Summer Internship Report

Submitted by Name: Tanishqa Garg
Roll No: 21CSU434

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING, SCHOOL OF ENGINEERING AND TECHNOLOGY

THE NORTHCAP UNIVERSITY GURUGRAM-122017

Internship Period: 11 July 2023 to 1 September 2023

Copy of Certificate of completion (issued by the industry you have done an internship)

Redcliffe labs

Date: 7/9/2023

Ref. No.: RCL6782/IC /2023-24

To Whom It May Concern

This is to certify that **Ms. Tanishqa Garg** was employed with us as **Intern** From **11/7/2023** To **1/9/2023**.

She is methodical & observant in Her approach to the subject and can handle work pressure.

Her ability to analyze the given task is satisfactory. She can withstand the stress & pressure

of busy work schedule, never remonstrates and is thoroughly dependable.

We wish "All the Best" in your future endeavors.

Sincerely,

Redcliffe Lifetech Pvt. Ltd.
Authorized Signatory

Kuldeep Singh Rawat

TABLE OF CONTENTS

LEARNING OBJECTIVES/INTERNSHIP OBJECTIVES

- Internships are generally thought of to be reserved for college students looking to gain¬ experience in a particular field. However, a wide array of people can benefit from Training Internships in order to receive real world experience and develop their skills.
- An objective for this position should emphasize the skills you already possess in the area¬ and your interest in learning more
- Internships are utilized in a number of different career fields, including architecture,¬ engineering, healthcare, economics, advertising and many more.
- Some internship is used to allow individuals to perform scientific research while others¬ are specifically designed to allow people to gain first-hand experience working.
- Utilizing internships is a great way to build your resume and develop skills that can be¬ emphasized in your resume for future jobs. When you are applying for a Training Internship, make sure to highlight any special skills or talents that can make you stand apart from the rest of the applicants so that you have an improved chance of landing the position

ACKNOWLEDGEMENT

I am deeply thankful for my internship at Redcliffe Labs , which was made possible by the guidance of exceptional individuals. I extend my gratitude to Mr Sachin Sharma my inspiring project manager, whose expertise and support greatly enhanced my skills and industry understanding. I also appreciate Ms Megha Sharma , the HR professional who provided me with this opportunity and ensured a seamless onboarding process.

Thanks to the entire Redcliffe Labs team for fostering a collaborative and innovative environment that facilitated my personal and professional growth. I am also grateful to my fellow interns and colleagues for their camaraderie and shared experiences.

My time at Redcliffe Labs Tech has been transformative, and I eagerly look forward to applying the knowledge and skills gained in my future endeavors.

Best Regards,
Tanishqa Garg

ABSTRACT

Redcliffe Labs is India’s most trusted and fastest-growing diagnostics services and precision medicine company focusing on preventive health checkup, routine, specialized, reproductive health investigations, oncology and rare disease diagnostics. It is also fastest growing technology empowered diagnostics service provider having its home sample collection service in more than 220+ cities with 80+ Labs and 2000+ Walk-in Wellness and Collection Centres across India.

During the summer, I had the privilege of joining Redcliffe Labs as a Product Analyst Intern, contributing to the company's data-driven strategies and insights. My role centered around automating the process , web scraping and building integration between softwares . This experience allowed me to apply my academic knowledge in a real-world context. I learned new software like postman, Flask, Bootstrap along my journey.

WEEKLY OVERVIEW OF INTERNSHIP ACTIVITIES

	DATE	DAY	NAME OF THE TOPIC/MODULE/COMPLETED
1st WEEK	11/7/23	Tuesday	On boarding and office tour
	12/7/23	Wednesday	Introduction -Web scraping
	13/7/23	Thusday	Amazon Web Scraping
	14/7/23	Friday	Introduction to Excel automation
	15/7/23	Saturday	Excel Project

	DATE	DAY	NAME OF THE TOPIC/MODULE/COMPLETED
2nd WEEK	17/7/23	Monday	Continuing Excel Automation Project
	18/7/23	Tuesday	Adding automate color scheme in Excel
	19/7/23	Wednesday	Project Updates and submission
	20/7/23	Thursday	Introduction to Postman Software
	21/7/23	Friday	Exploring Postman with company rest API
	22/7/23	Saturday	Data Entry with Postal API

	DATE	DAY	NAME OF THE TOPIC/MODULE/COMPLETED
3rd WEEK	24/7/23	Monday	Find Wellness aggregator using justdial API
	25/7/23	Tuesday	Introduction to Django and Bootstrap
	26/7/23	Wednesday	Making Website
	27/7/23	Thusday	Continue adding features to website
	28/7/23	Friday	Automating Email fetch
	29/7/23	Saturday	Automating Email fetch

	DATE	DAY	NAME OF THE TOPIC/MODULE/COMPLETED
4th WEEK	1/8/23	Tuesday	Introduction to web driver
	2/8/23	Wednesday	Automate form filling
	3/8/23	Thusday	Automate form filling
	4/8/23	Friday	Introduction to Google Analytics
	5/8/23	Saturday	Making Funnel for getting business insights

5th WEEK	DATE	DAY	NAME OF THE TOPIC/MODULE/COMPLETED
	7/8/23	Monday	Image Processing
	8/8/23	Tuesday	Web Scraping Of AdvisoryLabs
	9/8/23	Wednesday	Web Scraping of Justdial
	10/8/23	Thusday	Teaching Mail Merge to Employees
	11/8/23	Friday	Introduction to Jira and Notion
	12/8/23	Saturday	Working on Jira and Japier Integration

6th WEEK	DATE	DAY	NAME OF THE TOPIC/MODULE/COMPLETED
	14/8/23	Monday	Self Learning Javascript
	16/8/23	Wednesday	Find GreyTHr like software
	17/8/23	Thusday	Contact Companies for Pricing
	18/8/23	Friday	Self Learning Javascript
	19/8/23	Saturday	Self Learning Javascript

7th WEEK	DATE	DAY	NAME OF THE TOPIC/MODULE/COMPLETED
	21/8/23	Monday	Setting up JIira
	22/8/23	Tuesday	Setting up Notion
	23/8/23	Wednesday	Working on Integration
	24/8/23	Thusday	Working on Integration
	25/8/23	Friday	Working on Integration
	26/8/23	Saturday	Updates

8th WEEK	DATE	DAY	NAME OF THE TOPIC/MODULE/COMPLETED
	28/7/23	Monday	Setting up Manager's Notion and Jira
	29/7/23	Tuesday	Web Scraping of Terapanth directory
	1/9/23	Friday	Exit Formalities

INTRODUCTION

During my 7 weeks tenure as Product Analyst Intern at Redcliffe Labs , I had the privilege to contribute to various engineering aspects that are integral to the company's operations. My role allowed me to immerse myself in a dynamic and innovative environment, gaining hands-on experience in several key engineering domains.

1. **Excel Automation** : During my internship, I honed my skills in Excel automation using Python. I automated data processing tasks, creating custom scripts to transform raw data into pivot-table-like reports. This experience showcased the efficiency and accuracy that Python brings to data manipulation in Excel, enhancing my technical abilities.

2. **Web Scraping**: During my internship, I had a fascinating experience diving into the world of web scraping. It was like being a digital explorer, on a mission to collect important information hidden within the complex web pages of the internet.

3. **Web Development**: During my internship, I had the exciting opportunity to immerse myself in the world of web development, harnessing the power of Python frameworks such as Flask and Django. These projects allowed me to take a hands-on approach to building and maintaining interactive, user-friendly websites that were designed to provide an exceptional user experience

4. **Jira and Notion Integration :**The integration project aimed to bridge the gap between two essential too used within our organization: Jira Software, a widely-used project management and issue tracking platform, and Notion, a versatile and collaborative workspace. The significance of this integration lay in streamlining communication, improving efficiency, and enhancing data synchronization between teams that rely on these platforms.

SYSTEM REQUIREMENTS SPECIFICATIONS

System configurations

The software requirement specification can produce at the culmination of the analysis task. The function and performance allocated to software as part of system engineering are refined by established a complete information description, a detailed functional description, a representation of system behavior, and indication of performance and design constrain, appropriate validate criteria, and other information pertinent to requirements

Software requirements:

Operating System: Windows 10

Coding Language: HTML,CSS, JavaScript,Bootstrap,Flask and Python

Hardware Requirements:

Processor : Intel core i3

Memory : 8GB RAM

Hard Disk : 1TB

SOCIAL RELEVANCE OF THE PROJECT-EXCEL AUTOMATION

The social relevance of automating Excel formatting cannot be understated. In an era where data is pivotal to informed decision-making, efficiency in data processing plays a crucial role. Automating Excel formatting not only reduces manual labor but also frees up valuable time that can be redirected towards more strategic and impactful tasks. This project contributes to increased productivity, reduces the risk of human errors, and aligns with the broader societal trend of automating repetitive tasks for enhanced work-life balance

TRAINING DESCRIPTION

During this project, I undertook the task of automating Excel formatting using Python. The initial workflow involved receiving an Excel file containing raw data, which required conversion into a more structured format resembling a pivot table. The manual process of achieving this transformation was not only time-consuming but also prone to errors.

To address this challenge, I leveraged Python along with libraries like pandas to create a custom script. This script, when applied to the raw data file, instantly generated a formatted Excel document resembling a pivot table. This Python-based automation not only drastically reduced processing time but also ensured consistent and error-free formatting

GitHub Link: https://github.com/tanishqa11/excel-automation

SOURCE CODE

```python
#enter  directory of file
df=pd.read_excel(r"C:\Users\Dell\Desktop\internship\project1\file1.xlsx")
df['Date'] = pd.to_datetime(df['Date'])
today_date = datetime.now().strftime('%dth_%B')
top_7_dates = df['Date'].nlargest(7)
day_names = top_7_dates.dt.strftime('%A')

def agent(df,sheet):
        same_date_values = df.loc[df['Date'].isin(top_7_dates), 'AGENT_BOOKING'].tolist()
        seven_days_ago = top_7_dates - timedelta(days=7)
        seven_days_values = df.loc[df['Date'].isin(seven_days_ago), 'AGENT_BOOKING'].tolist()
        fourteen_days_ago = top_7_dates - timedelta(days=14)
        fourteen_days_values = df.loc[df['Date'].isin(fourteen_days_ago), 'AGENT_BOOKING'].tolist()

        twenty_eight_days_ago = top_7_dates - timedelta(days=28)
        twenty_eight_days_values = df.loc[df['Date'].isin(twenty_eight_days_ago), 'AGENT_BOOKING'].tolist()
        fiftysix_days_ago = top_7_dates - timedelta(days=56)
        fiftysix_days_values = df.loc[df['Date'].isin(fiftysix_days_ago), 'AGENT_BOOKING'].tolist()

        same_date_values1 = df.loc[df['Date'].isin(top_7_dates), 'AGENT_CENTER'].tolist()
        seven_days_ago1 = top_7_dates - timedelta(days=7)
        seven_days_values1 = df.loc[df['Date'].isin(seven_days_ago1), 'AGENT_CENTER'].tolist()
        fourteen_days_ago1 = top_7_dates - timedelta(days=14)
        fourteen_days_values1 = df.loc[df['Date'].isin(fourteen_days_ago1), 'AGENT_CENTER'].tolist()
        twenty_eight_days_ago1 = top_7_dates - timedelta(days=28)
        twenty_eight_days_values1 = df.loc[df['Date'].isin(twenty_eight_days_ago1), 'AGENT_CENTER'].tolist()
        fiftysix_days_ago1 = top_7_dates - timedelta(days=56)
        fiftysix_days_values1 = df.loc[df['Date'].isin(fiftysix_days_ago1), 'AGENT_CENTER'].tolist()
        ratio_same=[int(x / y + 0.5)for x, y in zip(same_date_values, same_date_values1)]
        ratio_seven=[int(x / y + 0.5) for x, y in zip(seven_days_values, seven_days_values1)]
        ratio_fourteen=[int(x / y + 0.5) for x, y in zip(fourteen_days_values, fourteen_days_values1)]
        ratio_twenty_eight=[int(x / y + 0.5) for x, y in zip(twenty_eight_days_values, twenty_eight_days_values1)]
        ratio_fifty_six=[int(x / y + 0.5) for x, y in zip(fiftysix_days_values, fiftysix_days_values1)]
        global table_data
        table_data = pd.DataFrame({
            'Date': top_7_dates,
            'Days': day_names,
            'Same Date': same_date_values,
            "-14 Days":fourteen_days_values,
            '-7 Days': seven_days_values,
            "-28 Days":twenty_eight_days_values,
            "-56 Days ":fiftysix_days_values,
            'Same date':same_date_values1,
            '-7 days': seven_days_values1,
            "-14 days":fourteen_days_values1,
            "-28 days":twenty_eight_days_values1,
            "-56 days ":fiftysix_days_values1,
            "same date":ratio_same,
            "-7 day":ratio_seven,
            "14 day":ratio_fourteen,
            "-28 day":ratio_twenty_eight,
            "-56 day":ratio_fifty_six
        })
        total_row = table_data.sum(numeric_only=True)
        total_row['Date'] = ' WEEK TOTAL'
        total_row['Days'] = ''


        # Append the total row to the DataFrame
        table_data = table_data._append(total_row, ignore_index=True)
        header = pd.MultiIndex.from_tuples([('AGENT ', 'Date'),
                                            ('', 'Days'),
                                            ("PICKUP COUNT ",'Same Date'),
                                            ( '','-7 Days'),( '','-14 Days'),
                                            ("",'-28 Days'),
                                            ("",'-56 Days'),
                                            ( 'PARTNER COUNT ','Same date'),
                                            ( "",'-7 days'), ( '','-14 Days'),

                                            ( "",'-28 days'),("","-56 days"),
                                            ("RATIO ","same date"),
                                            ( "",'-7 day'), ( '','-14 Days'),

                                            ( "",'-28 day'),("","-56 day")
                                            ])

        # Set the multi-index header to the existing table
        table_data.columns = header
        with pd.ExcelWriter(fr'C:\Users\Dell\Desktop\internship\project1\{today_date}.xlsx', engine='openpyxl') as writer:
            # Save table1 to Sheet1
            table_data.to_excel(writer, sheet_name=sheet)
        wb=openpyxl.load_workbook(fr"C:\Users\Dell\Desktop\internship\project1\{today_date}.xlsx")
        ws=wb["AGENT"]
        rows = ws.iter_rows(min_row=1, max_row=ws.max_row, min_col=1, max_col=ws.max_column)
        for row in rows:
            for cell in row:
                cell.border = Border(left=Side(style='thin'), right=Side(style='thin'),
                                     top=Side(style='thin'), bottom=Side(style='thin'))

        fill_pattern=PatternFill(patternType="solid", fgColor="6495ED")
        for col_idx in range(2, len(header) + 2):
            column_letter = get_column_letter(col_idx)
            ws[column_letter+"2"].fill=fill_pattern
        ws['B1'].fill=PatternFill(patternType="solid",fgColor="FFF000")
        for i in range(4,15,5):
            column_letter1 = get_column_letter(i)
            ws[column_letter1+"1"].fill=PatternFill(patternType="solid", fgColor="6495ED")
        for idx in range(4,11):
            for j in range(4, len(header) + 1):
                column=get_column_letter(j)+str(idx)
                next_column = get_column_letter(j + 1) + str(idx)

                if ws[column].value>ws[next_column].value:
                    ws[column].fill=PatternFill(patternType="solid",fgColor="90EE90")
                elif ws[column].value==ws[next_column].value:
                    ws[column].fill=PatternFill(patternType="solid",fgColor="FFFFFF")
                else:
                    ws[column].fill=PatternFill(patternType="solid",fgColor="FF0000")

        wb.save(fr"C:\Users\Dell\Desktop\internship\project1\{today_date}.xlsx")
        return table_data

def api(df,sheet):...

def frame(df,sheet):...

# Display the table with the header
print(agent(df,"AGENT"))
print(api(df,"API"))
print(frame(df,"FRAME"))
def percentage1():
    Heading=["Date","Day","Same Date","-7 Days","-14 Days","-28 Days","-56 Days","Same date","-7 days","-14 Days","-28 days","-56 days","Same date","-7 days","-14 Days","-28 days","-56 days"]
    new_table= pd.DataFrame(columns=Heading)
    row1=table_data.iloc[-1].values
    new_table = new_table._append(pd.Series(row1, index=Heading), ignore_index=True)
    row2=table_data1.iloc[-1].values
    new_table = new_table._append(pd.Series(row2, index=Heading), ignore_index=True)
    row3=table_data2.iloc[-1].values
    new_table = new_table._append(pd.Series(row3, index=Heading), ignore_index=True)
    totals= new_table.drop(columns=["Date",'Day']).sum()

    # Calculate the percentage for each row (excluding the "Grand total" row)
    percentage_values = new_table.drop(columns=["Date", "Day"])
    percentage_values = percentage_values.div(totals) * 100
    percentage_values = percentage_values.applymap(lambda value: f"{int(value)}%")
    percentage_values.insert(loc=0,column="Name",value=["AGENT","API","IFRAME"])
    with pd.ExcelWriter(fr'C:\Users\Dell\Desktop\internship\project1\{today_date}.xlsx', mode='a', engine='openpyxl') as writer:
        percentage_values.to_excel(writer, sheet_name="percentage")
percentage1()
print("done ,please check your file ")
```

PROJECT PREVIEW

AGENT		PICKUP COUNT					PARTNER COUNT					RATIO				
Date	Days	Same Date	-7 Days	-14 Days	-28 Days	-56 Days	Same date	-7 days	-14 days	-28 days	-56 days	same date	-7 day	-14 Days	-28 day	-56 day
7/19/2023	Wednesday	365	398	375	401	336	48	50	39	42	40	8	8	10	10	8
7/20/2023	Thursday	276	354	299	398	299	47	46	50	46	42	6	7	7	9	7
7/21/2023	Friday	315	370	384	499	334	46	50	52	51	51	7	8	7	10	7
7/22/2023	Saturday	475	603	601	675	467	60	52	45	49	46	8	12	13	14	10
7/23/2023	Sunday	421	410	429	443	375	58	43	41	50	46	7	10	10	9	8
7/24/2023	Monday	286	328	313	461	403	44	48	33	38	41	7	7	10	12	10
7/25/2023	Tuesday	466	365	336	480	339	60	51	40	46	37	8	7	9	10	9
WEEK TOTAL		2604	2828	2737	3357	2553	363	340	300	322	303	51	59	66	74	59

API		PICKUP COUNT					PARTNER COUNT					RATIO				
Date	Days	Same Date	-7 Days	-14 Days	-28 Days	-56 Days	Same date	-7 days	-14 days	-28 days	-56 days	same date	-7 day	-14 day	-28 day	-56 day
7/19/2023	Wednesday	167	147	122	108	141	11	11	11	7	8	15	13	11	15	18
7/20/2023	Thursday	141	124	104	135	123	7	8	9	9	11	20	16	12	15	11
7/21/2023	Friday	141	116	117	116	87	8	7	10	8	10	18	17	12	15	9
7/22/2023	Saturday	215	193	162	156	127	9	11	10	9	9	24	18	16	17	14
7/23/2023	Sunday	225	178	150	176	133	10	10	11	10	8	23	18	14	18	17
7/24/2023	Monday	107	104	130	90	81	8	9	9	10	6	13	12	14	9	14
7/25/2023	Tuesday	101	123	151	103	99	10	8	9	9	9	10	15	17	11	11
WEEK TOTAL		1097	985	936	884	791	63	64	69	62	61	123	109	96	100	94

IFRAME		PICKUP COUNT					PARTNER COUNT					RATIO				
Date	Days	Same Date	-7 Days	-14 Days	-28 Days	-56 Days	Same date	-7 days	-14 days	-28 days	-56 days	same date	-7 day	-14 day	-28 day	-56 day
7/19/2023	Wednesday	12	11	9	20	25	3	5	5	5	8	4	2	2	4	3
7/20/2023	Thursday	4	13	9	15	18	3	5	3	5	5	1	3	3	3	4
7/21/2023	Friday	9	10	12	19	19	3	5	5	6	5	3	2	2	3	4
7/22/2023	Saturday	6	16	9	40	21	4	5	5	6	5	2	3	2	7	4
7/23/2023	Sunday	8	11	10	23	33	5	3	6	7	6	2	4	2	3	6
7/24/2023	Monday	1	9	2	11	9	1	3	2	3	4	1	3	1	4	2
7/25/2023	Tuesday	4	8	6	25	21	3	4	3	6	5	1	2	2	4	4
WEEK TOTAL		44	78	57	153	146	22	30	29	38	38	14	19	14	28	27

Name	Same Date	-7 Days	-14 Days	-28 Days	-56 Days	Same date	-7 days	-14 days	-28 days	-56 days	Same date	-7 days	-14 Days	-28 days	-56 days
AGENT	69%	72%	73%	76%	73%	81%	78%	75%	76%	75%	27%	31%	37%	36%	32%
API	29%	25%	25%	20%	22%	14%	14%	17%	14%	15%	65%	58%	54%	49%	52%
IFRAME	1%	2%	1%	3%	4%	4%	6%	7%	9%	9%	7%	10%	7%	13%	15%

API	Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7	Week 8	Grand Total
Alyve_Health	18	7	16	10	9	1	31	20	112
BEATO		2			1				3
Bajaj_Finserv	646	513	576	617	527	518	481	430	4308
BreatheWellBeing	3	9	6	8	4	7	7	3	47
CircleHealth-(Alignment)	4		1		4	27	6	8	50
Eka.Care	98	83	54	64	60	46	45	42	492
Ekincare	194	202	94	64	55	35	35	50	729
FLIPHEALTH	6	14	15	8	9	12	17	20	101
Fitterfly								1	1
HealthifyMe	13	33	20	23	27	34	33	30	213
IHO	62	59	56	34	39	41	27	46	364
KENKO	3	1			4	1	2	2	13
Kochhacare		1							1
Lybrate	9	8	58	83	122	133	158	92	663
TWIN_IFRAME			1						1
Visit_Health_PPMC	39	52	36	28	27	27	25	33	267
ZYLA	6	3	4		1				14
WEEK TOTAL	1101	987	937	939	889	882	867	777	7379

IFRAME	Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7	Week 8	Grand Total
Accenture					1				1
Bank_Retirees_Wellfare_Association	2		2	1	1			1	7
Fitterfly	13	22	25	69	75	75	78	73	430
MEDCORDS		1					1		2
Macromill_Howden						2			2
Mangohomz			1		1			1	3
Niyox	1								1
Plum	2	3	4	7	3	1	2	2	24
Plum Advance	2	3	4	10	2	1	1	3	26
Plum Comprehensive		5	2	44	42	31	17	21	162
Plum Essential	10	28	6	13	7	10	10	8	92
RGI-CSB								1	1
SecureNow	1			2	4	3	3		13
TWIN_IFRAME	9	14	12	6	10	8	13	11	83
VantageCircle				5	2	1	2	2	12
WEEK TOTAL	40	76	56	157	148	132	127	123	859

SCOPE OF THE PROJECT

The scope of this project extends beyond automating Excel formatting. It opens the door to explore further automation opportunities within data processing tasks. Expanding the project's scope could involve developing additional scripts or integrating it with broader data management systems, fostering a culture of automation within the organization.

FUTURE MODIFICATIONS OF THE PROJECT

- User-Friendly Interface: Developing a user-friendly interface for non-technical users to apply the automation easily.

- Error Handling: Implementing robust error-handling mechanisms to gracefully manage unexpected data scenarios.

SOCIAL RELEVANCE OF THE PROJECT-WEB SCRAPING

The social relevance of web scraping for information retrieval lies in its capacity to efficiently extract valuable insights from online platforms. In today's data-driven world, access to timely and relevant information is critical for informed decision-making. By scraping data from websites like Amazon, Justdial, and Advisory Labs, this project contributes to enhancing competitiveness, market analysis, and business intelligence. It empowers organizations to stay up-to-date with market trends and make data-driven decisions, ultimately fostering economic growth and innovation.

TRAINING DESCRIPTION

In this project, I undertook the task of web scraping, a process of automating data extraction from various websites. The goal was to retrieve specific information from websites like Amazon, Justdial, and Advisory Labs, as per my manager's requirements.

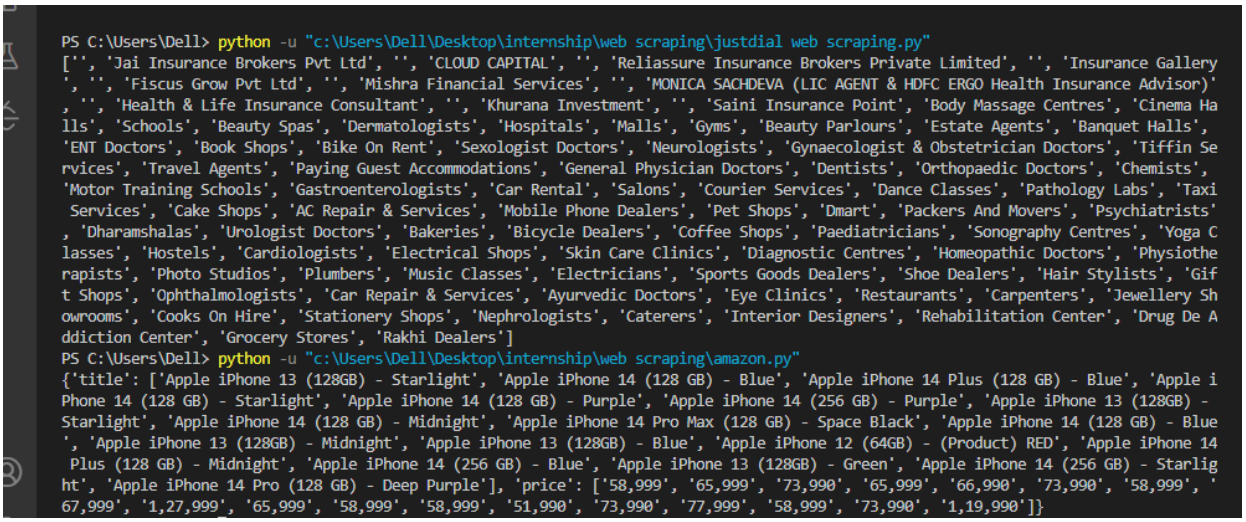
I utilized Python and web scraping libraries such as BeautifulSoup to craft custom scripts. These scripts navigated the target websites, mimicking human interactions to extract the desired data

SOURCE CODE

```
from bs4 import BeautifulSoup
import requests
import pandas as pd
data={"title":[],"price":[]}

header={"User-Agent":"Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36"}
url="https://www.amazon.in/s?k=iphone"
response=requests.get(url,headers=header)
html=response.text
soup=BeautifulSoup(html,"html.parser")
spans = soup.find_all("span", class_="a-size-medium a-color-base a-text-
normal")
prices = soup.find_all("span", class_="a-price-whole")

for span, price in zip(spans, prices):
    data["title"].append(span.get_text())
    data["price"].append(price.get_text())

print(data)
```

```
from bs4 import BeautifulSoup
import requests
import pandas as pd
import json
data={"title":[],"address":[]}

header={"User-Agent":"Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36"}
url="https://www.justdial.com/Delhi/Insurance-Agents/nct-10271175"

response=requests.get(url,headers=header)
html=response.text
soup=BeautifulSoup(html,"html.parser")
insurance_agent_links = soup.select('a[href^="/Delhi/"]')

# Loop through the links to extract names
for link in insurance_agent_links:
    name = link.get_text(strip=True)
    data["title"].append(name)

# Print the extracted names
print(data["title"])
```

PROJECT PREVIEW



```
PS C:\Users\Dell> python -u "c:\Users\Dell\Desktop\internship\web scraping\justdial web scraping.py"
['', 'Jai Insurance Brokers Pvt Ltd', '', 'CLOUD CAPITAL', '', 'Reliassure Insurance Brokers Private Limited', '', 'Insurance Gallery
', '', 'Fiscus Grow Pvt Ltd', '', 'Mishra Financial Services', '', 'MONICA SACHDEVA (LIC AGENT & HDFC ERGO Health Insurance Advisor)'
, '', 'Health & Life Insurance Consultant', '', 'Khurana Investment', '', 'Saini Insurance Point', 'Body Massage Centres', 'Cinema Ha
lls', 'Schools', 'Beauty Spas', 'Dermatologists', 'Hospitals', 'Malls', 'Gyms', 'Beauty Parlours', 'Estate Agents', 'Banquet Halls',
'ENT Doctors', 'Book Shops', 'Bike On Rent', 'Sexologist Doctors', 'Neurologists', 'Gynaecologist & Obstetrician Doctors', 'Tiffin Se
rvices', 'Travel Agents', 'Paying Guest Accommodations', 'General Physician Doctors', 'Dentists', 'Orthopaedic Doctors', 'Chemists',
'Motor Training Schools', 'Gastroenterologists', 'Car Rental', 'Salons', 'Courier Services', 'Dance Classes', 'Pathology Labs', 'Taxi
 Services', 'Cake Shops', 'AC Repair & Services', 'Mobile Phone Dealers', 'Pet Shops', 'Dmart', 'Packers And Movers', 'Psychiatrists'
, 'Dharamshalas', 'Urologist Doctors', 'Bakeries', 'Bicycle Dealers', 'Coffee Shops', 'Paediatricians', 'Sonography Centres', 'Yoga C
lasses', 'Hostels', 'Cardiologists', 'Electrical Shops', 'Skin Care Clinics', 'Diagnostic Centres', 'Homeopathic Doctors', 'Physiothe
rapists', 'Photo Studios', 'Plumbers', 'Music Classes', 'Electricians', 'Sports Goods Dealers', 'Shoe Dealers', 'Hair Stylists', 'Gif
t Shops', 'Ophthalmologists', 'Car Repair & Services', 'Ayurvedic Doctors', 'Eye Clinics', 'Restaurants', 'Carpenters', 'Jewellery Sh
owrooms', 'Cooks On Hire', 'Stationery Shops', 'Nephrologists', 'Caterers', 'Interior Designers', 'Rehabilitation Center', 'Drug De A
ddiction Center', 'Grocery Stores', 'Rakhi Dealers']
PS C:\Users\Dell> python -u "c:\Users\Dell\Desktop\internship\web scraping\amazon.py"
{'title': ['Apple iPhone 13 (128GB) - Starlight', 'Apple iPhone 14 (128 GB) - Blue', 'Apple iPhone 14 Plus (128 GB) - Blue', 'Apple i
Phone 14 (128 GB) - Starlight', 'Apple iPhone 14 (128 GB) - Purple', 'Apple iPhone 14 (256 GB) - Purple', 'Apple iPhone 13 (128GB) -
Starlight', 'Apple iPhone 14 (128 GB) - Midnight', 'Apple iPhone 14 Pro Max (128 GB) - Space Black', 'Apple iPhone 14 (128 GB) - Blue
', 'Apple iPhone 13 (128GB) - Midnight', 'Apple iPhone 13 (128GB) - Blue', 'Apple iPhone 12 (64GB) - (Product) RED', 'Apple iPhone 14
 Plus (128 GB) - Midnight', 'Apple iPhone 14 (256 GB) - Blue', 'Apple iPhone 13 (128GB) - Green', 'Apple iPhone 14 (256 GB) - Starlig
ht', 'Apple iPhone 14 Pro (128 GB) - Deep Purple'], 'price': ['58,999', '65,999', '73,990', '65,999', '66,990', '73,990', '58,999', '
67,999', '1,27,999', '65,999', '58,999', '58,999', '51,990', '73,990', '77,999', '58,999', '73,990', '1,19,990']}
```

LIMITATIONS OF THE PROJECT

It's important to acknowledge some limitations of web scraping. Firstly, web scraping is subject to the structure and layout of target websites. Changes in website design or structure may require script adjustments.

SCOPE OF THE PROJECT

The scope of this project extends beyond the specific websites targeted. It showcases the potential for web scraping to extract data from diverse online sources. The skills developed during this project can be applied to gather data from various websites, opening doors to numerous applications, from market research to competitive analysis.

SOCIAL RELEVANCE OF THE PROJECT-API

- The project addresses a practical need for mapping postal codes to geographical areas.
- It can be beneficial for businesses, logistics, and various applications that require location-based data.

TRAINING DESCRIPTION

- Training will involve learning how to make HTTP requests to an external API.
- Understanding the structure of the postal API and how to parse its responses.
- Implementing data extraction and processing techniques to map postal codes to areas.
- Learning error handling and validation for dealing with missing or invalid postal codes.
- Gaining knowledge of data storage and retrieval to efficiently handle API responses.

SOURCE CODE

```
import pandas as pd
import numpy as np
import requests

#reading the file
df= pd.read_excel(r"C:\Users\Dell\Desktop\internship\project4 (5-wellness
)\Active CC List PAN INDIA.xlsx", sheet_name="REDCLIFF CC")
df=df[["CC Name","City","ContactNo.","Pin Code"]]
df=df.dropna()
df['Pin Code'] = df['Pin Code'].astype(int)
df['Pin Code'] = df['Pin Code'].astype(str)
list1=df["Pin Code"]
df = df.assign(Area='NAN')
df=df.assign(Source="REDCLIFFE CC")

for i in range(0,len(list1)):
    PINCODE=list1[i]
    API_ENDPOINT = f'https://api.postalpincode.in/pincode/{PINCODE}'
    response = requests.get(API_ENDPOINT)
    if response.status_code == 200:
        print("ok")
        data = response.json()  # Assuming the API returns JSON data.
        if data[0]["Status"]=="Success":
             for post_office in data[0]['PostOffice']:
                        area_name = post_office['District']
                        df["Area"][i]=area_name

        else:
            print("Not found")
print(df)
print("Area names updated successfully!")
```

PROJECT PREVIEW

	CC Name	City	ContactNo.	Pin Code	Area	Source
0	Sai Pathology	Lucknow	7081377700	226010	Lucknow	REDCLIFFE CC
1	Manohar and Son`s Group	Lucknow	9839042606	226022	Lucknow	REDCLIFFE CC
2	Harihar Nagar Collection Center	Lucknow	9450686155	226016	Lucknow	REDCLIFFE CC
3	Redcliffe Collection center Aichar Greater Noida	Greater Noida	9999388500	201308	NAN	REDCLIFFE CC
4	Sindhu Diagnostic Center	Gurugram	9311493569	122009	Gurgaon	REDCLIFFE CC
5	Health Collection Center	Meerut	8476961921	250222	Meerut	REDCLIFFE CC
6	Redcliffe Collection Center (Kalyanpuri)	Delhi	9210480553	110091	East Delhi	REDCLIFFE CC
7	Radhey Collection Center	Gurugram	7906604752	122001	Gurgaon	REDCLIFFE CC
8	Redcliffe collection Center (Sec03 Faridabad)	Ballabhgarh	7290964909	121004	Faridabad	REDCLIFFE CC

```
Area names updated successfully!
```

LIMITATIONS OF THE PROJECT

- Handling a large volume of postal code requests may require managing rate limits imposed by the API provider.
- There could be issues with incomplete or missing postal code data, leading to incomplete area mapping.

. SCOPE OF THE PROJECT

- Handling edge cases, such as international postal codes, can be part of the project scope.
- Integration with other location-based services or databases could expand the scope.

SOCIAL RELEVANCE OF THE PROJECT-FORM FILLING AUTOMATION

- Building a user-friendly website with automated form filling has practical applications in various industries, such as e-commerce, online services, and data collection.
- It enhances user experience by reducing manual input and potential errors.
- The project can improve the efficiency of online processes, making them more accessible to users.

TRAINING DESCRIPTION:

- Learning web development fundamentals, including HTML, CSS, and Bootstrap for creating the website's frontend.
- Understanding the principles of responsive design to ensure the site works well on various devices.
- Exploring the Selenium WebDriver framework for browser automation.
- Implementing test scripts to interact with web elements, fill out forms, and submit data.
- Practicing error handling, test case design, and debugging for robust automation.

Github Link :https://github.com/tanishqa11/project6.github.io

SOURCE CODE

Form.html

```
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.1/dist/css/bootstrap.min.css"
rel="stylesheet">
  <link href="https://getbootstrap.com/docs/5.3/assets/css/docs.css" rel="stylesheet">
  <title>My website </title>
  <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.1/dist/js/bootstrap.bundle.min.js"></script
>
</head>
<body class="p-3 m-0 border-0 bd-example m-0 border-0">
  <nav class="navbar navbar-expand-lg bg-body-tertiary">
    <div class="container-fluid">
      <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-
target="#navbarSupportedContent"
        aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle
navigation">
        <span class="navbar-toggler-icon"></span>
      </button>
      <div class="collapse navbar-collapse" id="navbarSupportedContent">
        <ul class="navbar-nav me-auto mb-2 mb-lg-0">
          <li class="nav-item">
            <a class="nav-link active" aria-current="page"
href="C:\Users\Dell\Desktop\internship\project6\about.html">About us</a>
          </li>
          <li class="nav-item">
            <a class="nav-link active" aria-current="page"
href="C:\Users\Tanishqa\Desktop\project 6\contact.html">Contact Us</a>
          </li>
          <li class="nav-item">
            <a class="nav-link active" aria-current="page"
href="C:\Users\Dell\Desktop\internship\project6\form.html">GeneralEnquiry</a>
          </li>
          <li class="nav-item">
            <a class="nav-link active" aria-current="page"
href="C:\Users\Dell\Desktop\internship\project6\index.html">Home</a>
          </li>
          <li class="nav-item">
            <a class="nav-link active" aria-current="page"
href="C:\Users\Dell\Desktop\internship\project6\mydata.html">Data</a>
          </li>
      </div>
    </div>
  </nav>
  <form method="post" id="myForm">
    <p>First Name <input type="text" id="name" name="firstname" required maxlength="20">
last name <input type="text"
        id="lname" name="lastname" maxlength="20"></p>
    <P> Gender : <input type="radio" id="gender" name="gender">Male
      <input type="radio" id="gender" name="gender">Female</P>
    <p> Address <input type="text" id="address" name="address" required
maxlength="200"></p>
    <p>Age :<select id="age" name="Age">
        <option name="below 18">Below 18</option>
        <option name="18-15">18-25</option>
        <option name="25-40">25-40</option>
```

```
        <option name="40+">40+</option>
      </select></p>
    <p>City<input type="text" id="city" name="city" required maxlength="20"> Pincode <input
type="text" id='code'
        name="pincode" required minlength="6"></p>
    <p>Mobile <input type="tel" id="phone" name="phone" size="10" required
maxlength="10"></p>
    <p> Email <input type="email" id="mail" name="email" required
placeholder="example@gmail.com"
        title="Please enter a valid Gmail address (example@gmail.com)"></p>
    <p> password <input type="password" id="password" name="password" required
minlength="8"
        title="Password must be at least 8 characters long"></p>
    <br><br>
    <input type="submit" value="submit" name="submit">
    <span id="success"></span>
  </form>     <script>
    document.addEventListener("DOMContentLoaded", function () {
      var form = document.getElementById("myForm");
      var extractedData = JSON.parse(localStorage.getItem("myobject"));
      if (extractedData) {
        document.getElementById('name').value = extractedData.firstname || '';
        document.getElementById('lname').value = extractedData.lastname || '';
        if (extractedData.gender === "Male") {
          document.getElementById('gender').checked = true;
        } else if (extractedData.gender === "Female") {
          document.getElementById('gender').checked = true;
        }
        document.getElementById('address').value = extractedData.address || '';.. #similar code
      }
      form.addEventListener("submit", function (e) {
        e.preventDefault();
        var selectedGender = document.querySelector('input[name="gender"]:checked');
        var genderValue = selectedGender ? selectedGender.value : '';
        var formData = {
          form: document.getElementById("myForm"), #similar code
  };
        localStorage.setItem("myobject", JSON.stringify(formData))
        window.location.href = "mydata.html";
      });
    });
  </script> </body> </html>
```

Main.py(fetching data from email)

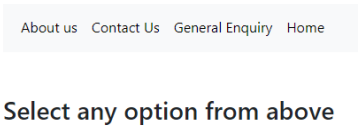
```
import imaplib
import email
import re
from bs4 import BeautifulSoup
mail = imaplib.IMAP4_SSL('imap.gmail.com'
mail.login('tanishqagargdevi@gmail.com', *)
mail.select('Inbox')
key = 'Subject'
value = 'visit'
search_criterion = f'SUBJECT "{value}"'
# Search for emails with specific subject
result, data = mail.search(None, search_criterion)
email_ids = data[0].split()  #IDs of all emails that we want to fetch
import json
extracted_data_list = []
for email_id in email_ids:
    result, msg_data = mail.fetch(email_id, '(RFC822)')
    msg = email.message_from_bytes(msg_data[0][1])
    for part in msg.walk():
        if part.get_content_type() == "text/plain":
```

```
            email_body = part.get_payload(decode=True).decode()
            name_match = re.search(r'\*Name:\*\s*(\w+)', email_body) #similarcode
            if name_match and age_match and gender_match and phone_match and test_match and
collection_match and user_match and user_phone_match and mail_match and  mail_match and
source_match and  address_match:
                name = name_match.group(1) #similarcode
                extracted_data = {
                        "name": name,"age": age,"gender": gender,"phone": phone, "test":
test_list,
                        "collection_date": collection,"user": user,"user_phone":
user_phone,"mail": user_mail,
                        "source_type": source_type,"user_address": user_address
                    }
                extracted_data_list.append(extracted_data)
            else :
                print("some error occured")
with open('extracted_data.json', 'w') as json_file:
    json.dump(extracted_data_list, json_file)
print("Data saved to extracted_data.json")
```

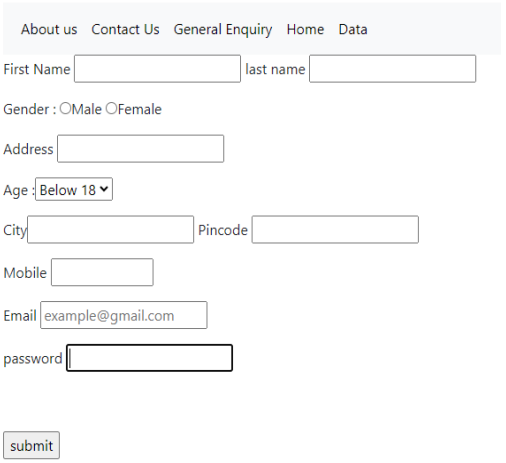
Filling.py(filling form)

```
from selenium import webdriver
from selenium.webdriver.support.ui import Select
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.by import By
import json
import time
# Load the extracted data from local storage
with open(r'extracted_data.json', 'r') as f:
    extracted_data = json.load(f)
driver = webdriver.Chrome()
driver.get(r'C:\Users\Dell\Downloads\chromedriver-win32')
driver.get('https://tanishqa11.github.io/project6.github.io/form.html')
driver.find_element("id",'name').send_keys(extracted_data[0]['name']) #similar code
if int(extracted_data[0]["age"])<18:
    age="below 18"
elif int(extracted_data[0]["age"])>18 and int(extracted_data[0]["age"])<25:
    age="18-25"
if int(extracted_data[0]["age"])>25 and int(extracted_data[0]["age"])<40:
    age="25-40"
if int(extracted_data[0]["age"])>40:
    age="40+"
age_select=driver.find_element("id","age")
select=Select(age_select)
select.select_by_value(age)
driver.find_element("name",'submit').click()
```

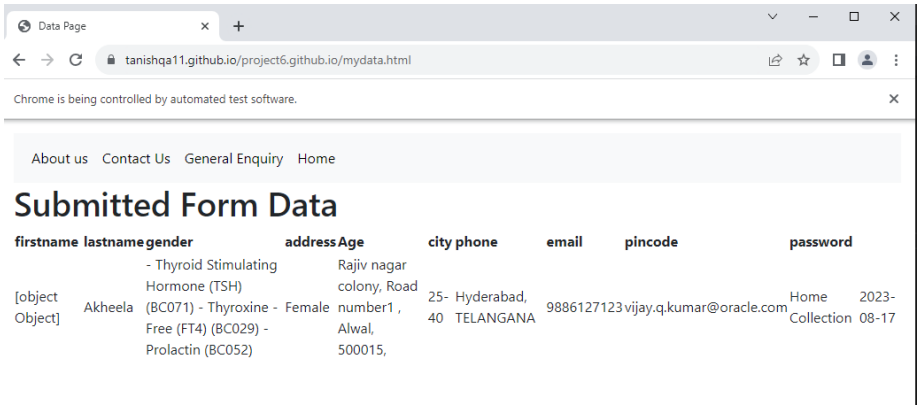
PROJECT PREVIEW



Select any option from above



Automatic form filling



LIMITATIONS OF THE PROJECT

- Automation may require regular maintenance as websites can undergo updates or changes in structure.
- The project's success depends on the stability and consistency of the website's structure and form elements.
- Ethical considerations, such as respecting website terms of service, should be taken into account to avoid potential legal issues.

FUTURE MODIFICATIONS OF THE PROJECT

- Expanding the website to include additional features beyond form filling.
- Enhancing the automation scripts to cover more complex scenarios and user interactions.
- Integrating the project into a larger software development or quality assurance process.
- Exploring options for parallel test execution and reporting for efficient testing.
- Staying updated with Bootstrap and Selenium updates to ensure compatibility and security.

SOCIAL RELEVANCE OF THE PROJECT-SOFTWARE INTEGRATION

- This project addresses the need for seamless collaboration and data synchronization among team members and stakeholders.
- It streamlines project management and documentation, saving time and reducing the risk of data inconsistencies.
- The integration enhances transparency and communication in project development and management.

TRAINING DESCRIPTION

- Understanding the Jira and Notion APIs and their capabilities for data retrieval and manipulation.
- Developing scripts or applications that can fetch data from Jira, such as issue details and due dates, and update Notion databases accordingly.
- Testing and debugging the integration to ensure reliability and data accuracy.

Github Link:**https://github.com/tanishqa11/project_8**

SOURCE CODE

App.py

```python
from flask import Flask, render_template, request
import subprocess
import json
app = Flask(__name__)
attachment_data_list = []
@app.route('/')
def index():
    return render_template('index.html')
@app.route('/run_script', methods=['POST'])
def run_script():
    subprocess.run(['python', 'jira api.py'])
    return "done"
@app.route('/notion_to_jira', methods=['POST'])
def notion_to_jira():
    subprocess.run(['python', 'notion to jira.py'])
    return "Done"
if __name__ == '__main__':
    app.run(debug=True)
```

Index.html

C: > Users > Dell > Desktop > internship > project8 > templates > index.html > html

```html
<!DOCTYPE html>
<html>
<head>
    <title>Run Script</title>
</head>
<body>
    <h1>Run Your Script</h1>
    <form method="POST" action="/run_script">
        <button type="submit">update notion </button>
    </form>
<form method="POST" action="/notion_to_jira">
    <button type="submit" name="download_attachment">update jira</button>
</form>
</body>
</html>
```

Jira api.py

```python
#authenticating jira
JIRA_API_URL = "https://tanishqa.atlassian.net/rest/api/3/search"
```

```python
#collecting data from jira project #edited
    jira_data.append({
        "TicketId":i["key"],
       "Title": i["fields"]["summary"],
```

```python
# Notion API and authentication details
NOTION_AUTH_TOKEN="secret_KkECYe0VpFyAyGZhrg6IG2NG6CSCmtDYjxmL0mVoEtI"
```

```python
for result in search_data.get("results", []):
   #collecting notion data
```

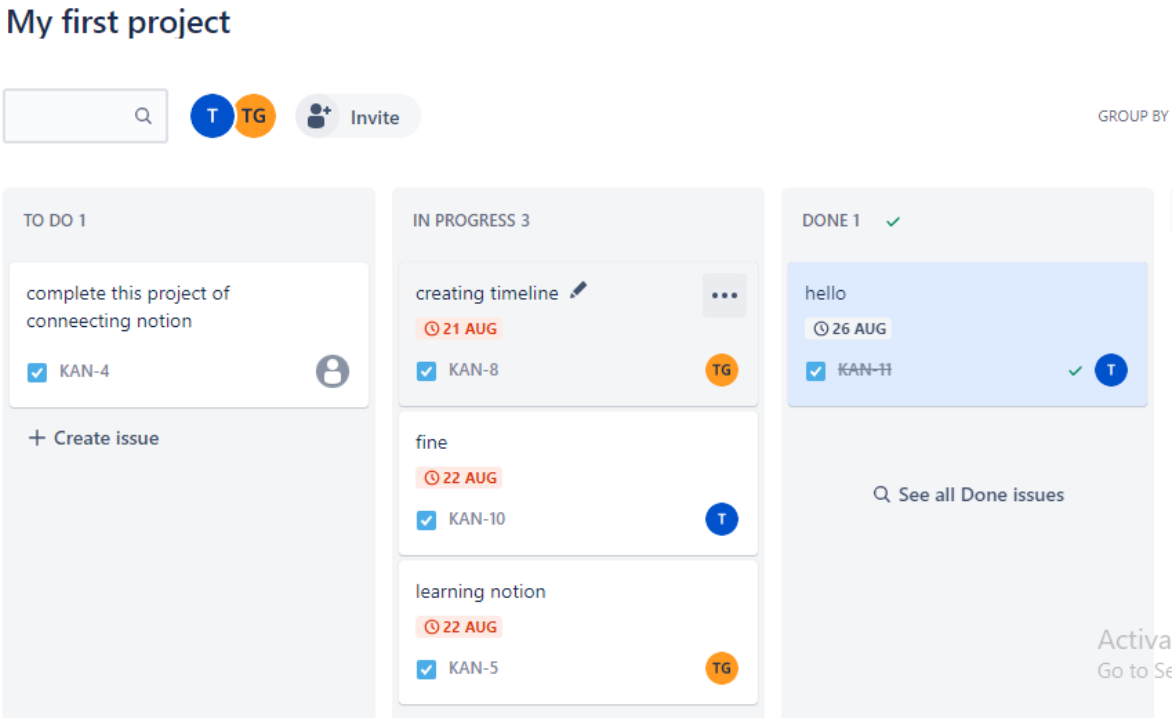
```
    title_property=result.get("properties", {}).get("Title ", {}).get("title", [])
updating notion
def create_notion_payload(JIRA, NOTION):
    notion_payload = {
        "properties": {}}
    if JIRA["status"] != NOTION["status"]:
        notion_payload["properties"]["status"] = {
            "multi_select": [{"name": JIRA["status"]}]}
```

```
# Update value in Notion
            notion_payload = create_notion_payload(data_entry, existing_entry)
            if notion_payload:
                response =
requests.patch(f"https://api.notion.com/v1/pages/{existing_page_id}",
headers=notion_headers, json=notion_payload)
                if response.status_code == 200:
                    print(f"Updated for '{jira_title}' in Notion.")
                else:
                    print("Error updating  in  Notion:", response.status_code)
```

Notion to jira.py

```
  Update JIRA issue fields based on Notion data
for jira_entry in jira_data:
    jira_title = jira_entry["Title"]
    jira_duedate = jira_entry["duedate"]

    existing_entry = existing_data.get(jira_title)

    if existing_entry:

        if existing_entry["duedate"] != jira_duedate:
            # Update JIRA issue with new due date
            jira_issue_key = existing_entry["jira_issue_key"]  # Assuming you have the
issue key in existing_data
            jira_issue_url =
f"https://tanishqa.atlassian.net/rest/api/3/issue/{jira_issue_key}"
            jira_issue_payload = {
                "fields": {
                    "duedate": existing_entry["duedate"]
                }
            }
            response = requests.put(
                jira_issue_url,
                headers=headers,
                json=jira_issue_payload,
                auth=("tanishqagargdevi@gmail.com", JIRA_AUTH_TOKEN)
            )
            if response.status_code == 204:
                print(f"Updated due date for '{jira_title}' in JIRA.")
            else:
                print("Error updating due date in JIRA:", response.status_code)
                print(response.json())
    else:
        print(f"Title '{jira_title}' not found in existing Notion data.")
```

PROJECT PREVIEW



LIMITATIONS OF THE PROJECT

- The success of the integration depends on the availability and stability of the Jira and Notion APIs, which can change over time.
- Maintaining the integration may require updates as Jira and Notion introduce new features or changes to their APIs.

FUTURE MODIFICATIONS OF THE PROJECT:

- Integrating with other project management or documentation tools to create a more comprehensive ecosystem
- Implementing error handling and reporting mechanisms for better visibility into data synchronization issues.
- Collaborating with other developers or teams to share the integration as an open-source project or a commercial product.

ANALYSIS

In embarking on these projects, I have ventured into various domains of technology and data manipulation. The first project involved web scraping to gather local business data, addressing the need for easy access to such information. Through this, I honed my skills in parsing HTML, handling HTTP requests, and ethical data usage. The API integration project further expanded my expertise, converting pin codes to area codes, facilitating location-based queries. While dealing with REST APIs and JSON data formats, I encountered the challenge of external API dependencies, prompting a consideration for data reliability. These projects showcased my adaptability and training in managing web data.

Subsequently, I delved into web development with Bootstrap and automation using Selenium to streamline form filling processes. This not only improved user convenience but also added to my proficiency in web development and automation. The integration project between Jira and Notion for data synchronization emphasized the importance of seamless collaboration in project management. While working on API integrations and webhooks, I understood the significance of stable APIs and complex data handling. These projects collectively represent a diverse skill set, highlighting my commitment to mastering new technologies and enhancing user experiences.

CONCLUSION

In conclusion, these projects collectively represent my journey into the realms of data manipulation, web development, automation, and seamless collaboration. They underline the versatility and adaptability I have demonstrated in handling various technological challenges. From web scraping to API integration, each project offered unique insights into data extraction, transformation, and utilization, reinforcing my commitment to ethical data practices. The projects involving web development and automation exhibited my dedication to enhancing user experiences and streamlining processes. Lastly, the Jira and Notion integration project underscored the importance of efficient data synchronization in project management, highlighting the need for stable APIs and intricate data handling. Overall, these projects showcase my ever-evolving skill set and a continuous pursuit of excellence in the ever-evolving landscape of technology and data.

BIBLIOGRAPHY

- YouTube
- Code with Harry: https://www.codewithharry.com/
- Flask Documentation: https://flask.palletsprojects.com/en/2.3.x/
- Selenium Documentation: https://selenium-python.readthedocs.io/index.html
- Bootstrap: https://getbootstrap.com/