value()  
  
  
  
[driver.find\_element(By.XPATH,"//input[@type='submit']").click](mailto:driver.find_element(By.XPATH,"//input[@type='submit']").click)()  
  
message = driver.find\_element(By.CLASS\_NAME,"alert-success").text  
print(message)  
assert "Success" in message  
  
[driver.find\_element(By.XPATH,"(//input[@type='text'])[3]").send\_keys("HelloWorld](mailto:driver.find_element(By.XPATH,"(//input[@type='text'])[3]").send_keys("HelloWorld)!")  
[driver.find\_element(By.XPATH,"(//input[@type='text'])[3]").clear](mailto:driver.find_element(By.XPATH,"(//input[@type='text'])[3]").clear)()

**FindElementsTest.py**

import time  
  
from selenium import webdriver  
from selenium.webdriver.chrome.service import Service  
from selenium.webdriver.common.by import By  
  
service\_obj = Service("/Users/Mujeeb Rahaman/Downloads/PythonDocs/chromedriver-win64/chromedriver-win64/chromedriver.exe")  
driver = webdriver.Chrome(service=service\_obj)  
  
driver.get("https://rahulshettyacademy.com/dropdownsPractise/")  
  
driver.find\_element(By.ID,"autosuggest").send\_keys("ind")  
time.sleep(2)  
countries = driver.find\_elements(By.CSS\_SELECTOR,"li[class='ui-menu-item'] a")  
print(len(countries))  
  
for country in countries:  
 if country.text == "India":  
 country.click()  
 break  
  
# print(driver.find\_element(By.ID,"autosuggest").text)  
assert driver.find\_element(By.ID,"autosuggest").get\_attribute("value") == "India"

**UiControls.py**

from selenium import webdriver  
from selenium.webdriver.chrome.service import Service  
from selenium.webdriver.common.by import By  
  
service\_obj = Service("/Users/Mujeeb Rahaman/Downloads/PythonDocs/chromedriver-win64/chromedriver-win64/chromedriver.exe")  
driver = webdriver.Chrome(service=service\_obj)  
  
driver.get("https://rahulshettyacademy.com/AutomationPractice/")  
  
checkboxes = driver.find\_elements(By.XPATH,"//input[@type='checkbox']")  
print(len(checkboxes))  
  
for checkbox in checkboxes:  
 if checkbox.get\_attribute("value") == "option2":  
 checkbox.click()  
 checkbox.is\_selected()  
 break  
  
radioButtons = driver.find\_elements(By.CSS\_SELECTOR,".radioButton")  
radioButtons[2].click()  
assert radioButtons[2].is\_selected()  
  
assert driver.find\_element(By.ID,"displayed-text").is\_displayed()  
driver.find\_element(By.ID,"hide-textbox").click()  
assert not driver.find\_element(By.ID,"displayed-text").is\_displayed()

**Alerts.py**

from selenium import webdriver  
from selenium.webdriver.chrome.service import Service  
from selenium.webdriver.common.by import By  
  
name = "Mujeeb"  
service\_obj = Service("/Users/Mujeeb Rahaman/Downloads/PythonDocs/chromedriver-win64/chromedriver-win64/chromedriver.exe")  
driver = webdriver.Chrome(service=service\_obj)  
  
driver.get("https://rahulshettyacademy.com/AutomationPractice/")  
  
driver.find\_element(By.ID,"name").send\_keys(name)  
driver.find\_element(By.ID,"alertbtn").click()  
alert = driver.switch\_to.alert  
alertText = alert.text  
print(alertText)  
assert name in alertText  
alert.accept()  
# alert.dismiss()

------------------------------------------------------------------------------------------------------------------------------------------  
**Synchronization (Explicit & Implicit Waits) in Selenium WebDriver**

**WaitDemo.py**

import time  
  
from selenium import webdriver  
from selenium.webdriver.chrome.service import Service  
from selenium.webdriver.common.by import By  
  
name = "Mujeeb"  
service\_obj = Service("/Users/Mujeeb Rahaman/Downloads/PythonDocs/chromedriver-win64/chromedriver-win64/chromedriver.exe")  
driver = webdriver.Chrome(service=service\_obj)  
driver.implicitly\_wait(5)  
  
# 5 seconds is max timeout..  
  
driver.get("https://rahulshettyacademy.com/seleniumPractise/#/")  
  
driver.find\_element(By.CSS\_SELECTOR,"input[type='search']").send\_keys("ber")  
time.sleep(2)  
results = driver.find\_elements(By.XPATH,"//div[@class='products']/div") #list  
count = len(results)  
assert count > 0  
for result in results:  
 result.find\_element(By.XPATH,"div/button").click()  
  
  
driver.find\_element(By.CSS\_SELECTOR,"a[class='cart-icon']").click()  
driver.find\_element(By.XPATH,"//button[text()='PROCEED TO CHECKOUT']").click()  
driver.find\_element(By.CSS\_SELECTOR,".promoCode").send\_keys("rahulshettyacademy")  
driver.find\_element(By.CSS\_SELECTOR,"button[class='promoBtn']").click()  
print(driver.find\_element(By.CLASS\_NAME,"promoInfo").text)

**ExplicitwaitDemo.py**

import time  
  
from selenium import webdriver  
from selenium.webdriver.chrome.service import Service  
from selenium.webdriver.common.by import By  
from selenium.webdriver.support import expected\_conditions  
from selenium.webdriver.support.wait import WebDriverWait  
  
name = "Mujeeb"  
service\_obj = Service("/Users/Mujeeb Rahaman/Downloads/PythonDocs/chromedriver-win64/chromedriver-win64/chromedriver.exe")  
driver = webdriver.Chrome(service=service\_obj)  
driver.implicitly\_wait(2)  
  
# 5 seconds is max timeout..  
  
driver.get("https://rahulshettyacademy.com/seleniumPractise/#/")  
  
driver.find\_element(By.CSS\_SELECTOR,"input[type='search']").send\_keys("ber")  
time.sleep(2)  
results = driver.find\_elements(By.XPATH,"//div[@class='products']/div") #list  
count = len(results)  
assert count > 0  
for result in results:  
 result.find\_element(By.XPATH,"div/button").click()  
  
  
driver.find\_element(By.CSS\_SELECTOR,"a[class='cart-icon']").click()  
driver.find\_element(By.XPATH,"//button[text()='PROCEED TO CHECKOUT']").click()  
driver.find\_element(By.CSS\_SELECTOR,".promoCode").send\_keys("rahulshettyacademy")  
driver.find\_element(By.CSS\_SELECTOR,"button[class='promoBtn']").click()  
wait = WebDriverWait(driver,10)  
wait.until(expected\_conditions.visibility\_of\_element\_located((By.CSS\_SELECTOR,".promoInfo")))  
print(driver.find\_element(By.CLASS\_NAME,"promoInfo").text)

------------------------------------------------------------------------------------------------------------------------------------------  
**Deep Dive into Functional Automation using Python**

**ExplicitwaitDemo.py**

import time  
  
from selenium import webdriver  
from selenium.webdriver.chrome.service import Service  
from selenium.webdriver.common.by import By  
from selenium.webdriver.support import expected\_conditions  
from selenium.webdriver.support.wait import WebDriverWait  
expectedList = ['Cucumber - 1 Kg', 'Raspberry - 1/4 Kg', 'Strawberry - 1/4 Kg']  
actualList = []  
  
name = "Mujeeb"  
service\_obj = Service("/Users/Mujeeb Rahaman/Downloads/PythonDocs/chromedriver-win64/chromedriver-win64/chromedriver.exe")  
driver = webdriver.Chrome(service=service\_obj)  
driver.implicitly\_wait(2)  
  
# 5 seconds is max timeout..  
  
driver.get("https://rahulshettyacademy.com/seleniumPractise/#/")  
  
driver.find\_element(By.CSS\_SELECTOR,"input[type='search']").send\_keys("ber")  
time.sleep(2)  
results = driver.find\_elements(By.XPATH,"//div[@class='products']/div") #list  
count = len(results)  
assert count > 0  
for result in results:  
 actualList.append(result.find\_element(By.XPATH,"h4").text)  
 result.find\_element(By.XPATH,"div/button").click()  
  
# print(actualList)  
assert expectedList == actualList  
  
driver.find\_element(By.CSS\_SELECTOR,"a[class='cart-icon']").click()  
driver.find\_element(By.XPATH,"//button[text()='PROCEED TO CHECKOUT']").click()  
  
#Sum Validation  
prices = driver.find\_elements(By.CSS\_SELECTOR,"tr td:nth-child(5) p")  
sum = 0  
for price in prices:  
 sum = sum + int(price.text)  
  
print(sum)  
  
totalAmount = int(driver.find\_element(By.CSS\_SELECTOR,".totAmt").text)  
  
assert sum == totalAmount  
  
driver.find\_element(By.CSS\_SELECTOR,".promoCode").send\_keys("rahulshettyacademy")  
driver.find\_element(By.CSS\_SELECTOR,"button[class='promoBtn']").click()  
wait = WebDriverWait(driver,10)  
wait.until(expected\_conditions.visibility\_of\_element\_located((By.CSS\_SELECTOR,".promoInfo")))  
print(driver.find\_element(By.CLASS\_NAME,"promoInfo").text)  
  
discountAmount = float(driver.find\_element(By.CSS\_SELECTOR,".discountAmt").text)  
assert totalAmount > discountAmount  
  
  
------------------------------------------------------------------------------------------------------------------------------------------  
**Handling Advanced User Interactions and Child windows, Frames**

**ActionsDemo.py**

import time  
  
from selenium import webdriver  
from selenium.webdriver import ActionChains  
from selenium.webdriver.chrome.service import Service  
from selenium.webdriver.common.by import By  
  
service\_obj = Service("/Users/Mujeeb Rahaman/Downloads/PythonDocs/chromedriver-win64/chromedriver-win64/chromedriver.exe")  
driver = webdriver.Chrome(service=service\_obj)  
  
driver.get("https://rahulshettyacademy.com/AutomationPractice/")  
action = ActionChains(driver)  
# action.double\_click(driver.find\_element(By.))  
# action.context\_click()  
# action.drag\_and\_drop()  
action.move\_to\_element(driver.find\_element(By.ID,"mousehover")).perform()  
# action.context\_click(driver.find\_element(By.LINK\_TEXT,"Top")).perform()  
action.move\_to\_element(driver.find\_element(By.LINK\_TEXT,"Reload")).click().perform()

**ChildWindow.py**

from selenium import webdriver  
from selenium.webdriver.chrome.service import Service  
from selenium.webdriver.common.by import By  
  
service\_obj = Service("/Users/Mujeeb Rahaman/Downloads/PythonDocs/chromedriver-win64/chromedriver-win64/chromedriver.exe")  
driver = webdriver.Chrome(service=service\_obj)  
  
driver.get("https://the-internet.herokuapp.com/windows")  
  
driver.find\_element(By.LINK\_TEXT,"Click Here").click()  
windowsOpened = driver.window\_handles  
  
driver.switch\_to.window(windowsOpened[1])  
print(driver.find\_element(By.TAG\_NAME,"h3").text)  
driver.close()  
driver.switch\_to.window(windowsOpened[0])  
assert "Opening a new window" == driver.find\_element(By.TAG\_NAME,"h3").text

**ChildWindowAssignment.py**

from selenium import webdriver  
from selenium.webdriver.chrome.service import Service  
from selenium.webdriver.common.by import By  
from selenium.webdriver.support import expected\_conditions  
from selenium.webdriver.support.wait import WebDriverWait  
  
service\_obj = Service("/Users/Mujeeb Rahaman/Downloads/PythonDocs/chromedriver-win64/chromedriver-win64/chromedriver.exe")  
driver = webdriver.Chrome(service=service\_obj)  
driver.implicitly\_wait(4)  
  
driver.get("https://rahulshettyacademy.com/loginpagePractise/")  
driver.find\_element(By.CSS\_SELECTOR, ".blinkingText").click()  
windowsOpened = driver.window\_handles  
  
driver.switch\_to.window(windowsOpened[1])  
message = driver.find\_element(By.CSS\_SELECTOR, ".red").text  
var = message.split("at")[1].strip().split(" ")[0]  
driver.close()  
driver.switch\_to.window(windowsOpened[0])  
driver.find\_element(By.ID, "username").send\_keys(var)  
driver.find\_element(By.ID, "password").send\_keys(var)  
driver.find\_element(By.CSS\_SELECTOR, "#signInBtn").click()  
wait = WebDriverWait(driver,10)  
wait.until(expected\_conditions.visibility\_of\_element\_located((By.CSS\_SELECTOR, ".alert-danger")))  
print(driver.find\_element(By.CSS\_SELECTOR, ".alert-danger").text)

**FramesDemo.py**

from selenium import webdriver  
from selenium.webdriver.chrome.service import Service  
from selenium.webdriver.common.by import By  
  
service\_obj = Service("/Users/Mujeeb Rahaman/Downloads/PythonDocs/chromedriver-win64/chromedriver-win64/chromedriver.exe")  
driver = webdriver.Chrome(service=service\_obj)  
driver.implicitly\_wait(2)  
  
driver.get("https://the-internet.herokuapp.com/iframe")  
driver.switch\_to.frame("mce\_0\_ifr")  
driver.find\_element(By.ID,"tinymce").clear()  
driver.find\_element(By.ID,"tinymce").send\_keys("I am able to automate frames")  
driver.switch\_to.default\_content()  
print(driver.find\_element(By.CSS\_SELECTOR,"h3").text)

------------------------------------------------------------------------------------------------------------------------------------------  
**Selenium Python Miscellaneous**

**Miscellaneous.py**

from selenium import webdriver  
from selenium.webdriver.chrome.service import Service  
  
chrome\_options = webdriver.ChromeOptions()  
chrome\_options.add\_argument("headless")  
chrome\_options.add\_argument("--ignore-certificate-errors")  
service\_obj = Service("/Users/Mujeeb Rahaman/Downloads/PythonDocs/chromedriver-win64/chromedriver-win64/chromedriver.exe")  
driver = webdriver.Chrome(service=service\_obj, options=chrome\_options)  
  
driver.get("https://rahulshettyacademy.com/AutomationPractice/")  
  
  
driver.execute\_script("window.scrollBy(0,document.body.scrollHeight)")  
driver.get\_screenshot\_as\_file("Screen.png")

**SortedTables.py**

from selenium import webdriver  
from selenium.webdriver.chrome.service import Service  
from selenium.webdriver.common.by import By  
  
service\_obj = Service("/Users/Mujeeb Rahaman/Downloads/PythonDocs/chromedriver-win64/chromedriver-win64/chromedriver.exe")  
driver = webdriver.Chrome(service=service\_obj)  
driver.get("https://rahulshettyacademy.com/seleniumPractise/#/offers")  
browserSortedveggies = []  
# click on column header  
driver.find\_element(By.XPATH,"//span[text()='Veg/fruit name']").click()  
# collect all veggie names -> BrowserSortedveggieList  
veggieWebElements = driver.find\_elements(By.XPATH,"//tr/td[1]")  
for ele in veggieWebElements:  
 browserSortedveggies.append(ele.text)  
  
originalBrowserSortedList = browserSortedveggies.copy()  
# Sort this BrowserSortedveggieList => newSortedList  
browserSortedveggies.sort()  
# BrowserSortedveggieList == newSortedList  
assert browserSortedveggies == originalBrowserSortedList

**ChromeOptionsDemo.py**

from selenium import webdriver  
from selenium.webdriver.chrome.service import Service  
from selenium.webdriver.common.by import By  
  
chrome\_options = webdriver.ChromeOptions()  
chrome\_options.add\_argument("--start-maximized")  
chrome\_options.add\_argument("headless")  
chrome\_options.add\_argument("--ignore-certificate-errors")  
  
  
service\_obj = Service("/Users/Mujeeb Rahaman/Downloads/PythonDocs/chromedriver-win64/chromedriver-win64/chromedriver.exe")  
driver = webdriver.Chrome(service=service\_obj,options=chrome\_options)  
  
driver.get("https://rahulshettyacademy.com/angularpractice/")  
print(driver.title)

------------------------------------------------------------------------------------------------------------------------------------------  
**End to End Practice Project with Complete methods in Selenium**

**E2eTest.py**

from selenium import webdriver  
from selenium.webdriver.chrome.service import Service  
from selenium.webdriver.common.by import By  
from selenium.webdriver.support import expected\_conditions  
from selenium.webdriver.support.wait import WebDriverWait  
  
service\_obj = Service("/Users/Mujeeb Rahaman/Downloads/PythonDocs/chromedriver-win64/chromedriver-win64/chromedriver.exe")  
driver = webdriver.Chrome(service=service\_obj)  
driver.implicitly\_wait(5)  
  
driver.get("https://rahulshettyacademy.com/angularpractice/")  
  
driver.find\_element(By.CSS\_SELECTOR,"a[href\*='shop']").click()  
# a[href\*='shop'] -css selectir or //a[contains(@href,'shop')] -xpath  
  
products = driver.find\_elements(By.XPATH,"//div[@class='card h-100']")  
  
for product in products:  
 productName = product.find\_element(By.XPATH,"div/h4/a").text  
 if productName == "Blackberry":  
 product.find\_element(By.XPATH,"div/button").click()  
 break  
  
driver.find\_element(By.CLASS\_NAME,"navbar-toggler-icon").click()  
driver.find\_element(By.CSS\_SELECTOR,"a[class$='btn-primary']").click()  
driver.find\_element(By.XPATH,"//button[@class='btn btn-success']").click()  
driver.find\_element(By.ID,"country").send\_keys("India")  
  
wait = WebDriverWait(driver,10)  
wait.until(expected\_conditions.visibility\_of\_element\_located((By.LINK\_TEXT,"India")))  
driver.find\_element(By.LINK\_TEXT,"India").click()  
  
  
driver.find\_element(By.CLASS\_NAME,"checkbox-primary").click()  
[driver.find\_element(By.XPATH,"//input[@type='submit']").click](mailto:driver.find_element(By.XPATH,"//input[@type='submit']").click)()  
successMessage = driver.find\_element(By.CLASS\_NAME,"alert-success").text  
assert "Success! Thank you! Your order will be delivered in next few weeks :-)." in successMessage  
driver.close()  
  
------------------------------------------------------------------------------------------------------------------------------------------  
**Framework Design Plan**

**Part 1: - Pytest Unit Testing Framework**

**Part 2: - Understanding Logging and HTML Reports**

**Part 3: - Implement Selenium Python Framework from Scratch (PageObject DesignPattern)**

**Part 4: - Data Driven Framework with Excel to Selenium Python Tests**

**Part 5: - GIT Version Control**

------------------------------------------------------------------------------------------------------------------------------------------  
**Plan**

* **Install Pytest – pip install pytest**
* **Naming conventions to follow for Pytest Test**
* **Running Pytest from command Line and PyCharm**
* **Running selected test files using Pytest**
* **Running selected test Methods based on matching keywords**
* **Pytest Tags mechanism to run test based on functionality**
* **Failing and Skipping tests with Annotations using Pytest**
* **What are fixtures and importance of their Hooks in Pytest**
* **How Fixtures can be configured in Conftest file for better readability**
* **Different scopes of fixtures and their related annotations to setup Pre and Post Conditions of the Test**
* **How Parameterization can be achieved for Tests with multiple sets of data**
* **How to pass commabd Line Arguments into PyTests**
* **Html report Generation for Pytests Execution**

**pip install pytest-html – to install html reports library**

**pytest –html=report.html -- to run test**

------------------------------------------------------------------------------------------------------------------------------------------

**Part 1: - Pytest – Unit Testing Framework of Python**

**pytestDemo --package**

**Test\_demo1.py**

# Any pytest file should start with test\_ or end with \_test pytest method name should start with test Any code should  
# be wrapped in method only Method name should have sense -k Stands for method names execution , -s login in output,  
# -v stands for more info metadata You can run specific file py.test <filename> You can mark (tag) test  
# @pytest.mark.smoke and then run with -m You can skip tests with @pytest.mark.skip You can run the test without  
# loggin in document if fail with @pytest.mark.xfail Fixtures are used as setup and tear down methods for test cases-  
# conftest file to generalize fixture and make it available to all test cases  
# datatdriven and parameterization can be done with return statement in tuple format  
# When you define fixture scope to class only, it will run once before class is initiated and at the end  
# To get reports during command line run use py.test --html=report.html  
  
  
  
import pytest  
  
  
@pytest.mark.smoke  
def test\_firstProgram(setup):  
 print("Hello")  
  
@pytest.mark.xfail  
def test\_SecondGreetCreditCard():  
 print("Good Morning")  
  
def test\_crossBrowser(crossBrowser):  
 print(crossBrowser[1])

**Test\_demo2.py**

import pytest  
  
  
@pytest.mark.smoke  
@pytest.mark.skip  
def test\_firstProgram():  
 msg = "Hello" #operations  
 assert msg == "Hi","Test Failed because string do not match"  
  
def test\_SecondCreditCard():  
 a = 4  
 b = 6  
 assert a + 2 == 6,"Addition do not match"  
  
  
def test\_fixtureDemo(setup):  
 print("I will execute steps in fixture method")

**Test\_fixtureDemo.py**

import pytest  
  
  
@pytest.mark.usefixtures("setup")  
class TestExample:  
  
 def test\_fixtureDemo(self):  
 print("I will execute steps in fixture method")  
  
 def test\_fixtureDemo1(self):  
 print("I will execute steps in fixture method")  
  
 def test\_fixtureDemo2(self):  
 print("I will execute steps in fixture method")  
  
 def test\_fixtureDemo3(self):  
 print("I will execute steps in fixture method")

**Conftest.py**

import pytest  
  
  
@pytest.fixture(scope="class")  
def setup():  
 print("I will be executed first")  
 yield  
 print("I will be executed last")  
  
@pytest.fixture()  
def dataLoad():  
 print("User profile data is being created")  
 return ["[Mujeeb","Rahaman","mujeeb.rahaman@cognine.com](mailto:Mujeeb","Rahaman","mujeeb.rahaman@cognine.com)"]  
  
  
@pytest.fixture(params=[("chrome","Mujeeb","Rahaman"), ("firefox","Mujeeb"),("IE","Rahaman")])  
def crossBrowser(request):  
 return request.param

**Test\_fixturesData.py**

import pytest  
  
  
@pytest.mark.usefixtures("dataLoad")  
class TestExample2:  
  
 def test\_editProfile(self, dataLoad):  
 print(dataLoad[0])  
 print(dataLoad[2])  
  
**------------------------------------------------------------------------------------------------------------------------------------------**  
**Part – 2 Logging and Generating HTML Repots in Python Framework**

**LOGS**

**Debug**

**Info**

**Warning**

**Error**

**Critical**

**Test\_logging.py**

import logging  
  
def test\_logging():  
 logger = logging.getLogger(\_\_name\_\_)  
  
 fileHandler = logging.FileHandler('logfile.log')  
 formatter = logging.Formatter("%(asctime)s : %(levelname)s : %(name)s : %(message)s")  
 fileHandler.setFormatter(formatter)  
  
 logger.addHandler(fileHandler) #fileHandler object  
  
 logger.setLevel(logging.DEBUG)  
 logger.debug("A debug statement is executed")  
 logger.info("Information statement")  
 logger.debug("A debug statement is executed")  
 logger.warning("Something is in warning mode")  
 logger.error("A Major error had happened")  
 logger.critical("Critical Issue")

**Test\_fixturesData.py**

import pytest  
  
from pytestDemo.BaseClass import BaseClass  
  
  
@pytest.mark.usefixtures("dataLoad")  
class TestExample2(BaseClass):  
  
 def test\_editProfile(self, dataLoad):  
 log = self.getLogger()  
 log.info(dataLoad[0])  
 log.info(dataLoad[2])  
 print(dataLoad[0])  
  
 print(dataLoad[2])

**BaseClass.py**

import inspect  
import logging  
  
  
class BaseClass:  
 def getLogger(self):  
 loggerName = inspect.stack()[1][3]  
 logger = logging.getLogger(loggerName)  
  
 fileHandler = logging.FileHandler('logfile.log')  
 formatter = logging.Formatter("%(asctime)s : %(levelname)s : %(name)s : %(message)s")  
 fileHandler.setFormatter(formatter)  
  
 logger.addHandler(fileHandler) # fileHandler object  
  
 logger.setLevel(logging.DEBUG)  
 return logger

**Logfile.log**

2023-09-18 16:06:04,660 : INFO : *pytestDemo.BaseClass* : Mujeeb  
2023-09-18 16:06:04,660 : INFO : *pytestDemo.BaseClass* : [*mujeeb.rahaman*@*cognine.com*](mailto:mujeeb.rahaman@cognine.com)*2023-09-18 16:06:24,019 : INFO : test\_editProfile : Mujeeb*  
*2023-09-18 16:06:24,019 : INFO : test\_editProfile :* [*mujeeb.rahaman*@*cognine.com*](mailto:mujeeb.rahaman@cognine.com)*2023-09-18 16:07:20,626 : INFO : test\_editProfile : Mujeeb*  
*2023-09-18 16:07:20,626 : INFO : test\_editProfile :* [*mujeeb.rahaman*@*cognine.com*](mailto:mujeeb.rahaman@cognine.com)*2023-09-18 16:07:20,631 : DEBUG : pytestDemo.test\_logging* : A debug statement is executed  
2023-09-18 16:07:20,631 : INFO : *pytestDemo.test\_logging* : Information statement  
2023-09-18 16:07:20,631 : DEBUG : *pytestDemo.test\_logging* : A debug statement is executed  
2023-09-18 16:07:20,631 : WARNING : *pytestDemo.test\_logging* : Something is in warning mode  
2023-09-18 16:07:20,631 : ERROR : *pytestDemo.test\_logging* : A Major error had happened  
2023-09-18 16:07:20,632 : CRITICAL : *pytestDemo.test\_logging* : Critical Issue

**Report.html**

**------------------------------------------------------------------------------------------------------------------------------------------**  
**Part – 3 Develop End to End Selenium Python Framework from scratch**

* **Standards of writing selenium test in framework**
* **Creating Browser invocation Fixtures in conftest.py**
* **Setting us Base Class to hold all common utilities**
* **Inheriting Base Class to all test to remove Fixture redundant Code**
* **Passing Command Line Options to select browser at run time**
* **Implementing Page Object Mechanism**
* **Smarter way of returning Page Objects from Navigation Methods**
* **Creating Selenium WebDriver Custom Utilities in Base Class**
* **Parameterizing WebDriver tests with Multiple Data Sets**
* **Implementing Loggin feature to WebDriver test**
* **Test Execution HTML reporting**
* **Automatic Screenshot Capture on Test Failures**
* **Integrating Selenium Python Framework to Jenkins CI tool with Jenkin build Parameterization**
* **GitHub Basics for Project version control**

**Refer github to get the project**

**------------------------------------------------------------------------------------------------------------------------------------------**  
**Part – 4 Integrating Framework to Jenkins and setup Various Job Parameters**

**Download jenkins.war**

**java -jar jenkins.war --httpPort = 9090 -- to start jenkins run the command from where the jenkins.war file is present**

java -jar jenkins.war --enable-future-java --httpPort=xxxx

**Setup all jenkins job and run**

**------------------------------------------------------------------------------------------------------------------------------------------**  
**Part – 5 Read/Write data from Excel to Selenium Python Framework**

**ExcelDemo.py**

import openpyxl  
  
book = openpyxl.load\_workbook("C:\\Users\\Mujeeb Rahaman\\Downloads\\PythonDocs\\Pythondemo.xlsx")  
  
sheet = book.active  
Dict = {}  
cell = sheet.cell(row=1,column=2)  
print(cell.value)  
  
sheet.cell(row=2,column=2).value = "Mujeeb"  
  
print(sheet.cell(row=2,column=2).value)  
  
print(sheet.max\_row)  
print(sheet.max\_column)  
  
print(sheet['A5'].value)  
  
for i in range(1,sheet.max\_row+1): #To get rows  
 if sheet.cell(row=i,column=1).value == "Testcase2":  
 for j in range(2,sheet.max\_column+1): #To get columns  
 [#Dict["email"]="alia.bhatt@gmail.com](mailto:#Dict["email"]="alia.bhatt@gmail.com)"  
 Dict[sheet.cell(row=1,column=j).value]= sheet.cell(row=i,column=j).value  
  
print(Dict)

**HomePageData.py**

import openpyxl  
  
  
class HomePageData():  
  
 test\_HomePage\_data = [[{"firstname":"Mujeeb","email":"mujeeb.rahaman@cognine.com","password":"Mujeeb@123","gender":"Male","myMessage":"HelloWorld!"},{"firstname":"Alia","email":"alia.bhatt@gmail.com","password":"Alia@123","gender":"Female","myMessage":"MyWorld](mailto:{"firstname":"Mujeeb","email":"mujeeb.rahaman@cognine.com","password":"Mujeeb@123","gender":"Male","myMessage":"HelloWorld!"},{"firstname":"Alia","email":"alia.bhatt@gmail.com","password":"Alia@123","gender":"Female","myMessage":"MyWorld)!!"}]  
  
 @staticmethod  
 def getTestData(test\_case\_name):  
 Dict = {}  
 book = openpyxl.load\_workbook("C:\\Users\\Mujeeb Rahaman\\Downloads\\PythonDocs\\Pythondemo.xlsx")  
 sheet = book.active  
  
 for i in range(1, sheet.max\_row + 1): # To get rows  
 if sheet.cell(row=i, column=1).value == test\_case\_name:  
 for j in range(2, sheet.max\_column + 1): # To get columns  
 # [Dict["email"]="alia.bhatt@gmail.com](mailto:Dict["email"]="alia.bhatt@gmail.com)"  
 Dict[sheet.cell(row=1, column=j).value] = sheet.cell(row=i, column=j).value  
  
 return[Dict]

**Test\_HomePage.py**

import pytest  
  
from TestData.HomePageData import HomePageData  
from pageObjects.HomePage import HomePage  
from tests.utilitites.BaseClass import BaseClass  
  
  
class TestHomePage(BaseClass):  
  
 def test\_formSubmission(self,getData):  
  
 log = self.getLogger()  
 homePage = HomePage(self.driver)  
 log.info("First Name is " + getData["firstname"])  
 homePage.getName().clear()  
 homePage.getName().send\_keys(getData["firstname"])  
 homePage.getEmail().send\_keys(getData["email"])  
 homePage.getPassword().send\_keys(getData["password"])  
 homePage.getStudent().click()  
 homePage.getIcreamChecBox().click()  
 self.selectOptionByText(homePage.getGender(),getData["gender"])  
 homePage.getHomeSubmitButton().click()  
 message = homePage.getAlertMessage().text  
 print(message)  
 assert "Success" in message  
 homePage.getInputText().send\_keys(getData["myMessage"])  
 homePage.getInputText().clear()  
 self.driver.refresh()  
  
 # @pytest.fixture(params = HomePageData.test\_HomePage\_data)  
 @pytest.fixture(params = HomePageData.getTestData("Testcase2"))  
 def getData(self,request):  
 return request.param