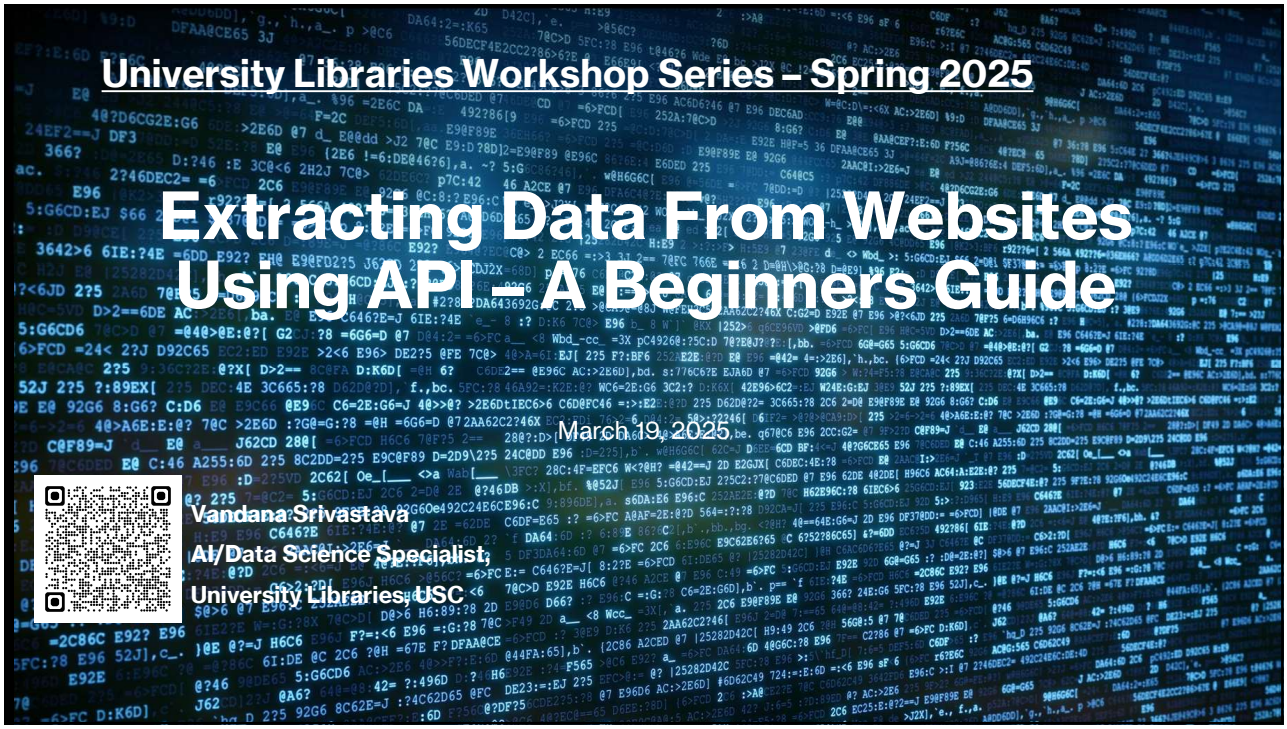
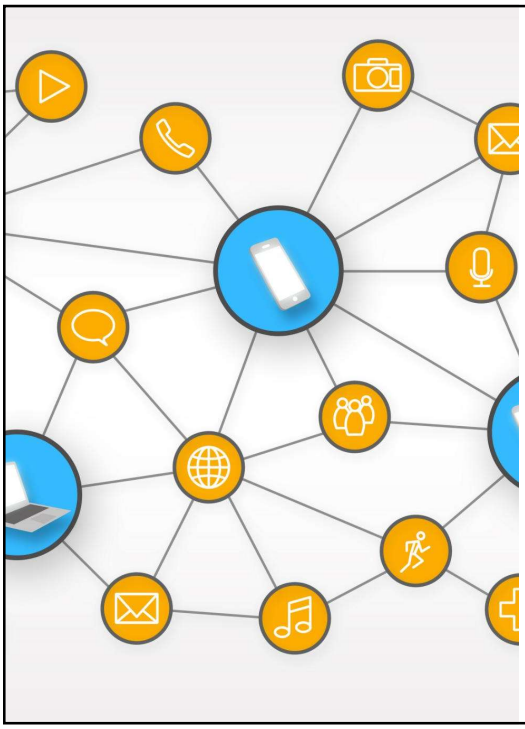



1

## Agenda Items

- Understanding APIs
- Setting Up for API Data Extraction
- Techniques for Extracting Data Using NY Times API
- Data Processing and Storage
- Best Practices and Considerations


2



# Definition and Purpose of APIs

**What is an API?**

**- A**pplication **P**rogramming **I**nterface

- An API defines the methods and data formats that applications use to communicate effectively with each other.

**Purpose of APIs**

The primary purpose of APIs is to enable integration between different software applications, enhancing functionality and interoperability.

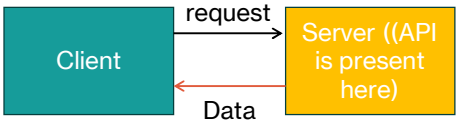
**Benefits for Developers**

APIs allow developers to leverage existing services without rebuilding them, saving time and resources in the development process.

3

# How API Works?

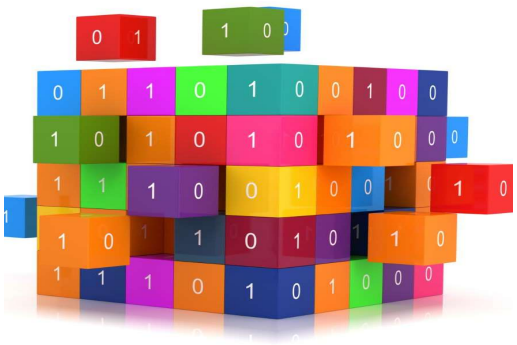
- API allows communication and sharing of information or data between two programs or softwares
- To get a desired information from the server, you as the client would **send a request**
  - which is a combination of URL and HTTP methods.
  - In addition to the method, we can also add request parameter or header to modify the response we get from the server, sent by the API.
- Using API to extract data is straight forward
  - it provides access to dedicated section of the required data through an endpoint
  - Reduces time to extract data




https://medium.com/@favourphilic/how-to-extract-data-using-api-0fce951e8a79

4

# Types of APIs (REST, SOAP, GraphQL)



**REST APIs**

REST APIs use a **stateless communication*** protocol and are widely used for web services, enabling scalability and flexibility.

**SOAP APIs**

SOAP APIs are protocol-based and rely on XML for message formatting, providing a rigid structure and robust security features.

**GraphQL APIs**

GraphQL allows clients to request exactly the data they need, making it efficient and flexible for modern applications.

* each request from a client to the server is treated independently, and the server does not maintain any session information or state between requests, requiring all necessary data to be included within each request itself.

5

# Common Use Cases for APIs

**Data Retrieval**

APIs enable the retrieval of data from online services, allowing applications to access real-time information easily.

**Application Integration**

APIs facilitate the integration of third-party applications, enhancing functionality and interoperability between different systems.

**Task Automation**

APIs are used to automate repetitive tasks, making processes more efficient and reducing manual effort.

**Payment Processing**

APIs simplify payment processing, allowing businesses to securely handle transactions through various payment gateways.

6

# Registering and Obtaining API Keys

### Account Registration

To start using an API, you must first register for an account with the API provider.

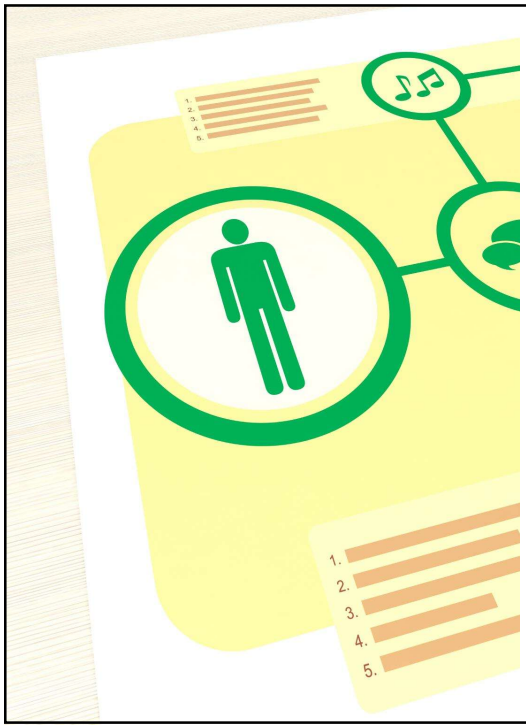
### Obtaining API Keys

After registration, you can request your unique API keys, which are essential for accessing the API.

### Authentication Process

API keys are used to authenticate your requests, ensuring secure access to the API's resources.

7



# API Documentation and Endpoints

**Importance of API Documentation**

API documentation is essential for developers as it guides them in understanding how to properly use an API.

**Endpoints and Methods**

Understanding endpoints and request methods is vital for making successful API calls and ensuring proper data retrieval.

**Parameters and Response Formats**

Documentation includes details on parameters and response formats, crucial for handling requests and interpreting responses correctly.

8

## Authentication and Authorization Methods

**Importance of Authentication**

Authentication is essential for APIs to verify user identity and protect sensitive data from unauthorized access.

**API Keys**

API keys are unique identifiers used to authenticate requests made to an API, providing a basic level of security.

**OAuth Tokens**

OAuth tokens allow users to grant third-party applications access to their data without sharing passwords, enhancing security.

**Basic Authentication**

Basic Authentication encodes user credentials in a simple format, but is less secure compared to other methods.

9

# Use Case: Extracting NY Times Article Data Using Their API

10

# Step 0: What Data Should Be Extracted?

- **Decide on the kind of data you are interested in.**
- For example, we are interested in:

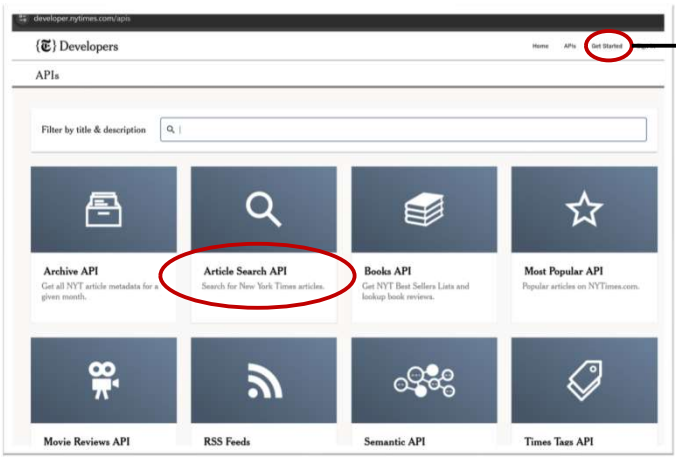
*Extracting articles from first page of the NY Times newspaper since 2000*

- **Read website requirements and specs for downloading data using API**
  - New York Times has rate limits to call their API.
  - They allow 500 requests per day and 5 requests per minute.
  - You should sleep 12 seconds between calls to avoid hitting the per minute rate limit.
  - If you need a higher rate limit, please contact at code@nytimes.com. (https://developer.nytimes.com/faq).

11

# STEP 1: Getting API Key

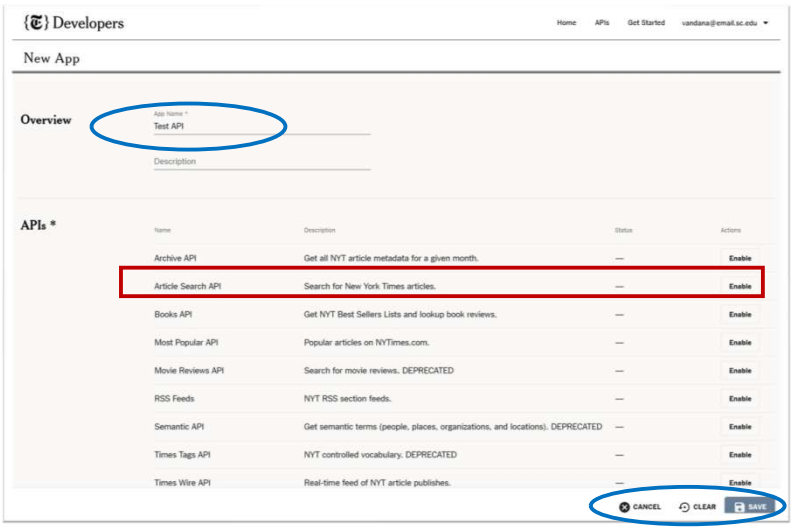
1. Go to *developer.nytimes.com/apis*





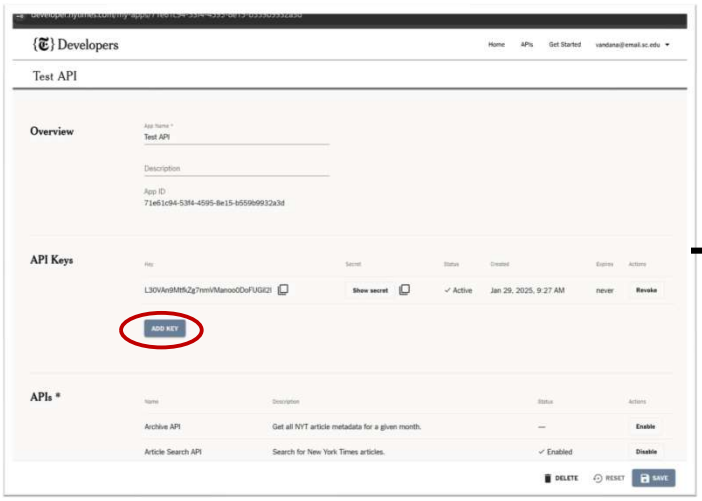
12

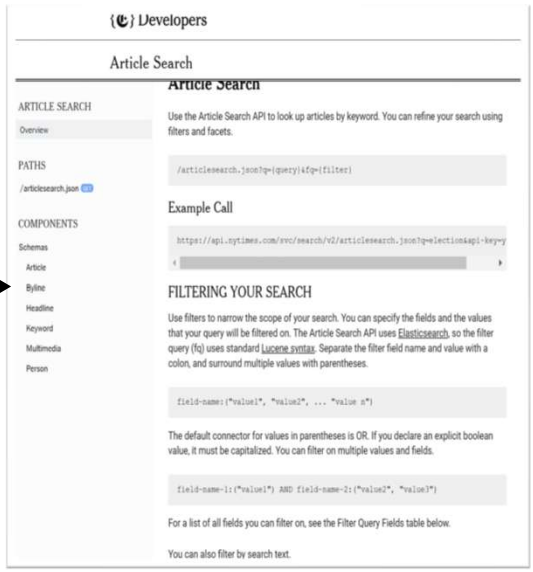
# Step 1 continued: Getting API Key


13

# Step 1 continued: Getting API Key




14

# Extracting Data Using Python

Install Important Packages

1

```
pip install requests
```

```
Requirement already satisfied: requests in c:\users\bipva\anaconda3\lib\site-packages (2.31.0)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\bipva\anaconda3\lib\site-packages (from requests) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in c:\users\bipva\anaconda3\lib\site-packages (from requests) (3.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\bipva\anaconda3\lib\site-packages (from requests) (2.0.7)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\bipva\anaconda3\lib\site-packages (from requests) (2024.6.2)
Note: you may need to restart the kernel to use updated packages.
```

```
import pandas as pd
import requests
import json
```

2

12 second delay in sending request -- fetching daily articles except Sunday

```
from datetime import datetime, timedelta
import time  # Added to use sleep for delay

API_KEY = 'fXwHTwqM3iAB9pia59aGsWIh0W0kbUeh' # Use your API key
```

3

```
def fetch_articles(start_date, end_date):#, page=0):
    filter = 'print_page:1 AND (print_section:("A") OR (!_exists_:print_section))'
    url = f"https://api.nytimes.com/svc/search/v2/articlesearch.json?fq={filter}"
    params = {
        'begin_date': start_date.strftime('%Y%m%d'),
        'end_date': end_date.strftime('%Y%m%d'),
        'api-key': API_KEY
    }

    try:
        response = requests.get(url, params=params)
        response_data = response.json()
        # Debug: print the raw response if there's an issue
        print(f"Fetching data for {start_date}: {response.status_code}")

        # Check if the 'response' key exists
        if 'response' in response_data and 'docs' in response_data['response']:
            return response_data['response']['docs']
        else:
            # Handle missing 'response' key
            print(f"No 'response' found for {start_date}. Full response: {response_data}")
            return []

    except Exception as e:
        # Print error details
        print(f"Error fetching data for {start_date}: {e}")
        return []
```

Function to fetch articles

15

# Extracting Data Using Python continued...

Fetching article between Jan 5 and Jan 10, 2000
- Data is fetched and aved as JSON object
- Convert JSON to pandas dataframe

4

```
# Fetch all front-page articles from 5th Jan 2000 to 10th Jan 2000
def fetch_front_page_data(start_year, end_year):
    current_date = datetime(start_year, 1, 5)   # make changes accordingly
    end_date = datetime(end_year, 1, 10)        # make changes accordingly
    front_page_data = []

    while current_date <= end_date:
        articles = fetch_articles(current_date, current_date)
        if articles:
            front_page_data.extend(articles)

        # Add 12-second delay between each API call
        time.sleep(12)

        current_date += timedelta(days=1)
    return front_page_data

# Fetch front-page data for the years 2000 to 2000
front_page_data_2000_to_2000 = fetch_front_page_data(2000, 2000)

# Save to a file
with open('nyt_front_page_2000_2000.json', 'w') as f:
    json.dump(front_page_data_2000_to_2000, f)

print(f"Data fetched successfully. Number of articles: {len(front_page_data_2000_to_2000)}")
```

The code considers the 12 seconds sleep between API calls.

Each request fetches 10 data points (news articles) from page 1.

Read the data in pandas dataframe

```
import pandas as pd
df = pd.read_json('nyt_front_page_2000_2000.json')
df
```

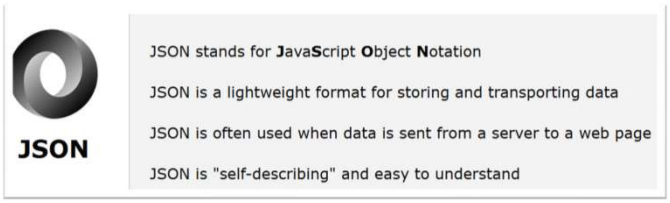
Read 2 columns "headline" and "publication date" -- similarly data from other columns can be seen

```
df[['headline', 'pub_date']]
```

| | headline | pub_date |
|---|---|---|
| 0 | {'main': 'STOCK PRICES DROP IN U.S. AND EUROPE... | 2000-01-05T05:00:00+0000 |
| 1 | {'main': 'Computer Technicians Learn They Are ... | 2000-01-05T05:00:00+0000 |
| 2 | {'main': 'Jets Are on a Losing Streak: 2 Coach... | 2000-01-05T05:00:00+0000 |
| 3 | {'main': '$8 Million Offered to End Attica Inm... | 2000-01-05T05:00:00+0000 |
| 4 | {'main': 'ENDORSED BY DOLE, BUSH STEPS UP RUN ... | 2000-01-05T05:00:00+0000 |

16

# Storing API Data (JSON Format)




SON Example

his example defines an employees object: an array of 3 employee records (objects):

```
{
"employees":[
    {"firstName":"John", "lastName":"Doe"},
    {"firstName":"Anna", "lastName":"Smith"},
    {"firstName":"Peter", "lastName":"Jones"}
]
}
```

JSON Syntax Rules

- Data is in name/value pairs
- Data is separated by commas
- Curly braces hold objects { }
- Square brackets hold arrays [ ]

https://www.w3schools.com/whatis/whatis_json.asp

17

# Making API Requests (GET, POST, PUT, DELETE)

**GET Method**

The GET method is used to retrieve data from a server. It is the most commonly used HTTP request method.

**POST Method**

The POST method is used to send data to a server. It is typically used for submitting forms or creating new resources.

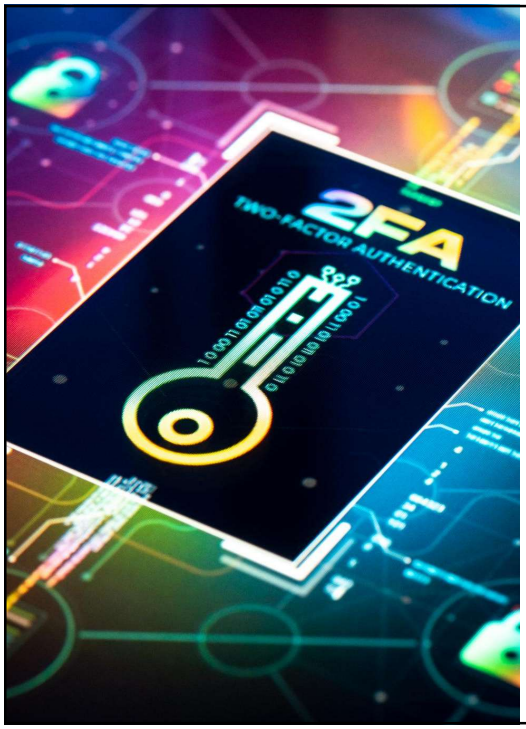
**PUT Method**

The PUT method is used to update existing data on a server. It replaces the current representation of the target resource.

**DELETE Method**

The DELETE method is used to remove data from a server. It is a way to delete resources from a web application.

18




## Handling Responses and Error Codes

**Importance of Response Handling**

Proper response handling is crucial for understanding the data returned by API calls and ensuring correct application behavior.

**Parsing Returned Data**

Parsing the returned data allows developers to extract necessary information and utilize it effectively within applications.

**Managing Error Codes**

Managing error codes is essential for troubleshooting and ensuring that applications can gracefully handle unexpected issues.

19

# Data Processing and Storage

20

# Parsing JSON and XML Responses

**Importance of Data Formats**

APIs often deliver data in JSON or XML formats, which are essential for web and application development.

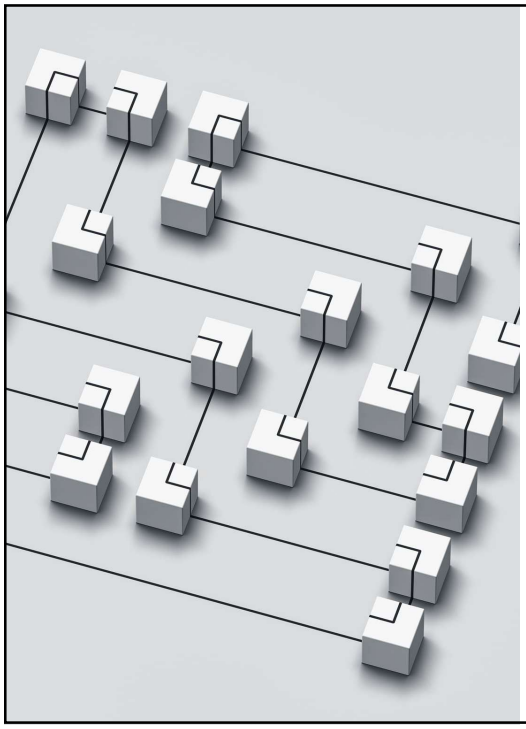
**Parsing Techniques**

Understanding different parsing techniques for JSON and XML enables developers to extract and use data efficiently.

**Integration into Applications**

Effective parsing is crucial for integrating JSON and XML data into applications, improving functionality and user experience.

21



# Storing Data in Databases

**Importance of Storage**

Storing processed data correctly in a database is essential for maintaining data integrity and availability.

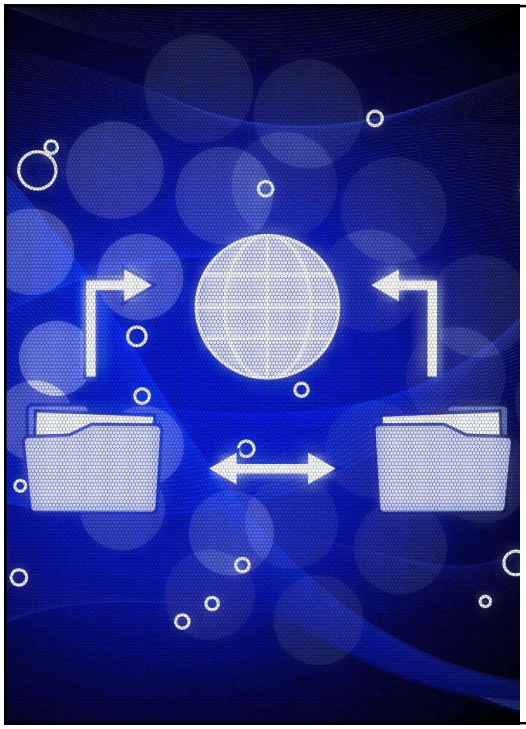
**Choosing the Right Database**

Selecting the appropriate database system is crucial for achieving optimal performance and scalability based on the data needs.

**Schema Design**

Effective schema design is key to ensuring efficient data retrieval and management within the database.

22



## Data Cleaning and Validation

**Importance of Data Accuracy**

Ensuring data accuracy is crucial for making informed decisions based on reliable information.

**Filtering Invalid Entries**

The cleaning process involves identifying and removing invalid or irrelevant data entries to improve data quality.

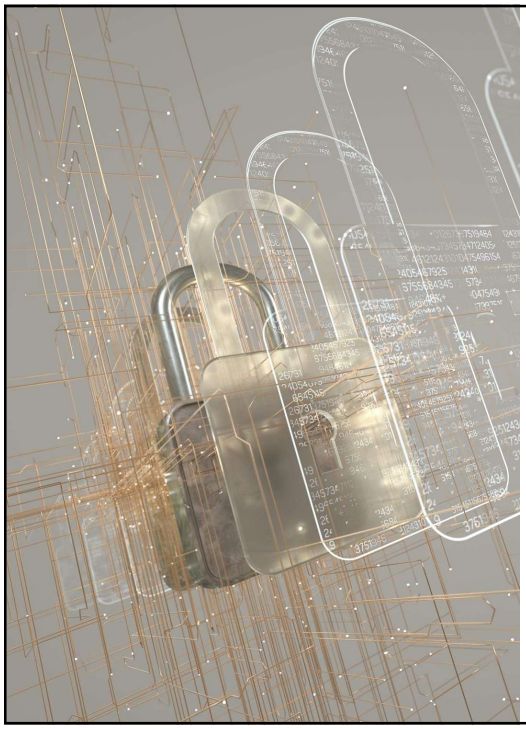
**Correcting Inconsistencies**

Data validation includes correcting discrepancies in datasets to maintain uniformity and reliability.

23

# Best Practices and Considerations

24



# Ensuring Data Security and Privacy

**Importance of Data Security**

Data security is crucial for ensuring that sensitive information is protected against unauthorized access and breaches.

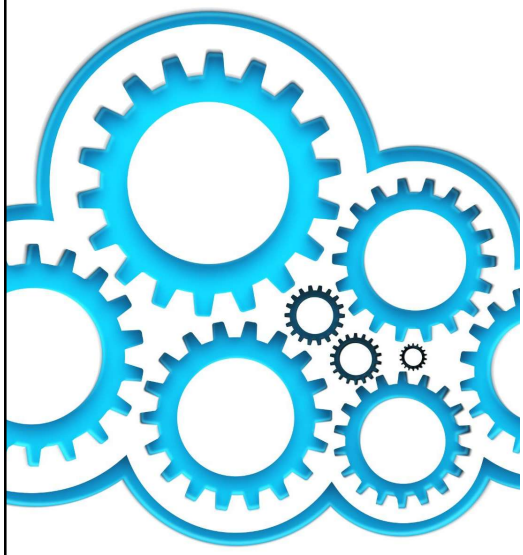
**Use of Encryption**

Encryption is a vital technique for safeguarding data as it transforms information into a secure format that can only be read by authorized users.

**Best Practices for APIs**

Following best practices, such as using secure connections and regular audits, is essential for maintaining data privacy while working with APIs.

25



# Optimizing Performance and Efficiency

**API Performance Optimization**

Optimizing API calls can significantly enhance application performance by reducing latency and improving response times.

**Caching Responses**

Implementing caching mechanisms for API responses helps in reducing repeated data fetches, leading to improved efficiency.

**Minimizing Request Payloads**

Reducing the size of request payloads can decrease processing time and enhance overall application speed.

26

# Monitoring and Maintaining APIs


**Importance of Monitoring**

Regular monitoring is essential to maintain the performance of API integrations and prevent disruptions in service.

**Tracking Usage Patterns**

Keeping track of usage patterns helps identify trends and optimize API performance based on user needs.

**Updating API Keys**

Regularly updating API keys is crucial for security and ensuring uninterrupted access to API services.

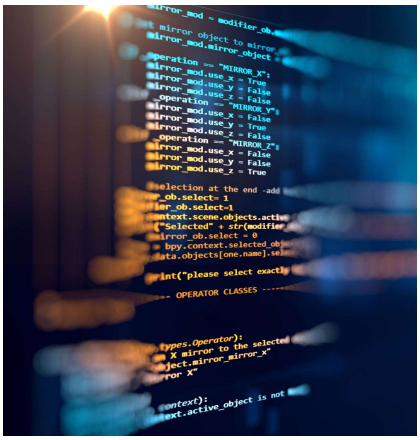
**Troubleshooting Issues**

Prompt troubleshooting of issues as they arise ensures that API integrations continue to function smoothly.

27

# Library Support for Extracting Data

- Assist with data extraction issues
  - Copyright issues
  - Format issues
  - Storage and access issues
- Consult on methods and coding
  - Which platform ?
  - What toolset of methods ?
- Introductory and Intermediate Workshops
- Contact: vandana@email.sc.edu
- Join our listserv at https://sc.libwizard.com/f/digireslib
- Answer short survey – https://sc.libwizard.com/id/5965cdf2b224b9a5f0aac7aeff20bc62



28

28

# Conclusion

**Importance of APIs**
APIs are essential for data extraction and integration in modern applications, making them invaluable tools for developers.

**Best Practices for API Usage**
Understanding and implementing best practices for API usage is crucial for maximizing their effectiveness and reliability.

**Enhancing Applications**
By leveraging APIs correctly, developers can enhance their applications with rich features and streamlined data access.

29

Vandana Srivastava:
vandana@email.sc.edu

# THANK YOU!

**FEEDBACK LINK**

**THANKS IN ADVANCE FOR SUBMITTING FEEDBACK!!**

30

30