



## DECLARATION

I hereby declare that the Project report entitled "**Web Scrapping using Python and Flask**" submitted to the Department of Computer Science & Engineering, SIET Nilokheri in partial fulfilment of the requirements for the award of the degree of Bachelor of Technology (ComputerEngineering) is a record of original work done by me, under the guidance and supervision of Mrs. Rama and it has not formed the basis for the award of any Degree/Diploma/Associateship/ Fellowship or other similar title to any candidate of any University.

Dated

Signature of the Candidate

## Approval or Certification

MSME Technology Center Bhiwadi Domain in Python approved Certificated by this Bhiwadi center.

  
**MSME TECHNOLOGY CENTRE BHIWADI**  
Government of India Society  
Ministry of Micro, Small and Medium Enterprises (MSME)

---

Ref: MSME/STC/23-24/2023BWD080675

Date: 22-Sept-2023


**TO WHOMSOEVER IT MAY CONCERN**

This is to certify that **Ms. BHARTI RAWAT** of **State Institute of Engineering & Technology, Nilokheri** has successfully completed Training cum Internship Program in **PYTHON CERTIFICATION COURSE** in our organization, duration **6 WEEK- (09-AUG-23 TO 22-SEPT-23)** in the academic year 2023-2024.

During the Training, Trainee was trained on **PYTHON CERTIFICATION COURSE, INSTALLING PYTHON & PYTHON IDES., PYTHON DATA STRUCTURES, FLOW CONTROL STATEMENTS, OBJECT ORIENTED PROGRAMMING, INHERITANCE, NUMPY, PANDAS, MATPLOTLIB ETC.**

During the tenure of training, we found trainee very sincere attentive and good behavior & We wish best of luck in all endeavors.

  
**Mr. SWAPNIL KERE**  
(COURSE CO-ORDINATOR)



  
**Mr. RAKESH PATI**  
(SENIOR MANAGER-TRAINING)

Fig. 1.1 Certificate

## Certificate

It is certified that the work contained in the project report titled “**Web Scrapping using Python and Flask,**” by “Bharti Rawat” has been carried out under my/our supervision and that this work has not been submitted elsewhere for a degree.

Signature of Supervisor

Name:

Department:

Signature of Head of Department

State Institute of Engg. & Tech.

Nilokheri

## **Acknowledgement**

The acknowledgement page depicts the gratitude, respect and thankfulness of the student towards the people who helped him in pursuing the project successfully and ensured successful completion and implementation of the project. In this page, the author expresses his gratitude and concern by using praising and thanks giving words.

## **Abstract**

"Web Scraping with Python and Flask: This project involves creating a web application using Flask to perform web scraping. Users input a URL, triggering a Python script to extract and display relevant data from the specified webpage. The application employs BeautifulSoup for parsing HTML and requests for fetching web content. User-friendly interfaces are implemented using Flask forms, allowing users to customize scraping parameters. The scraped data is presented dynamically on the web page.

## List of Figures

Fig. ii(1) Certificate	ii
Fig. 1.2 Web Scraping	1
Fig. 5.1 Output	12

# TABLE OF CONTENTS

	Page No.
Declaration	ii
Approval or Certification	iii
Certificate	iv
Acknowledgement	v
Abstract	vi
List of Figures	vii
Table of Content	viii
<b>Chapter</b>	
1 Introduction	1
2 Literature Survey	4
3 Methodology Used	6
4 Source Code of Project	8
5 Results / Snapshots	12
6 Conclusion and Recommendation	13
References	14



# **A PROJECT REPORT ON**

## **Web Scraping using**

# **PYTHON AND FLASK**

Project report submitted in partial fulfillment of the requirement for the degree of  
Bachelor of Technology (Computer Engg.)



Submitted By:

Bharti Rawat

(8721111)

**DEPARTMENT OF COMPUTER ENGINEERING**  
**STATE INSTITUTE OF ENGINEERING & TECHNOLOGY, NILOKHERI**  
**(KURUKSHETRA UNIVERSITY, KURUKSHETRA)**  
**(2023 – 2024)**

# CHAPTER 1 - INTRODUCTION

## 1.1 Introduction of Web Scraping with Python and Flask

In today's data-driven world, information is abundant on the internet, and extracting valuable insights from the web can be crucial for various applications. Web scraping is a powerful technique that allows us to automate the extraction of data from websites. In this project, we aim to develop a web scraping application using Python and Flask, a web framework for building web applications.



Fig. 1.2 Web Scraping

## 1.2 Objective

1. Data Extraction: The primary goal of the project is to create a web scraper that can navigate through websites, extract relevant data, and store it for further analysis. This could include scraping product details, news articles, or any other publicly available information.

2. User Interface with Flask: Implement a user-friendly web interface using Flask that allows users to input URLs, specify data extraction parameters, and initiate the

scraping process. Flask will serve as the backend framework to handle user requests and manage the scraping tasks.

3. Data Storage: Develop a mechanism to store the scraped data efficiently. This could involve using a database such as SQLite or MongoDB to organize and manage the collected information.

4. Scalability: Design the web scraper to be modular and scalable, enabling users to easily add new websites for scraping by extending the functionality of the application.

### **1.3 Technologies Used:**

1. Python: The core programming language for building the web scraper logic and backend functionality.

2. Flask: A lightweight web framework for Python, used to create the user interface and handle HTTP requests.

3. BeautifulSoup: A Python library for pulling data out of HTML and XML files. It will be used for parsing and navigating the HTML content of web pages.

4. Requests: A Python library for making HTTP requests. It will be used to fetch the HTML content of web pages.

5. Database (e.g., SQLite or MongoDB): To store and manage the scraped data.

## **1.4 Workflow**

1. User Input: Users input the URL of the website they want to scrape and specify the data they are interested in.
2. Scraping Logic: The web scraper, powered by BeautifulSoup, navigates through the HTML structure of the provided URL, extracts the specified data, and processes it.
3. Data Storage: The scraped data is stored in a database for easy retrieval and analysis.
4. User Interface: Flask provides a user interface where users can interact with the application, monitor the scraping process, and access the collected data.

## **1.5 Conclusion**

This project aims to showcase the power of web scraping using Python and Flask, providing users with a versatile tool to extract valuable information from the web efficiently. The combination of a robust scraping engine, a user-friendly interface, and data storage capabilities makes this project a valuable asset for researchers, analysts, and anyone seeking to harness the wealth of data available on the internet.

## 2. CHAPTER - LITERATURE SURVEY

Web scraping and web development using Python and Flask have gained significant attention in both academia and industry. Various research papers, articles, and tutorials explore different aspects of web scraping, data extraction, and web application development. Here is a brief literature survey covering key points relevant to this project:

### 1. Web Scraping Techniques:

- "Web Data Extraction Techniques: A Systematic Literature Review" by Rizwana Kausar et al. provides an overview of different web scraping techniques and methodologies.
- "A Survey of Web Information Extraction Systems" by R. Khusro and A. Kumar explores various methods used for extracting information from the web.

### 2. Python for Web Scraping:

- "Python Web Scraping: A Comprehensive Guide" by Ryan Mitchell is a widely recognized book that delves into the basics of web scraping using Python.
- "Web Scraping with Python: A Comprehensive Guide" by J.R. Parker provides practical insights into Python-based web scraping and data extraction.

### 3. Flask for Web Development:

- "Flask Web Development: Developing Web Applications with Python" by Miguel Grinberg is a comprehensive guide on web development using Flask.
- "Building Web Applications with Flask" by Italo Maia offers practical examples and best practices for developing web applications with Flask.

### 4. Best Practices and Ethics in Web Scraping:

- "Web Scraping: A Review of Best Practices" by Frank McCown et al. discusses ethical considerations, challenges, and best practices in web scraping.
- "Legal and Ethical Issues of Web Scraping" by Sebastian Manhart explores the legal and ethical aspects associated with web scraping activities.

### 5. Data Storage and Management:

- "Data Storage Techniques in Web Scraping" by A. Jain and S. Kumar surveys different methods for storing and managing data obtained through web scraping.

- "Web Scraping and Databases: A Survey" by V. Lakshmi and N. Meenakshi provides insights into integrating web scraping with various database systems.

#### 6. Scalability in Web Scraping:

- "Scalable Web Scraping with MongoDB and Python" by M. Khan provides guidance on building scalable web scraping solutions using MongoDB as a backend storage system.

- "Scalable Web Scraping Using Distributed Systems" by S. Ravi et al. explores the use of distributed systems to enhance the scalability of web scraping applications.

These references cover a range of topics related to web scraping, Python programming, Flask development, ethical considerations, and data storage techniques. By leveraging insights from these works, this project aims to integrate best practices and contribute to the existing body of knowledge in the field of web scraping and web application development.

### 3. CHAPTER - METHODOLOGY USED

The methodology for developing the Web Scraping with Python and Flask project involves several key steps, from planning and design to implementation and deployment. Here's an overview of the methodology used in this project:

1. **\*\*Requirements Analysis:\*\***

- Identify the project's goals and objectives.
- Define the scope of the web scraping application, specifying the types of websites it will support and the data to be extracted.

2. **\*\*Technology Stack Selection:\*\***

- Choose Python as the core programming language for its versatility and extensive libraries.
- Select Flask as the web framework to build the user interface and handle backend functionalities.
- Opt for BeautifulSoup and Requests libraries for web scraping operations.

3. **\*\*System Design:\*\***

- Define the architecture of the web scraping application, outlining the components and their interactions.
- Design the user interface with Flask templates to ensure a responsive and user-friendly experience.
- Plan the database schema for storing scraped data efficiently.

4. **\*\*Web Scraping Logic:\*\***

- Implement web scraping logic using BeautifulSoup to parse HTML content.
- Utilize the Requests library to fetch HTML content from specified URLs.
- Handle dynamic content loading and pagination if necessary.
- Extract relevant data based on user-defined parameters.

5. **\*\*User Interface Development:\*\***

- Use Flask to create routes and views for the web application.
- Design HTML templates for user input forms, result displays, and progress tracking.
- Implement interactivity with JavaScript if required for a smoother user experience.

6. **\*\*Database Integration:\*\***

- Integrate a database (e.g., SQLite or MongoDB) to store the scraped data.
- Develop functions to interact with the database, including storing and retrieving data.

#### 7. **\*\*Scalability Considerations:\*\***

- Design the web scraper to be modular, allowing easy addition of new scraping modules for different websites.
- Consider scalability by optimizing code and adopting best practices for handling large volumes of data.



## 4.CHAPTER - Source code of project

### A) HTML Code

```
<!DOCTYPE html>
<html lang="en" style="background-image:
url('../static/image/wallpaperflare.com_wallpaper.jpg');
background-attachment: fixed;">

<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  <title>TecNews Post</title>
  <!--<link rel="stylesheet"
href="{ {url_for('static',filename='css//style.css')}}">-->
  <link href="https://unpkg.com/tailwindcss@^2/dist/tailwind.min.css"
rel="stylesheet">

<body>
  <div class="mb-10 top-1 w-screen flex flex-col justify-center items-
center">
    
  </div>
  <div class="bottom-6 m-auto relative" id="container"
style="width:614px; height:700px;">
    <img src="" id="bg1" width="614px" height="700px">
    <span class="font-mono text-lg absolute text-white whitespace-pre-
line text-justify" id="News"
style="width: 550px; top: 100px; left: 30px; line-height: 2em;">
      { {News|safe}}
    <!-- These curly braces are a gateway for python variables and
functions (via flask) to Front-end. This is done with the help of Jinja
template engine.-->
  </span>
</div>
  <div id="canvasWrapper" class="mb-10 w-screen flex justify-center
items-center">
    <button
```

```

class="text-white p-3 rounded-md font-sans text-xl bg-gradient-
to-r from-green-400 to-blue-500 hover:from-pink-500 hover:to-yellow-
500"
    onclick="download()">Download Post</button>
</div>

<script>
    var newbg = document.getElementById("bg1");
    var background = new Image();
    background.src = "static/image/" + (Math.floor(Math.random() *
(7)) + 1) + ".jpg";
    newbg.src = background.src;
    console.log(newbg.src);
</script>
<script src="/static/js/h2c.min.js"></script>
<script>
    function download() {
        html2canvas(document.getElementById("container"), { height:
700, width: 614 }).then(canvas => {
            //document.body.appendChild(canvas);
            var a = document.createElement('a');
            a.href =
canvas.toDataURL("image/jpeg").replace("image/jpeg", "image/octet-
stream");
            a.download = 'somefilename.jpg';
            a.click();
        });
    }
</script>
</body>
</html>

```

## **B) CSS Code**

```
.gfg {  
    margin: 3%;  
    position: relative;  
    height: 700px;  
    width: 614px;  
}  
  
.first-txt {  
    font-family: monospace;  
    font-size: large;  
    position: absolute;  
    line-height: 2em;  
    top: 100px;  
    left: 30px;  
    color: rgb(255, 255, 255);  
    width: 550px;  
    white-space: pre-line;  
    text-align: justify;  
}
```

## **C) JavaScript Code**

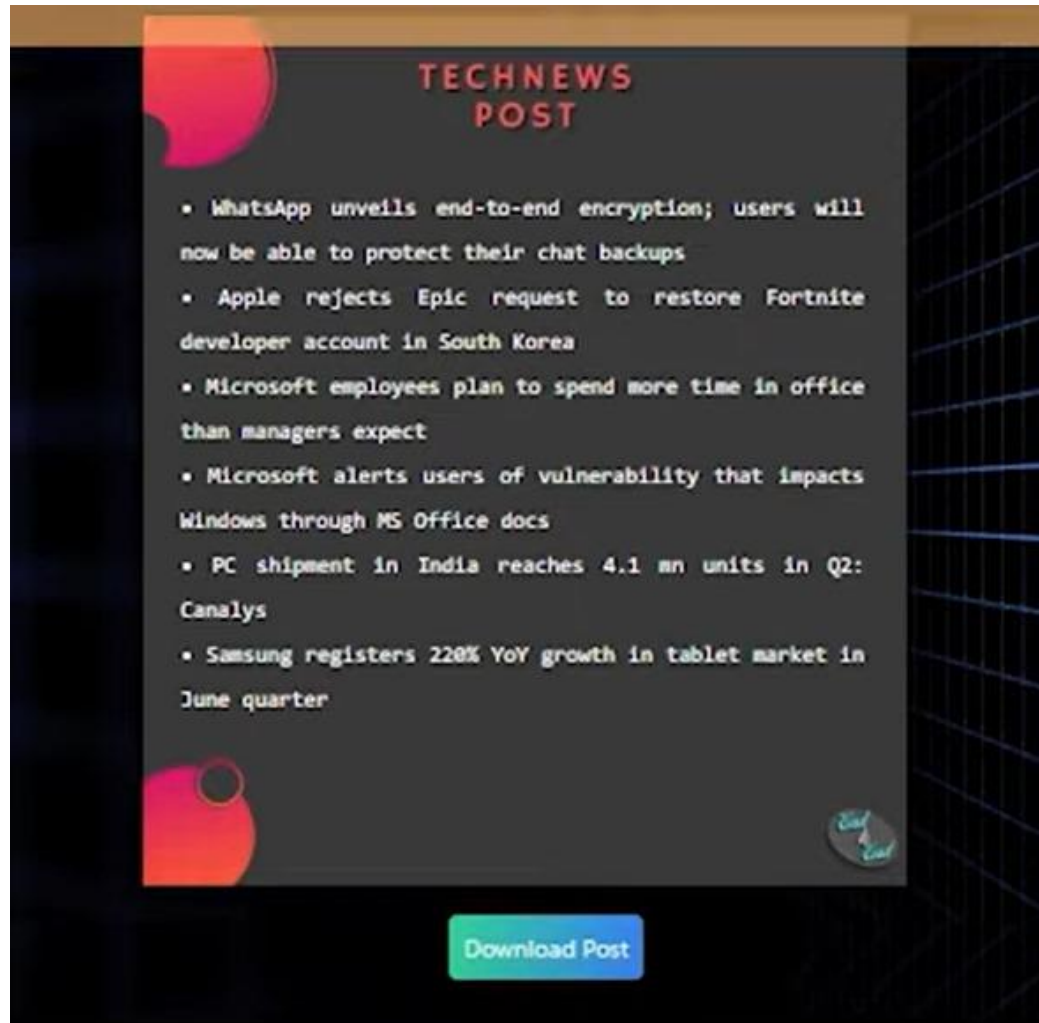
```
!function(A,e){ "object"==typeof exports&&"undefined"!=typeof  
module?module.exports=e():"function"==typeof  
define&&define.amd?define(e):(A="undefined"!=typeof  
globalThis?globalThis:A||self).html2canvas=e() }(this,function(){ "use strict";
```

## D) Python Code

```
from flask import Flask,url_for,render_template,request
from bs4 import BeautifulSoup
import requests

app = Flask(__name__)
@app.route('/',methods=["GET","POST"])
def index():
    url = "https://www.businessstoday.in/technology/news"
    req = requests.get(url)
    soup = BeautifulSoup(req.content, 'html.parser')
    #outerData = soup.find_all("div",class_="widget-listing",limit=6)
    #print(outerData)
    finalNews=""
    for data in soup.find_all("div",class_="widget-listing",limit=6):
        news=data.div.div.a["title"]
        finalNews += '\u2022 '+news+'\n'
    #print(finalNews)
    return render_template("index.html",News=finalNews)
```

#### 4. CHAPTER – Conclusion and Recommendation



## **6. CHAPTER - Conclusion and Recommendation**

The "Web Scraping with Python and Flask" project has successfully developed a versatile and user-friendly tool for automating data extraction from the web. Leveraging Python's rich libraries and Flask's web framework, the application offers an intuitive interface for users to specify data extraction parameters and seamlessly navigate through websites. The modular and scalable design, coupled with efficient data storage capabilities, ensures adaptability to diverse requirements and ease of managing collected information. The comprehensive documentation facilitates easy installation and usage, while ethical considerations underscore responsible scraping practices. Moving forward, it is recommended to continue refining the application, addressing potential security concerns, and exploring further optimization for enhanced performance. Additionally, the project could benefit from community contributions and continuous updates to accommodate evolving web technologies and user needs.

## References

- <https://www.w3schools.com/>
- <https://careerfoundry.com/en/tutorials/web-development-for-beginners/introduction-to-web-development/>
- <https://www.Udemy.com/bootstrap/>
- <https://www.codecademy.com/catalog/subject/web-development>
- <https://www.geeksforgeeks.org/web-development/>
- <https://getbootstrap.com/>
- <https://www.javatpoint.com/html-tutorial>
- <https://www.javatpoint.com/css-tutorial>
- <https://www.javatpoint.com/javascript-tutorial>
- <https://www.javatpoint.com/bootstrap-tutorial>
- <https://stackoverflow.com/>