CHAPTER 1
INTRODUCTION

Adidas AG (German production stylized in all lowercase since 1949) is a German athletic apparel and footwear corporation headquartered in Herzogenaurach, Bavaria, Germany. It is the largest sportswear manufacturer in Europe, and the second largest in the world, after Nike.

It is the holding company for the Adidas Group, which also owns an 8.33% stake of the football club Bayern München, and Runtastic, an Austrian fitness technology company. Adidas's revenue for 2018 was listed at €21.915 billion.

Christoph Dassler was a worker in ashoe factory, while his wife Pauline ran a small laundry in the Franconian town of Herzogenaurach, 20 km (12.4 mi) from the city of Nuremberg. Afterleaving school, their son,Rudolf Dassler, joined his father at the shoe factory.

When he returnedfrom fighting in WorldWar I, Rudolf was trained as a salesman at a porcelain factory, and laterin a leather trading business in Nuremberg.

In July 1924, Rudolf and his younger brother, Adolf, nicknamed "Adi", founded a shoe factory. They named the new business "Gebrüder Dassler Schuhfabrik" (*Dassler Brothers Shoe Factory*) which was the only business at the time that manufactured sports shoes. The pair started their venture in their mother's laundry.

At the time, electricity supplies in the town wereunreliable, and the brothers sometimes had to use pedal power from a stationary bicycle to runtheir equipment. In 1927, they moved into a separate building.

The brothers drove from Bavaria to the 1936 Summer Olympics in Berlin with a suitcase full of spikes and persuaded United States sprinter Jesse Owens to use them, the first sponsorship for an African American. Owens won four gold medals. Business boomed; the Dasslers were selling 200,000 pairs of shoes annually before World War II.

Both brothers joined the Nazi Party, but Rudolf was a keen Nazi, who applied to join, and was accepted into the Gestapo; they produced boots for the Wehrmacht. A growing rift between the brothers reached a breaking point during a 1943 Allied bomb attack. Adi and his wife climbed into a bomb shelter that Rudolf and his family were already in. "Here are the bloody bastards again," Adi remarked, apparently referring to the Allied war planes, but Rudolf, due to his apparent insecurity, was convinced his brother meant him and his family. When Rudolf was later picked up by American soldiers and accused of being a member of the Waffen SS, he was convinced that his brother had turned him.

1.1 WHAT IS SOFTWARE TESTING

Software testing is a critical process in software development aimed at evaluating the functionality, performance, usability, and security of a software application. Its primary goal is to ensure that the software meets specified requirements and operates correctly under various conditions. By creating and executing test cases, testers verify the software's behaviour, identify defects or bugs, and validate its reliability before release. Testing can be automated or manual and includes techniques like unit testing, integration testing, regression testing, and user acceptance testing. Ultimately, effective testing helps in delivering high-quality, user-friendly, and secure software products, reducing the risk of failures in production and enhancing overall user satisfaction.

1.2 TYPES OF SOFTWARE TESTING

There are several types of software testing, each serving different purposes in ensuring the quality and reliability of software products.
Here are some common types,

UNIT TESTING

Unit testing focuses on verifying the smallest parts of an application, such as functions or methods, in isolation from the rest of the system. The main aim is to validate that each unit of the software performs as expected.
Developers typically perform unit testing using automated test frameworks like JUnit for Java, NUnit for .NET, and PyTest for Python.

Advantages:

• Helps identify and fix bugs early in the development process.

• Facilitates easier debugging and code refactoring.

• Ensures that individual components work correctly before integration.

INTEGRATION TESTING

Integration testing examines how different software modules work together as a group It ensures that integrated components function correctly as a cohesive unit.

Common approaches include top-down (testing higher-level components first), bottom-up (testing lower-level components first), and hybrid integration.

Advantages:

• Identifies interface issues between modules.

• Detects integration errors early.

• Improves system reliability by ensuring components interact as expected.

SYSTEM TESTING

System testing involves testing the complete and integrated software system as a whole to ensure it meets specified requirements. It evaluates the system's compliance with functional and non-functional requirements, including performance, security, and usability.

Tests are conducted in an environment similar to the production environment to simulate real-world usage scenarios.

Advantages:

- Ensures the software meets end-user expectations and business goals.
- Provides comprehensive validation of the entire system.

ACCEPTANCE TESTING

Acceptance testing, also known as user acceptance testing (UAT), validates the software's readiness for deployment from an end-user's perspective. It confirms that the software meets business requirements and functions as intended in the user's operational environment. Types:

• **ALPHA TESTING:** Internal testing by the organization.

• **BETA TESTING:** External testing by a selected group of users.

Advantages:

• Ensures stakeholder satisfaction and validates business objectives.

• Identifies final adjustments needed before release.

REGRESSION TESTING

Regression testing verifies that recent changes or additions to the software haven't adversely affected existing functionality. It ensures that previously working features remain intact after code modifications, bug fixes, or enhancements. Approach: Automated regression tests are often employed to re-run test cases efficiently and consistently.

Advantages:

• Prevents regression bugs from reaching production.

• Maintains software quality over time.

• Supports continuous integration and delivery practices.

PERFORMANCE TESTING

Performance testing evaluates the software's responsiveness, stability, and scalability under expected workload conditions. It identifies performance bottlenecks, measures response times, and assesses resource usage (CPU, memory, network) to ensure optimal system performance.

Types:

• Load testing: Measuring system performance under normal and peak loads.

• Stress testing: Testing system behavior under extreme conditions.

• Scalability testing: Testing system ability to scale with increased load.

Advantages:

• Optimizes system performance.

• Enhances user experience.

• Supports capacity planning and scalability efforts.

SECURITY TESTING

Security testing identifies vulnerabilities and weaknesses in the software's security mechanisms. It ensures that sensitive data is protected from unauthorized access, breaches, and malicious attacks.

Approach:

• Penetration Testing: Ethical hacking to find security flaws.

• Vulnerability Scanning: Automated tools to scan for known vulnerabilities.

• Security Audits: Comprehensive reviews of security policies and procedures.

Advantages:

• Mitigates security risks.

• Safeguards user data.

• Enhances overall system trustworthiness and compliance.

USABILITY TESTING

Usability testing assesses the software's ease of use and intuitiveness from the end-user's perspective.

It ensures that the software interface is user-friendly, efficient, and meets usability standards. Conducting usability studies, user feedback surveys, and heuristic evaluations.

Advantages:

• Improves user satisfaction.

• Reduces training time and support costs.

• Enhances user adoption rates.

1.3 ABOUT THE PROJECT

This project involves the automation of the ADIDAS website using Python, leveraging various Python libraries and the Selenium WebDriver. The primary goal of this automation is to streamline and enhance the user experience by performing repetitive tasks efficiently and accurately.

Python is chosen for its simplicity and extensive support for automation tasks. **Selenium** is used as the core library to interact with web elements on the ADIDAS website, providing the capability to simulate user actions such as clicking, typing, and navigating through web pages.

This documentation details the steps and methodologies used to achieve the automation, including setting up the environment, writing scripts to automate specific tasks, handling dynamic web elements, and managing exceptions.

The project demonstrates the power of combining Python's programming capabilities with Selenium's web automation features tocreate a robust and scalable automation solution.

1.4 OBJECTIVES

AUTOMATE BROWSING AND SEARCHING

Develop scripts to automatically browse and search for items on ADIDAS, reducing the manual effort required for these tasks.

• **ENHANCE EFFICIENCY**

Streamline repetitive actions such as logging in, adding items to the cart, and checking out, to save time and improve user efficiency.

• **ENSURE ACCURACY**

Implement error handling and validation mechanisms to ensure accurate interactions with the eBay website, minimizing the risk of mistakes during automated tasks.

• **USER-FRIENDLY INTERFACE**

Design the automation script to be user-friendly, allowing even those with minimal technical knowledge to utilize its features effectively.

• **MAINTAINABILITY AND SCALABILITY**

Write clean, modular, and well-documented code to ensure the project is easy to maintain and can be scaled or modified to accommodate future requirements or additional functionalities.

• **PERFORMANCE MONITORING**

Include features to monitor the performance of the automation process, identifying potential bottlenecks and areas for improvement.

1.5 TECHNOLOGIES USED

• **PROGRAMMING LANGUAGE**

PYTHON : The primary language for developing the automation scripts due to its simplicity, readability, and extensive library support.

• **WEB AUTOMATION FRAMEWORK**

SELENIUM : A powerful tool for automating web browsers, allowing interaction with the ADIDAS website as a human would.

• LIBRARIES AND MODULES

SELENIUM WEBDRIVER : To control the browser and perform actions like clicking, typing, and navigating.

LOGGING : Python's built-in logging module for tracking the script's actions and errors.

• INTERFACE

CLI (COMMAND-LINE INTERFACE) : Basic interaction through the terminal or command prompt.

• DEVELOPMENT ENVIRONMENT

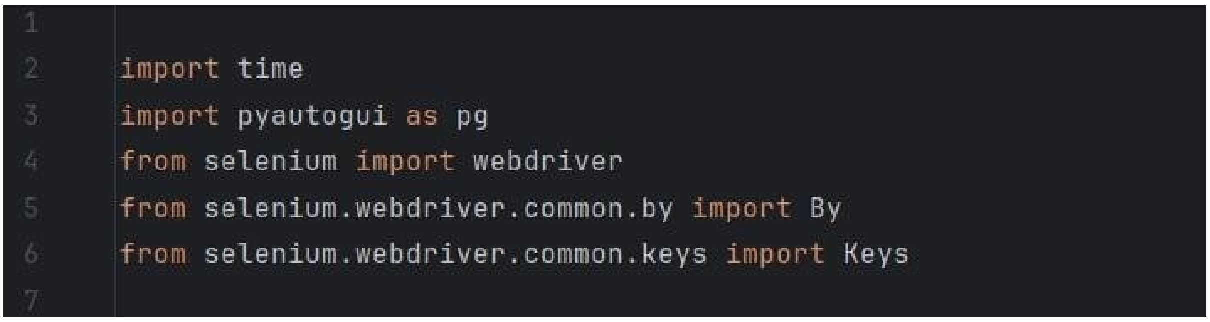
VSCODE: A versatile and lightweight code editor.

PYCHARM: A code will run using PyCharm also, we can use vs code and also PyCharm to code editor.

CHAPTER 2
IMPLEMENTATION

2.1 IMPORTING NECESSARY LIBRRIES

The scripts starts by importing the necessary libraries:



```python

import time
import pyautogui as pg
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.common.keys import Keys

```

The libraries imported are:

Selenium: The main Selenium library for automating web browsers.

Webdriver: The webdriver library for interacting with web browsers.

Time: A built-in Python library for working with time-related functions.

By: A library for **specifying** the type of element to find on a web page (e.g., by ID, by class name, etc.).

Select: A library for working with dropdown menus.

INSTALLATION REQUIRED FOR THIS LIBRARIES:

In terminal which is in vs code editor or PyCharm install by using,

- Pip install selenium
- Pip install webdriver-manager
- Pip install pytest
- Pip install selenium pyautogui

2.2 CREATING CHROME OPTIONS AND DRIVER INSTANCE

```
def adidas():
   # Initialize Chrome WebDriver
   driver = web driver. Chrome ()
   driver. maximize window()
```

WebDriver Manager automatically downloads and installs the correct Chrome Driver executable based on the system's architecture and Chrome version.

The script then maximizes the browser window using the driver.maximize_window() method.

OPENING EBAY WEBSITE AND MAXIMIZING THE BROWSER WINDOW

The script opens the eBay website using the driver.get() method:

```
try:
   # Navigate to adidas login page
   driver.get("https://in.adidas.com/in/en/account/login?action=login")
   time. sleep (8)
```

2.3 LOGIN OR REGISTER TO ADIDAS

The script LOGIN OR REGISTER to ADIDAS by giving the phone number and otp, after that to register, entering the first name, last name, email, and set the password, clicking the Continue button, and then again login to puma by entering the phone number or email and password, and clicking the Submit button again: The script uses WebDriver Wait to wait for the elementsto become clickable or visible before interacting with them.

If you need to login or register on Adidas, you can follow these steps:

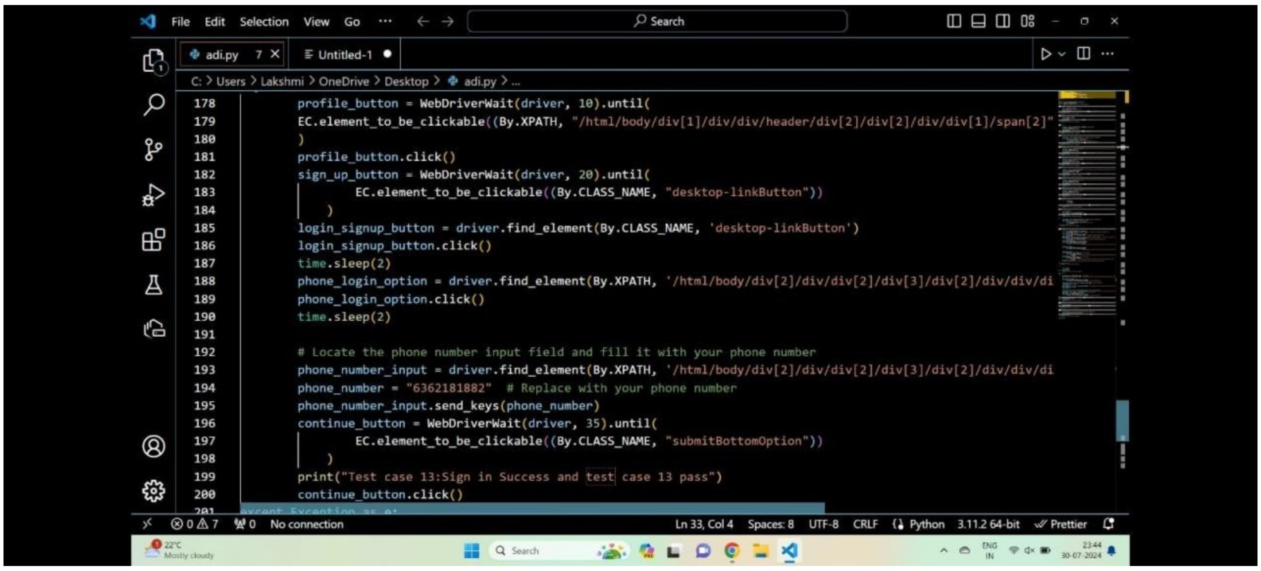
For Login:

1. **Visit the Adidas Website**: Go to the official Adidas website.
2. **Locate the Login Button**: Usually found in the top right corner of the homepage.
3. **Enter Credentials**: Input your email address and password.
4. **Click Login**: After entering your details, click the login button.

For Registration:

1. **Go to the Adidas Website**: Open the Adidas website.
2. **Find the Register Option**: This is often located near the login button.
3. **Fill Out the Form**: Provide your email address, create a password, and enter any other required details.
4. **Submit**: Click the register or create account button to complete the process.

Make sure to use a strong password and keep your account details secure!



```
        profile_button = WebDriverWait(driver, 10).until(
        EC.element_to_be_clickable((By.XPATH, "/html/body/div[1]/div/div/header/div[2]/div[2]/div/div[1]/span[2]"
        )
        profile_button.click()
        sign_up_button = WebDriverWait(driver, 20).until(
                EC.element_to_be_clickable((By.CLASS_NAME, "desktop-linkButton"))
            )
        login_signup_button = driver.find_element(By.CLASS_NAME, 'desktop-linkButton')
        login_signup_button.click()
        time.sleep(2)
        phone_login_option = driver.find_element(By.XPATH, '/html/body/div[2]/div/div[2]/div[3]/div[2]/div/div/di
        phone_login_option.click()
        time.sleep(2)

        # Locate the phone number input field and fill it with your phone number
        phone_number_input = driver.find_element(By.XPATH, '/html/body/div[2]/div/div[2]/div[3]/div[2]/div/div/di
        phone_number = "6362181882"  # Replace with your phone number
        phone_number_input.send_keys(phone_number)
        continue_button = WebDriverWait(driver, 35).until(
                EC.element_to_be_clickable((By.CLASS_NAME, "submitBottomOption"))
            )
        print("Test case 13:Sign in Success and test case 13 pass")
        continue_button.click()
```

CHAPTER 3

SOURCE CODE AND OUTPUTS

3.1 SOURCE CODE

```
import time
import pyautogui as pg
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.common.keys import Keys
def adidas():
    # Initialize Chrome WebDriver
    driver = web driver. Chrome ()
    driver. maximize window()

    try:
        # Navigate to adidas login page
        driver.get("https://in.adidas.com/in/en/account/login?action=login")
        time. sleep (8)

        # Enter phone number
        phone_entry =
driver.find_element(By.XPATH,'/html/body/div[1]/div[1]/main/div/div/div/div/div[2]/div/div/
div/div/form/div[1]/div[2]/input')
        phone_entry.send_keys("8747938311")
        time.sleep(5000)

        # Request OTP
        req_otp =
driver.find_element(By.XPATH,'/html/body/div[1]/div[1]/main/div/div/div/div/div[2]/div/div/
div/div/form/button/div/div')
        req_otp.click()
        time.sleep(1500)
        # Open search bar and search for a product
        search_open =
driver.find_element(By.XPATH,'/html/body/div[1]/div[1]/div[1]/nav/div/div/button[1]/div[2]/
div')
        search_open.click()
        time.sleep(3)

        search_bar = driver.find_element(By.NAME, 'Search')
        search_bar.send_keys('ADIDAS x one8 Core V2 Unisex Cap')
        time.sleep(2)
        search_bar.send_keys(Keys.ENTER)
        time.sleep(10)

        # Scroll using PyAutoGUI (for demonstration)
        pg.scroll(-400)

        # Click on the product link
        product_link = driver.find_element(By.PARTIAL_LINK_TEXT, 'ADIDAS x one8 CoreV2
Unisex Cap')
        product_link.click()
        time.sleep(10)

        # Scroll using PyAutoGUI (for demonstration)
        pg.scroll(-650)

        # Add to cart
        atc_button =
driver.find_element(By.XPATH,'/html/body/div[2]/div[1]/main/div/section/div[1]/section[2]/
div/div[7]/div[2]/button[1]')
        atc_button.click()
        time.sleep(6)

        # Keep the browser open until user interaction
        input("Enter a key to exit...")
```

```
  finally:
      # Close the browser session
      driver.quit()

if__name__== "_main_":
  adidas()
```

Selenium-based Test Suite for ADIDAS Website:

Test Case 01: Homepage and Navigation

1. **Verify Homepage Load**: Open the ADIDAS website. Confirm that the homepage loadswithout errors.
2. **Navigation Menu**: Click on each menu item (e.g., Men, Women, Kids, Sale). Ensure that the corresponding pages load correctly.
3. **Search Bar**: Enter a valid search query (e.g., “running shoes”). Verify that relevant products appear in the search results.

Test Case 02: Product Details Page (PDP)

1. **Product Availability**: Navigate to a product page. Check if product details (name, price, description) are displayed.
2. **Size and Color Selection**: Select different sizes and colors. Confirm that the product image updates accordingly.
3. **Add to Cart**: Click the “Add to Cart” button. Verify that the cart updates with the selected product.

Test Case 03: Shopping Cart and Checkout

1. **Cart Summary**: Go to the shopping cart. Validate the product details (name, price, quantity).
2. **Proceed to Checkout**: Click the “Checkout” button. Ensure that the checkout process begins.

Test Case 04: User Account and Login

1. **Login Page**: Navigate to the login page. Verify that the login form is displayed.
2. **Valid Credentials**: Enter valid login credentials. Confirm successful login.

3. **Invalid Credentials**: Attempt login with incorrect credentials. Check for appropriate error messages.

Test Case 05: Payment and Order Confirmation

1. **Payment Options**: Proceed to the payment page. Verify that payment options (credit card, PayPal, etc.) are available.
2. **Place Order**: Complete the payment process. Confirm that the order is placed successfully.
3. **Order Confirmation Email**: Check if an order confirmation email is received.

Test Case 06: Invalid Inputs

1. **Out-of-Stock Products**: Test what happens when a user tries to add an out-of-stock product to their cart. Verify that appropriate messages (e.g., "Product currently unavailable") are displayed.
2. **Search Bar**: Enter an invalid search query (e.g., special characters, excessively long query). Confirm that the website handles it gracefully.
3. **Login Credentials**: Test with incorrect or incomplete login credentials. Ensure proper error messages are shown.

Test Case 07: Boundary Conditions

1. **Quantity Limits**: Add the maximum allowed quantity of a product to the cart. Verify that the cart behaves correctly.
2. **Price Ranges**: Test products at different price points (e.g., very low-priced or high-priced items). Confirm that calculations (subtotal, tax, total) are accurate.

Test Case 08: Localization

Currency and Language: Change the website's currency (if supported). Ensure that prices update correctly. Switch the language and verify translations.

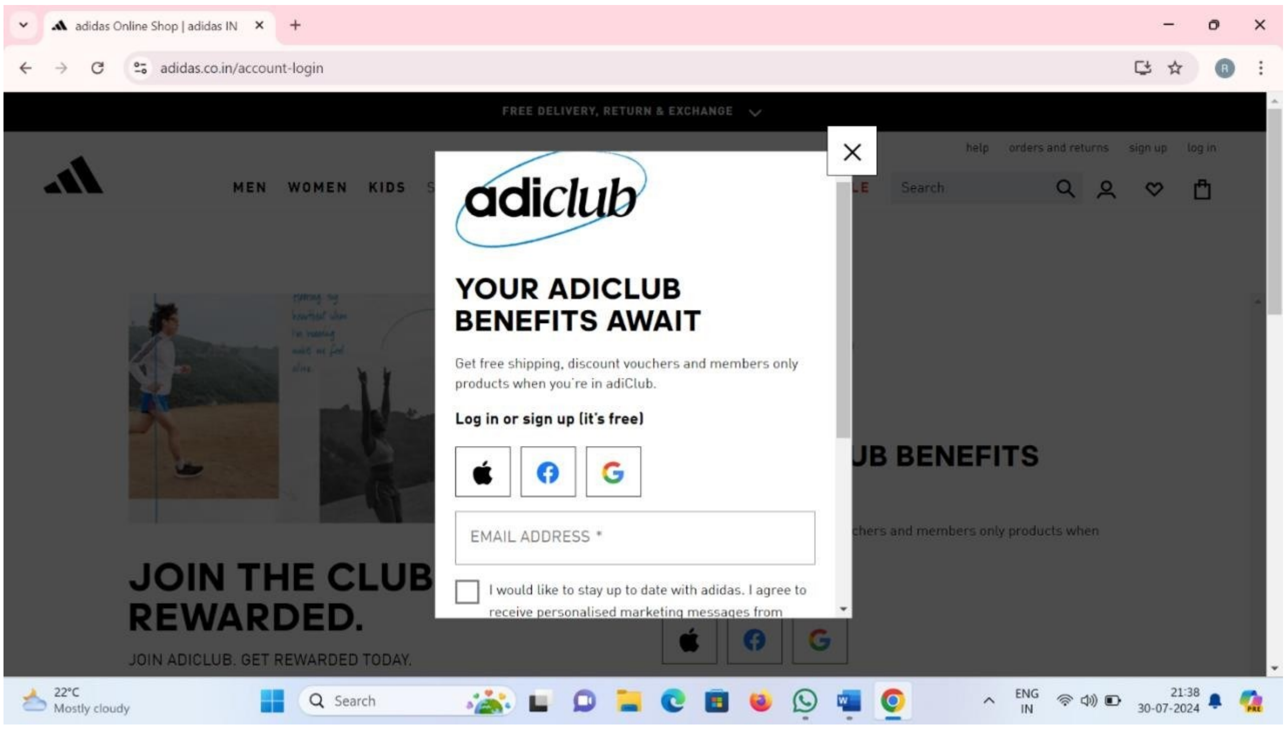
Test Case 09: Mobile Devices

1. **Responsive Design**: Access the website from various mobile devices (phones, tablets). Check if the layout adapts correctly.
2. **Touch Interactions**: Test touch gestures (swipe, pinch) on product images and buttons.

Test Case 10: Security

1. **Cross-Site Scripting (XSS)**: Attempt to inject malicious scripts into input fields. Confirm that the website sanitizes user inputs.
2. **Authentication Bypass**: Try accessing restricted pages without logging in. Ensure proper access controls are in place.

3.2 OUTPUTS



YOUR ADICLUB BENEFITS AWAIT

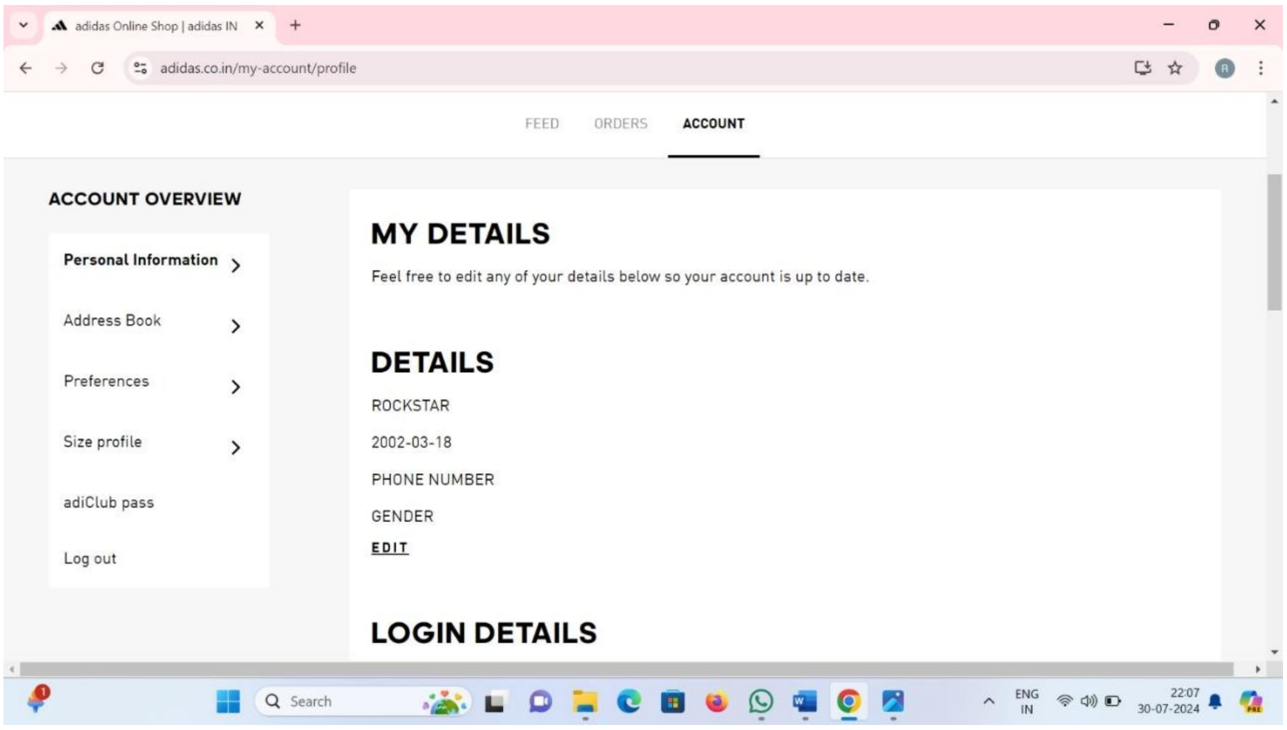
Get free shipping, discount vouchers and members only products when you're in adiClub.

Log in or sign up (it's free)

EMAIL ADDRESS *

☐ I would like to stay up to date with adidas. I agree to receive personalised marketing messages from

FIG 1.1: LOGIN PAGE



MY DETAILS

Feel free to edit any of your details below so your account is up to date.

DETAILS

ROCKSTAR

2002-03-18

PHONE NUMBER

GENDER

EDIT

LOGIN DETAILS

FIG 1.2: DASHBOARD OF MY ACCOUNT

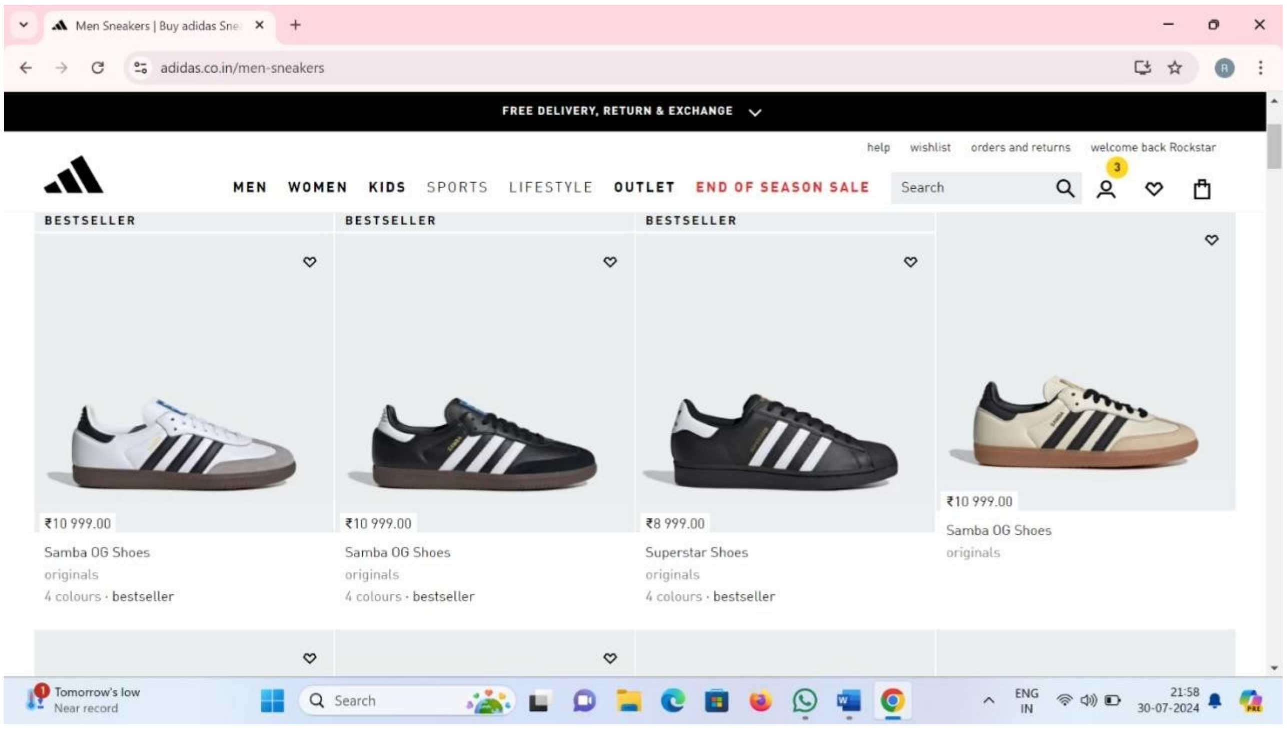


FIG 1.3: SCROLLING FUNCTION

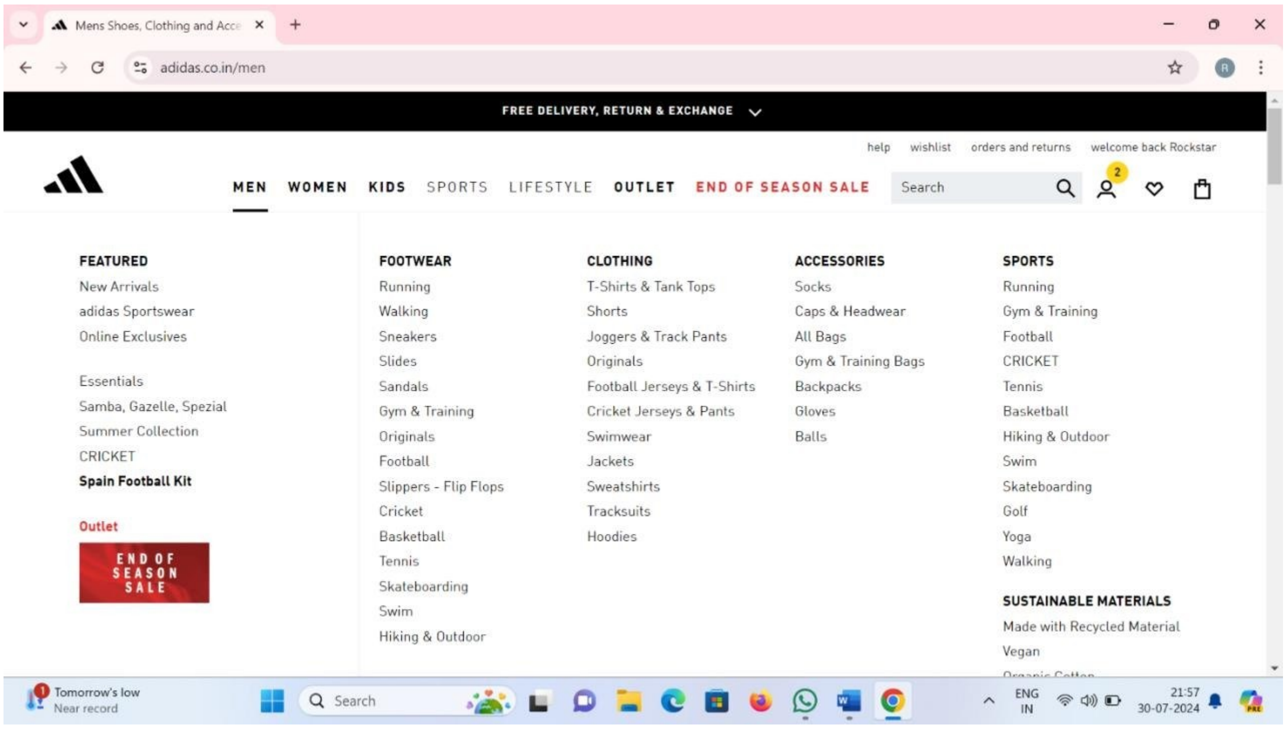
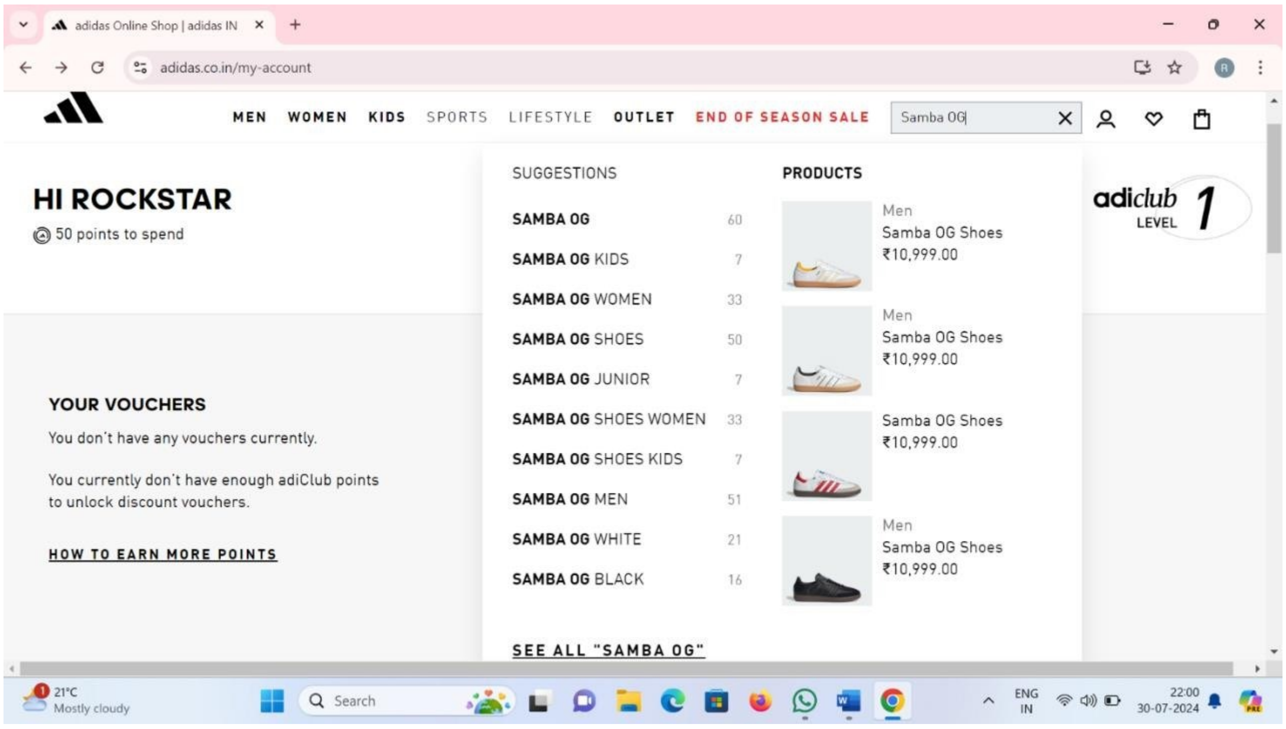


FIG 1.4: DETAILS OF SELECTED GROUP



HI ROCKSTAR

50 points to spend

YOUR VOUCHERS

You don't have any vouchers currently.

You currently don't have enough adiClub points to unlock discount vouchers.

HOW TO EARN MORE POINTS

SUGGESTIONS	
SAMBA OG	60
SAMBA OG KIDS	7
SAMBA OG WOMEN	33
SAMBA OG SHOES	50
SAMBA OG JUNIOR	7
SAMBA OG SHOES WOMEN	33
SAMBA OG SHOES KIDS	7
SAMBA OG MEN	51
SAMBA OG WHITE	21
SAMBA OG BLACK	16

PRODUCTS

Men Samba OG Shoes ₹10,999.00

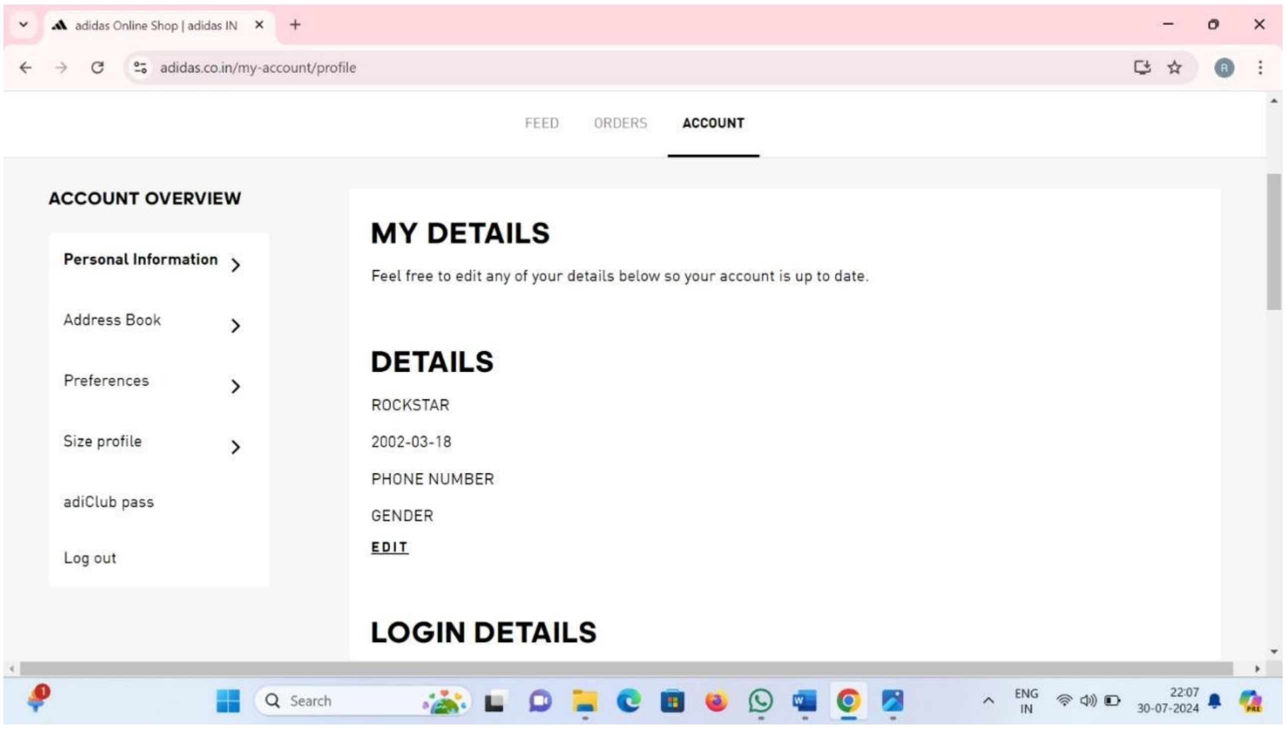
Men Samba OG Shoes ₹10,999.00

Samba OG Shoes ₹10,999.00

Men Samba OG Shoes ₹10,999.00

SEE ALL "SAMBA OG"

FIG 1.5: AUTOMATED SEARCH IN SEARCH BAR



ACCOUNT OVERVIEW

- Personal Information
- Address Book
- Preferences
- Size profile
- adiClub pass
- Log out

MY DETAILS

Feel free to edit any of your details below so your account is up to date.

DETAILS

ROCKSTAR

2002-03-18

PHONE NUMBER

GENDER

EDIT

LOGIN DETAILS

FIG 1.6: SELECT MY PROFILE TO VIEW WISHLIST AND LOGOUT

CONCLUSION

This ADIDAS automation project leverages the power of Python and Selenium to simplify and streamline interactions with the ADIDAS website. By automating repetitive tasks such as browsing, searching, and purchasing items, the project aims to save time and reduce manual effort. The use of robust error handling, logging, and security measures ensures accuracy and reliability in the automation process.

The project's modular design and well-documented code base make it maintainable and scalable, allowing for future enhancements and modifications. Through careful consideration of user experience, performance monitoring, and compliance with ADIDAS's terms of service,this project provides a practical and efficient solution for automating ADIDAS tasks.

In conclusion, Adidas is a global sportswear brand with a comprehensive range of products catering to both athletic and casual consumers. The company's commitment to innovation, marketing, and sustainability has positioned it as a key player in the competitive sportswear industry.

Overall, this ADIDAS automation project demonstrates the potential of automation in improving productivity and efficiency, offering valuable insights and tools for users looking to optimize their e-commerce activities.

REFERENCES

- Code creation:
  [https://youtu.be/CwLrdjgsJjU?si=M9LttQ8WJtJ9il_R]

- Documentation:
  Official Documentation:[https://docs.python.org/] and based on the creation of website.