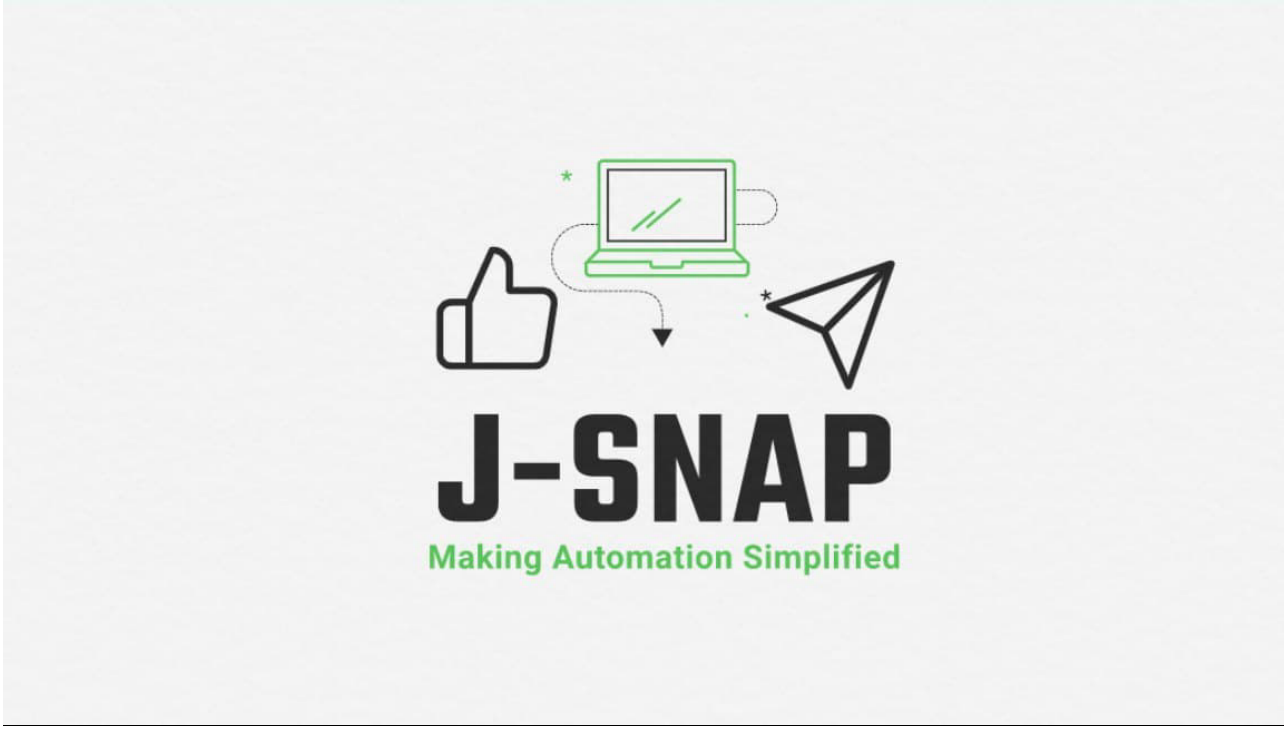
J-SNAP
Making Automation Simplified

Project By :-BATCH(B2-B3)

- Soham Kulkarni
- F.E- B-235
- Atharva Nikam
- F.E-B-263
- Aman Mansuri
- F.E- B-256
- Mayur Patil
- F.E- B-262

TABLE OF CONTENTS :-

What Is Automation?

Automation is the creation and application of technologies to produce and deliver goods and services with minimal human intervention. The implementation of automation technologies, techniques and processes improve the efficiency, reliability, and/or speed of many tasks that were previously performed by humans.

Automation is being used in a number of areas such as manufacturing, transport, utilities, defence, facilities, operations and lately, information technology.

Why Use Automation?

Usually, automation is employed to minimize labour or to substitute humans in the most menial or repetitive tasks. Automation is present in virtually all verticals and niches, although it’s more prevalent in manufacturing, utilities, transportation, and security.

For example, most manufacturing plants make use of some automated process in the form of robotic assembly lines. Human input is required only to define the processes and supervise them, while the assembling of the various components is left to the machines, which automatically convert raw materials into finished goods.

In the technology domain, the impact of automation is increasing rapidly, both in the software/hardware and machine layer. The implementation of new artificial intelligence (AI) and machine learning (ML) technologies is currently skyrocketing the evolution of this field.

Types of Automation:

Basic automation:
Basic automation takes simple, rudimentary tasks and automates them. This level of automation is about digitizing work by using tools to streamline and centralize routine tasks, such as using a shared messaging system instead of having information in disconnected silos. Business process management (BPM) and robotic process automation (RPA) are types of basic automation.

Process automation:
Process automation manages business processes for uniformity and transparency. It is typically handled by dedicated software and business apps. Using process automation can increase productivity and efficiency within your business. It can also deliver new insights

into business challenges and suggest solutions. Process mining and workflow automation are types of process automation.

Integration automation :

Integration automation is where machines can mimic human tasks and repeat the actions once humans define the machine rules. One example is the "digital worker." In recent years, people have defined digital workers as software robots that are trained to work with humans to perform specific tasks. They have a specific set of skills, and they can be "hired" to work on teams.

Artificial intelligence (AI) automation:

The most complex level of automation is artificial intelligence (AI) automation. The addition of AI means that machines can "learn" and make decisions based on past situations they have encountered and analyzed. For example, in customer service, virtual assistants powered can reduce costs while empowering both customers and human agents, creating an optimal customer service

experience.

Examples Of Automation:

In the information technology domain, a software script can test a software product and produce a report. There are also various software tools available in the market which can generate code for an application. The users only need to configure the tool and define the process.

Home automation – uses a combination of hardware and software technologies that enable control and management over appliances and devices within a home.

Network automation – the process of automating the configuration, management and operations of a computer network.

Office automation – involves using computers and software to digitize, store, process and communicate most routine tasks and processes in a standard office.

Automated website testing – streamlines and

standardizes website testing parameters for configuration changes that occur during the development phase.

Data center automation – enables the bulk of the data center operations to be performed by software programs. Includes automated system operations, also known as lights-out operations.

Test automation – software code goes through quality assurance (QA) testing automatically by scripts and other automation tools.

Automation and Job Loss:

Automation makes sure the techniques are used effectively in the delivery of products and services. However, it inherently causes many workers to become unnecessary (especially unskilled ones) and end up being displaced.

Automation will certainly have substantial negative effects on employment and wages for all those occupations that do not require particular training or skills. However, many of these employees could be easily retrained in new jobs, and the impact of this technology on our society is revolutionary enough to create new opportunities for everyone.

According to the World Bank's World Development Report 2019, the positive economic effects in terms of new industries and jobs available far outweigh the negative ones, but automation-based technological unemployment still is a cause for concern.

Despite advances in automation, some manual intervention is always advised, even if the tool can perform most of the tasks. Automation professionals involved in the creation, application, and monitoring of such technologies are in high demand.

What Is Social Media Automation?:

Social media automation is the process of optimizing social interactions using automated tools. This can include scheduling social posts ahead of time or republishing popular articles. Automating social media publication, engagement and management reduces the hours spent on maintaining and growing brand accounts. As a result, time and resources could be allocated toward other areas of the marketing budget and meeting strategic goals.

How to Use Tools for Social Media Automation:

1] Publish Posts During Peak Audience Times *:*
Optimize your reach by posting when your audience is most active on social media. For more details, check out our latest research on the best days and times to post based on engagement from different networks and industries.

2] Keep a Steady Queue of Posts *:*
Brands without a dedicated social media team may struggle with posting on a regular basis if marketers find themselves juggling multiple responsibilities. Social media automation lets you upload your content calendar so you can concentrate on other tasks. Additionally, if you went on vacation or want to publish posts outside of business hours, upload the posts and schedule publish dates days, weeks or months ahead of time.

3] Analyze Social Data :

Automated tools analyze the data in real-time and report key metrics like engagement levels, impressions or reach. Some tools will also have customized reports sent to your inbox on your chosen schedule.

4] Set Up Automatic Responses to Customers :

Improve customer care by using automated software that suggests replies to messages or chatbots that send responses to customer questions or comments.

Pros of Social Media Automation:

- Spend less time manually updating brand pages
- Maximize reach and impressions
- Stay active on social media beyond regular business hours
- Analyze social data in real-time

Examples of Social Media Automation in Action:

- Use chatbots to respond to user messages
- Schedule Facebook posts weeks ahead of campaigns
- Send tweets when your audience levels peak
- Get notifications of social conversations

WHAT'S J-SNAP?

J-Snap or JARVIS SOCIAL NETWORKING AUTOMATION PROGRAM

Is a Python language based social networking automation bot which can automate multiple social media networking platforms and carry out many of the day to day functions like Follow,like,comment etc. It utilizes Microsoft speech API and python Selenium library to work out all the important tasks in very simplified manner

TECHNOLOGY USED IN J-SNAP:

J-SNAP uses the Python code interpreter as well as Microsoft voice API in an integrated manner along with the "SELENIUM" library of python to do things in automated manner.

Python Language Info:

Python is a high-level, interpreted, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation.

Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms,including structured (particularly pr ocedural), object-oriented and functional programming. It is often described as a "batteries included" language due to its comprehensive standard library.

Library Python -

Python's large standard library provides tools

suited to many tasks, and is commonly cited as one of its greatest strengths. For Internet-facing applications, many standard formats and protocols such as MIME and HTTP are supported. It includes modules for creating graphical user interfaces, connecting to relational databases, generating pseudorandom numbers, arithmetic with arbitrary-precision decimals,[120] manipulating regular expressions, and unit testing.

Some parts of the standard library are covered by specifications—for example, the Web Server Gateway Interface (WSGI) implementation `wsgiref` follows PEP 333 but most are specified by their code, internal documentation, and test suites. However, because most of the standard library is cross-platform Python code, only a few modules need altering or rewriting for variant implementations.

As of January 2022, the Python Package Index (PyPI), the official repository for third-party Python software, contains over 350,000 packages with a wide range of

functionality

API Documentation Generators-

Tools that can generate documentation for Python API include pydoc (available as part of the standard library), Sphinx, Pdoc and its forks, Doxygen and Graphviz, among others

For JARVIS, MS API:

The **Speech Application Programming Interface** or **SAPI** is an API developed by Microsoft to allow the use of speech recognition and speech_synthesis within Wind owsapplications. To date, a number of versions of the API have been released, which have shipped either as part of a Speech SDK or as part of the Windows OS itself. Applications that use SAPI include Microsoft Office, Microsoft Agent and Microsoft Speech Server.

In general, all versions of the API have been designed such that a software developer can write an application to perform speech recognition and synthesis by using a standard set of interfaces, accessible from a variety of programming languages. In addition, it is possible for a 3rd-party company to produce their own Speech Recognition and Text-To-Speech engines or adapt existing engines to work with SAPI. In principle, as long as these engines conform to the defined interfaces they can be used instead of the Microsoft-

supplied engines.

In general, the Speech API is a freely redistributable component which can be shipped with any Windows application that wishes to use speech technology. Many versions (although not all) of the speech recognition and synthesis engines are also freely redistributable.

There have been two main ‘families’ of the Microsoft Speech API. SAPI versions 1 through 4 are all similar to each other, with extra features in each newer version. SAPI 5, however, was a completely new interface, released in 2000. Since then several sub-versions of this API have been released.

Basic Architecture-

The Speech API can be viewed as an interface or piece of middleware which sits between *applications* and speech *engines* (recognition and synthesis). In SAPI versions 1 to 4, applications could directly communicate with engines. The API included an abstract *interface definition* which

applications and engines conformed to. Applications could also use simplified higher-level objects rather than directly call methods on the engines.

In SAPI 5 however, applications and engines do not directly communicate with each other. Instead, each talks to a runtime component (**sapi.dll**). There is an API implemented by this component which applications use, and another set of interfaces for engines.

Typically in SAPI 5 applications issue calls through the API (for example to load a recognition grammar; start recognition; or provide text to be synthesized). The sapi.dll runtime component interprets these commands and processes them, where necessary calling on the engine through the engine interfaces (for example, the loading of grammar from a file is done in the runtime, but then the grammar data is passed to the recognition engine to actually use in recognition). The recognition and synthesis engines also generate events while processing (for example, to indicate an utterance has been recognized or to indicate

word boundaries in the synthesized speech). These pass in the reverse direction, from the engines, through the runtime DLL, and on to an *event sink* in the application.

In addition to the actual API definition and runtime DLL, other components are shipped with all versions of SAPI to make a complete Speech Software Development Kit. The following components are among those included in most versions of the Speech SDK:

- *API definition files* – in MIDL and as C or C++ header files.
- *Runtime components* – e.g. sapi.dll.
- *Control Panel applet* – to select and configure default speech recognizer and synthesizer.
- *Text-To-Speech engines* in multiple languages.
- *Speech Recognition engines* in multiple languages.
- *Redistributable components* to allow developers to package the engines and runtime with their application code to

produce a single installable application.

- *Sample application code*.
- *Sample engines* – implementations of the necessary engine interfaces but with no true speech processing which could be used as a sample for those porting an engine to SAPI.
- *Documentation*.

SELENIUM PYTHON LIBRARY:

Selenium is an umbrella project for a range of tools and libraries that enable and support the automation of web browsers.

It provides extensions to emulate user interaction with browsers, a distribution server for scaling browser allocation, and the infrastructure for implementations of the W3C WebDriver specification that lets you write interchangeable code for all major web browsers.

This project is made possible by volunteer contributors who have put in thousands of hours of their own time, and made the source code freely available for anyone to use, enjoy, and improve.

Selenium brings together browser vendors, engineers, and enthusiasts to further an open discussion around automation of the web platform. The project organises an annual

conference to teach and nurture the community.

At the core of Selenium is WebDriver, an interface to write instruction sets that can be run interchangeably in many browsers. Once you've installed everything, only a few lines of code get you inside a browser. You can find a more comprehensive example in Writing your first Selenium script.

J-SNAP COAD:

The most important part of our project is code.

```python
import pyttsx3
import speech_recognition as sr
import datetime
from selenium import webdriver
import time

from random import randint
from selenium.webdriver.common.by import By
from selenium.webdriver.chrome.service import Service
from webdriver_manager.chrome import ChromeDriverManager

engine = pyttsx3.init('sapi5')
voices = engine.getProperty('voices')
# print(voices[1].id)
engine.setProperty('voice', voices[0].id)


def speak(audio):
    engine.say(audio)
    engine.runAndWait()
```

```python


def wishMe():
    hour = int(datetime.datetime.now().hour)
    if 0 <= hour < 12:
        speak("Good Morning!")

    elif 12 <= hour < 18:
        speak("Good Afternoon!")

    else:
        speak("Good Evening!")

    speak("I am Jarvis Sir. Please tell me how may I help you")


def takeCommand():
    # It takes microphone input from the user and returns string output

    r = sr.Recognizer()
    with sr.Microphone() as source:
```

```python
def takeCommand():
    # It takes microphone input from the user and returns string output

    r = sr.Recognizer()
    with sr.Microphone() as source:
        print("Listening...")
        r.pause_threshold = 1
        audio = r.listen(source)

    try:
        print("Recognizing...")
        query = r.recognize_google(audio, language='en-in')
        print(f"User said: {query}\n")

    except Exception as e:
        # print(e)
        print("Say that again please...")
        return "None"
    return query
```

```python
def getRandomTime():
    randTime = randint(3, 5)
    return randTime


class InstaBot:
    def __init__(self, username, password):
        followFarm = ["ducks.ig"]
        comment_pallet = ["how cute", "Cutie pie", "super cute", "so adorable"]

        driver = webdriver.Chrome(service=Service(ChromeDriverManager().install()))
        driver.get("https://instagram.com")
        time.sleep(3)
        driver.find_element(By.XPATH, "//input[@name=\"username\"]") \
            .send_keys(username)
        time.sleep(4)
        driver.find_element(By.XPATH, "//input[@name=\"password\"]") \
            .send_keys(password)
        time.sleep(3)
        driver.find_element(By.XPATH, '//button[@type="submit"]') \
```

```
        time.sleep(3)
        driver.find_element(By.XPATH, '//button[@type="submit"]') \
            .click()
        # first not now

        time.sleep(5)
        driver.find_element(By.XPATH, "/html/body/div[1]/section/main/div/div/div/div/button") \
            .click()
        # second not now

        time.sleep(5)
        driver.find_element(By.XPATH,
                            "/html/body/div[1]/div/div[1]/div/div[2]/div/div/div[1]/div/div[
                             2]/div/div/div/div/div/div/div/div[3]/button[2]") \
            .click()

```

```
        # search

        time.sleep(10)
        driver.find_element(By.XPATH,
                            "/html/body/div[1]/div/div[1]/div/div[1]/div/div/div[1]/div[1]/section/nav/div[
                             2]/div/div/div[2]/input") \
            .send_keys(followFarm[0])

        # click on selection
        time.sleep(7)
        driver.find_element(By.XPATH,
                            "/html/body/div[1]/div/div[1]/div/div[1]/div/div/div[1]/div[1]/section/nav/div[
                             2]/div/div/div[2]/div[3]/div/div[2]/div/div[1]/a/div/div[2]/div[1]/div/div/div[
                             1]") \
            .click()

```

```
        # click on follow
        time.sleep(10)
        driver.find_element(By.XPATH,
                            "/html/body/div[1]/div/div[1]/div/div[1]/div/div/div[1]/div[
                             1]/section/main/div/header/section/div[1]/div[1]/div/div[2]/div/span/span[
                             1]/button") \
            .click()

        # click on first post
        time.sleep(7)
        driver.find_element(By.XPATH,
                            "/html/body/div[1]/div/div[1]/div/div[1]/div/div/div[1]/div[
                             1]/section/main/div/div[4]/article/div[1]/div/div[1]/div[1]") \
            .click()
```

```
    # like first post
    time.sleep(7)
    driver.find_element(By.XPATH,
                        "/html/body/div[1]/div/div[1]/div/div[2]/div/div/div[1]/div/div[3]/div/div/div/div/div[
                         2]/div/article/div/div[2]/div/div/div[2]/section[1]/span[1]/button") \
        .click()
```

```
        # comment first post
        time.sleep(3)
        driver.find_element(By.XPATH,
                            "/html/body/div[1]/div/div[1]/div/div[2]/div/div/div[1]/div/div[3]/div/div/div/div/div[
                             2]/div/article/div/div[2]/div/div/div[2]/section[1]/span[2]/button") \
            .click()

        time.sleep(5)
        driver.find_element(By.XPATH,
                            "/html/body/div[1]/div/div[1]/div/div[2]/div/div/div[1]/div/div[3]/div/div/div/div/div[
                             2]/div/article/div/div[2]/div/div/div[2]/section[3]/div/form/textarea") \
            .send_keys(comment_pallet[randint(0, (len(comment_pallet) - 1))])

        time.sleep(5)
        driver.find_element(By.XPATH,
                            "/html/body/div[1]/div/div[1]/div/div[2]/div/div/div[1]/div/div[3]/div/div/div/div/div[
                             2]/div/article/div/div[2]/div/div/div[2]/section[3]/div/form/button") \
            .click()
```

```
        time.sleep(7)
        driver.find_element(By.XPATH,
                            "/html/body/div[1]/div/div[1]/div/div[2]/div/div/div[1]/div/div[3]/div/div/div/div/div[
                             1]/div/div/div/button") \
            .click()

        time.sleep(7)
        driver.find_element(By.XPATH,
                            "/html/body/div[1]/div/div[1]/div/div[2]/div/div/div[1]/div/div[3]/div/div/div/div/div[
                             2]/div/article/div/div[2]/div/div/div[2]/section[1]/span[1]/button") \
            .click()

        time.sleep(7)
        driver.find_element(By.XPATH, "/html/body/div[1]/div/div[1]/div/div[2]/div/div/div[1]/div/div[2]/div/div") \
            .click()
        # log out

        time.sleep(10)
        driver.find_element(By.XPATH,
                            "/html/body/div[1]/div/div[1]/div/div[1]/div/div/div[1]/div[1]/section/nav/div[2]/div/div/div[
                             3]/div/div[6]/div[1]/span") \
            .click()
```

```
    time.sleep(3)
    driver.find_element(By.XPATH,
                        "/html/body/div[1]/div/div[1]/div/div[1]/div/div/div[1]/div[1]/section/nav/div[2]/div/div/div[
                         3]/div/div[6]/div[2]/div[2]/div[2]/div[2]/div/div/div/div")

    driver.quit()


if __name__ == "__main__":
    wishMe()
    while True:
        query = takeCommand().lower()

        if 'instagram' in query:
            speak('Automating instagram')
            InstaBot("JARVIS18264", "soham@123")
```

J-SNAP APPLICATION:

- J-SNAP is applicable for all social media apps in following way:

1. J-SNAP Log in to your social media accounts.
2. J-SNAP open the profile of verified user account.
3. J-SNAP follow the verified user.
4. J-SNAP open the posts of the verified user account and like the posts.
5. J-SNAP comments on the selected posts which can be customized.
6. J-SNAP log-out of your account.

•J-SNAP’s ADVANTAGES:

1. It saves our time which can be used for other essential tasks.

2. For social media influencers and celebrities with this automation, it becomes more accessible to interact with followers and maintain their social media presence effectively.

3.For small-time start-up and even normal people who cannot spare time to manage their social media this tool comes in handy.

4. For Users who cannot afford a social media manager it can be a possible alternative to make their social media presence more productive in cheap price.

• J-SNAP’s DISADVANTAGES:

1. J-SNAP doesn't do 100% perfect automation so, there is the possibility of error in working or sending massages

2. Your personal interaction with the community is reduced

3. Being a Bot if used too much frequently social media sites can Block the user account until it is added as an official extension

•REFERENCE:

- python language info - https://en.wikipedia.org/wiki/Python_(programming_language)
- Microsoft speech api info - https://en.wikipedia.org/wiki/Microsoft_Speech_API#
- selenium pyhton library info - https://www.selenium.dev/documentation/
- Topic info-

  https://www.ibm.com/topics/automation