SRM
INSTITUTE OF SCIENCE & TECHNOLOGY
Deemed to be University u/s 3 of UGC Act, 1956

SRM IST, NCR CAMPUS, MODINAGAR

DEPARTMENT OF COMPUTER APPLICATIONS

PRACTICAL FILE

INTRODUCTION TO ADVANCED COMPUTING (UDS21102J)

1ST YEAR, 1ST SEMESTER

SESSION 2022 – 2023

SUBMITTED TO: MRS. HIMANI TYAGI

SUBMITTED BY:

REG NO.:

BCA [DATA SCIENCE]

SRM INSTITUTE OF SCIENCE & TECHNOLOGY NCR CAMPUS, MODINAGAR

DEPARTMENT OF COMPUTER APPLICATIONS

Register No:

BONAFIDE CERTIFICATE

Certified to be the bonafide record of the work done by ________________ of BCA [Data Science], First year, 1st Semester for the award of BCA-Data Science degree course in the Department of Computer Applications in **Advanced Computing Lab** during the Academic year-2022-2023.

LAB IN-CHARGE **HEAD OF DEPARTMENT**

Submitted for the university examination held on_______________

INTERNAL EXAMINER-I INTERNAL EXAMINER-II

INDEX

08.	Understand core architectural components of GPU.			
09.	Understand core architectural components of TPU.			
10.	Develop service level indicator and service level Objective compliance			

PRACTICAL 1

Understand and Practice Basic/Advanced Computing functions

The five basic functions of a computer system are input, processing, output, controlling and storage. These five functions and are used to teach the fundamentals of information's systems.

1.Input or inserting data and instruction

Input can be defined as inserting data and instructions needed for a computer.

2.Processing

The process of converting data and inserted to the computer to information as required can be defined as processing.

3.Output or retrieving data or information

Giving the processed data and information to the external forces is done under output.

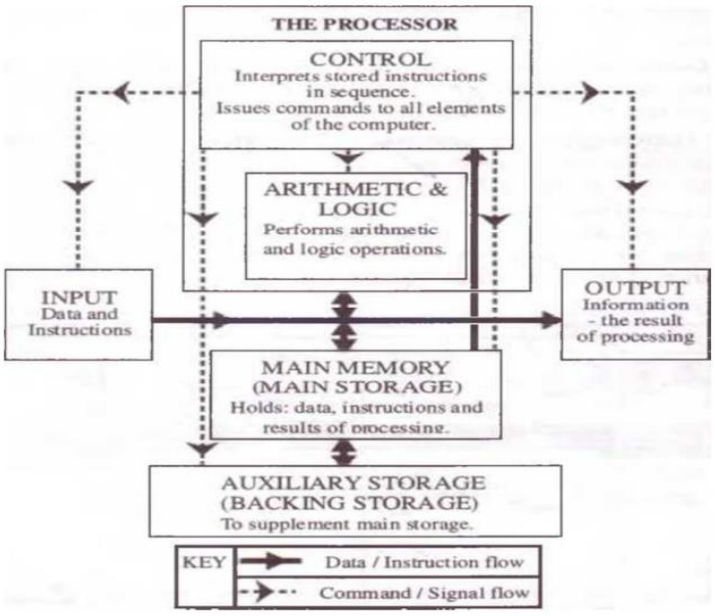
4.Storing data and information

Data and information are stored in a way that they can be reused later as required. This is done under storing.

5.Controlling

The functions of a computer system and its various devices should be controlled properly.

Under controlling the controlling activities done by computer system are considered.


THE PROCESSOR
CONTROL
Interprets stored instructions in sequence.
Issues commands to all elements of the computer.
ARITHMETIC & LOGIC
Performs arithmetic and logic operations.
INPUT
Data and Instructions
OUTPUT
Information - the result of processing
MAIN MEMORY (MAIN STORAGE)
Holds: data, instructions and results of processing.
AUXILIARY STORAGE (BACKING STORAGE)
To supplement main storage.
KEY
Data / Instruction flow
Command / Signal flow

The elements of a Computer System showing its "Logical Structure"

Cloud computing defined

Cloud computing is the on-demand availability of computing resources as services over the internet. It eliminates the need for enterprises to procure, configure, or manage resources themselves, and they only pay for what they use.

Understanding how cloud computing works

Cloud computing service models are based on the concept of sharing on-demand computing resources, software, and information over the internet. Companies or individuals pay to access a virtual pool of shared resources, including compute, storage, and networking services, which are located on remote servers that are owned and managed by service providers.

One of the many advantages of cloud computing is that you only pay for what you use. This allows organizations to scale faster and more efficiently without the burden of having to buy and maintain their own physical data centres and servers.

There are three different cloud computing deployment models: public cloud, private cloud, and hybrid cloud.

- **Public clouds** are run by third-party cloud service providers. They offer compute, storage, and network resources over the internet, enabling companies to access shared on-demand resources based on their unique requirements and business goals.
- **Private clouds** are built, managed, and owned by a single organization and privately hosted in their own data centres, commonly known as “on-premises” or “on-prem.” They provide greater control, security, and management of data while still enabling internal users to benefit from a shared pool of compute, storage, and network resources.
- **Hybrid clouds** combine public and private cloud models, allowing companies to leverage public cloud services and maintain the security and compliance capabilities commonly found in private cloud architectures.

Types of cloud computing

Infrastructure as a Service (IaaS)

IaaS contains the basic building blocks for cloud IT. It typically provides access to networking features, computers (virtual or on dedicated hardware), and data storage space. IaaS gives you the highest level of flexibility and management control over your IT resources. It is most like the existing IT resources with which many IT departments and developers are familiar.

Platform as a Service (PaaS)

PaaS removes the need for you to manage underlying infrastructure (usually hardware and operating systems), and allows you to focus on the deployment and management of your applications. This helps you be more efficient as you do not need to worry about resource procurement, capacity planning, software maintenance, patching, or any of the other undifferentiated heavy lifting involved in running your application.

Software as a Service (SaaS)

SaaS provides you with a complete product that is run and managed by the service provider. In most cases, people referring to SaaS are referring to end-user applications (such as web-based email). With a SaaS offering, you don’t have to think about how the service is maintained or how the underlying infrastructure is managed. You only need to think about how you will use that software.

Benefits of cloud computing

Agility

The cloud gives you easy access to a broad range of technologies so that you can innovate faster and build nearly anything that you can imagine. You can quickly spin up resources as you need them–from infrastructure services, such as compute, storage, and databases, to Internet of Things, machine learning, data lakes and analytics, and much more.

Elasticity

With cloud computing, you do not have to over-provision resources up front to handle peak levels of business activity in the future. Instead, you provision the number of resources that you need. You can scale these resources up or down to instantly grow and shrink capacity as your business needs change.

Cost savings

The cloud allows you to trade fixed expenses (such as data centers and physical servers) for variable expenses, and only pay for IT as you consume it. Plus, the variable expenses are much lower than what you would pay to do it yourself because of the economies of scale.

Deploy globally in minutes

With the cloud, you can expand to new geographic regions and deploy globally in minutes. For example, AWS has infrastructure all over the world, so you can deploy your application in multiple physical locations with just a few clicks. Putting applications in closer proximity to end users reduces latency and improves their experience.

PRACTICAL 2

Set up Cluster of 4 different computer systems

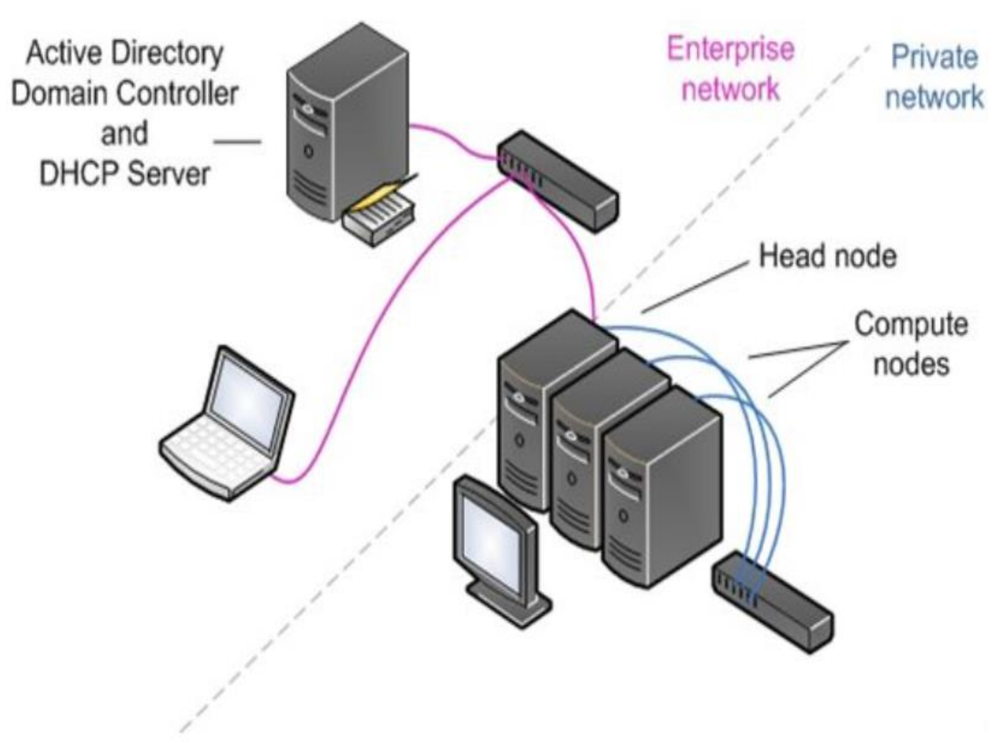
The plan

In this guide, we will walk through the steps to build an HPC cluster that consists of at least three nodes (computers with the server operating system installed). One node acts as a head node (to manage the cluster) and as a compute node (to run jobs). Additional nodes act only as compute nodes.

The cluster nodes are connected to each other on an isolated, private network. This network is basically used to coordinate the workload on your cluster. The head node has an additional network connection to the enterprise network (a network with an Active Directory domain controller that manages logging in, security, and authentication in your work environment). After your cluster is set up, you can submit and monitor jobs by logging in to the head node directly, or by connecting to the cluster from another computer on the enterprise network (if you install the HPC Pack client utilities on that computer).

Because we are building a small cluster, and to keep things simple, we will build a cluster with *preconfigured compute nodes*. That means that the computers that we add to the cluster already have the operating system installed, and we will manually install the HPC software on each node.

The following diagram illustrates the cluster plan:



What you need

Hardware

- 3 (or more) computers with the following recommended specifications*

    - **Processor (x64-based):** 2 GHz or faster
    - **RAM:** 2 GB or more
    - **Available disk space:** 80 GB or more
- Keyboard, monitor, and mouse for the head node
- 1 switch or hub for the private network
- 3 (or more) cables for the private network
- 1 extra Network Interface Card (NIC) for the head node
- 1 cable to connect the head node to the enterprise network

*You can see the minimum system requirements for Windows HPC here (http://go.microsoft.com/fwlink/?LinkId=194785). However, if you want to run specific software applications on your cluster, consult the software documentation for any additional system requirements.

Software

- Windows Server 2008 R2 operating system already installed on each node
- Microsoft HPC Pack 2008 R2 installation media

For information about how to download the software, see Download Windows HPC Server 2008 R2 software.

Network and permissions

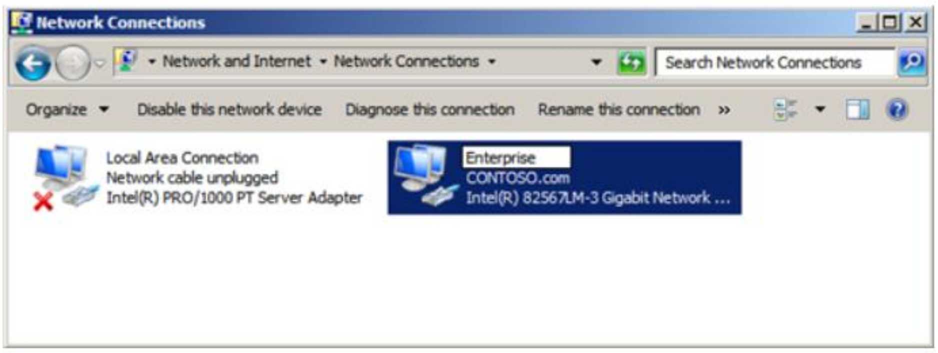
- You must have an Active Directory domain that you can join the cluster to. If you do not have an Active Directory domain controller in your environment, you can create your own domain by setting up the head node to act as a domain controller.
- To set up and use your cluster, you must use domain credentials (credentials that are managed by the Active Directory domain controller).
- You must have permission to add the cluster nodes to the Active Directory domain. This means that you must have permission to create computer objects in Active Directory Domain Services, and join computers to the domain. If you can buy a new computer and add it to your domain without calling your IT department, then you probably have sufficient permissions. If not, work with your domain administrator to add the cluster nodes to the domain. In some cases, a domain administrator can pre-create the computer objects for you in Active Directory Domain Services before you set up your cluster. For more information, see Deploy Nodes with Pre-created Computer Objects in Active Directory (http://go.microsoft.com/fwlink/?LinkId=194363).
- This guide also assumes that there is a DHCP server for your enterprise network. This means that an IP address is automatically assigned to your NIC when you connect it to the enterprise network.

Set up hardware and join a domain

1. Insert the extra NIC into the server that will act as the head node.
2. Connect the new NIC to the enterprise network. Do not connect the private network yet.
3. Plug in the head node, monitor, keyboard, and mouse.
4. Power up the head node.
5. Name the network connections:
    a. Open the **Network and Sharing Centre**.
    b. Click **Change adapter settings.** Connection information for the two NICs is displayed.

   c. Right-click the connection that is plugged in, click **Rename**, and then type "**Enterprise**", as illustrated in the following screenshot:

6. Rename the other connection "Private".
7. Close the Network and Sharing Centre.



Join your domain:

a. In the **Initial configuration tasks** console, click **Provide computer name and domain**.
b.
    a. In the **System Properties** dialog box, in the **Computer Name** tab, click the **Change** button.
    b. In the **Computer Name/Domain Changes** dialog box, type a name for the head node, for example "HEADNODE".
    c. Select **Domain**, type the name of your domain, and then click OK.
    d. When prompted, type the domain credentials that have permission to join the server to the domain. A dialog box appears to confirm the domain join.
    e. When prompted, restart the computer to apply these changes.
c. Connect the private network:
    a. Plug the switch in to a power source.
    b. Connect all nodes to the switch.

Install Microsoft HPC Pack 2008 R2 to create the head node

1. Log on to the head node with your domain credentials.
2. Insert the installation media for Microsoft HPC Pack 2008 R2 and run setup (setup.exe).
3. Follow the steps in the installation wizard, and select the following options:
    a. In **Select Installation Edition**, select **HPC Pack 2008 R2 Express** or **HPC Pack 2008 R2 Enterprise and HPC Pack 2008 R2 for Workstation**.
4. Accept the license terms.
5. In **Select Installation Type**, select **Create a new HPC Cluster by creating a head node**.
6. In the next three screens, accept the default installation locations for the HPC databases and for HPC Pack.
7. In **Select Method for Updates**, select **Use Microsoft Update**.
8. In **Customer Experience Improvement Program**, select **Yes** (this helps us improve the product in later versions).
9. At the end of the wizard, click **Finish**. The wizard starts HPC Cluster Manager, which is the administration console for configuring and monitoring your cluster.

Perform the initial configuration tasks on the head node

1. In HPC Cluster Manager, in the **Deployment To-do List**, click **Configure your network**.
2. In the **Network Configuration Wizard**:
   a. In **Network Topology Selection**, the wizard detects the installed adapters and preselects a topology accordingly. Verify that the selection is as expected (in this guide, we built a Topology 1 cluster).
   b. In **Enterprise Network Adapter Selection**, in the Network adapter drop-down list, select the network connection that you named "Enterprise".
   c. In **Private Network Adapter Selection**, select the network connection that you named "Private".
   d. In **Private Network Configuration**, accept the default settings.
   e. In **Firewall Setup**, accept the default settings.
   f. Click **Configure**.
3. In the to-do list, click **Provide installation credentials**.
   a. In the dialog box, you can type the same domain credentials that you used to join your head node to the domain. When you type your user name, include the domain name in the form DOMAIN\User.
4. In the to-do list, click **Configure the naming of new nodes**.

   a. In the dialog box, accept the defaults and click **OK**.

5. In the to-do list, click **Create a node template**.
6. In the **Create Node Template Wizard**:
   a. In **Choose Node Template Type**, select **Compute node template**.
   b. In **Specify Template Name**, type a name for the template. For example, "Preconfigured nodes".
   c. In **Select Deployment Type**, select **Without operating system**.
   d. In **Windows Updates**, select **Include a step in the template to download and install updates**.
7. Click **Create**.

Pre-configure the compute nodes

For each compute node, perform the following steps

1. Switch the keyboard, monitor and mouse to the compute node.
2. Log on to the node as an administrator.
3. In the **Initial configuration tasks** console, select **Enable remote desktop**.
4. Rename the node, for example NODE01.
5. Join the node to the domain and then restart the computer when prompted.
6. Log on to the node with your domain credentials.
7. Insert the installation media for Microsoft HPC Pack 2008 R2 and run setup (setup.exe).
8. Follow the steps in the installation wizard, and select the following options:
   a. In **Select Installation Edition**, select **HPC Pack 2008 R2 Express** or **HPC Pack 2008 R2 Enterprise and HPC Pack 2008 R2 for Workstation**.
9. Accept the license terms.
10. In **Select Installation Type**, select **Join an existing HPC Cluster by creating a new compute node**.

11. In **Join Cluster**, select or type the name that you gave to your head node (for example "HEADNODE").
12. In the next screen, accept the default installation location for HPC Pack.
13. In **Select Method for Updates**, select **Use Microsoft Update**.
14. At the end of the wizard, click **Finish**.

Join the nodes to the cluster

1. Switch the keyboard, monitor, and mouse to the head node.
2. In HPC Cluster Manager, click **Node Management**. Your preconfigured nodes appear in the node list. At this point, the nodes have **Node State** marked as **Unknown**, and **Node Health** as **Unapproved**.
3. In the node list, select the new nodes, right-click your selection, and then click **Assign node template**.
4. In the dialog box, select the node template that you created, "Preconfigured nodes", and then click **OK**.
5. To monitor the provisioning process:
    a. Select a node in the node list.
    b. In the Details Pane, select the **Provisioning** tab. Provisioning details are displayed.
6. When provisioning is complete, the nodes have **Node State** marked as **Offline**, and **Node Health** as **OK**.
7. Select all the nodes, right-click your selection, and then click **Bring Online**.

PRACTICAL 3

Extraction real time customer feeds from twitter

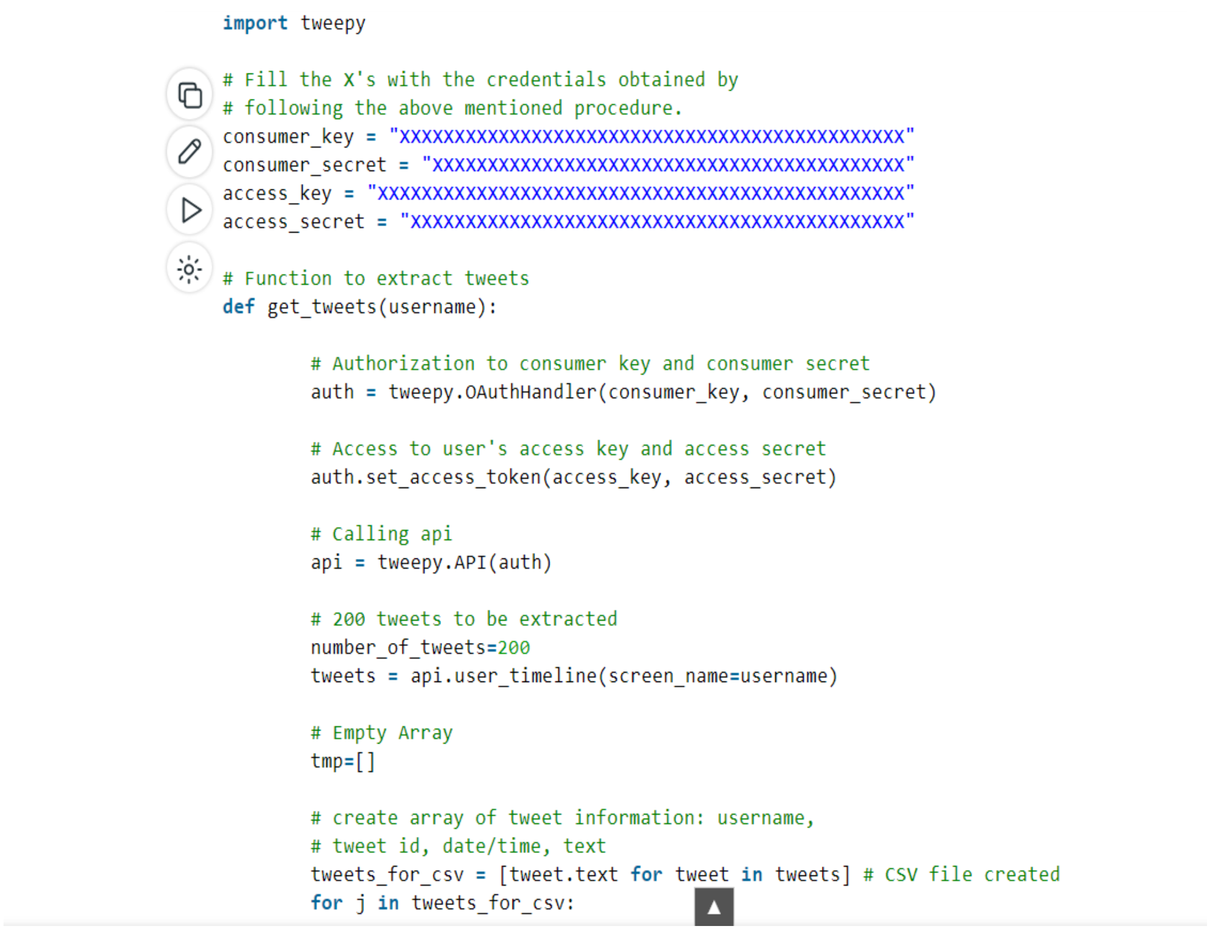
Extrication of Tweets using Tweepy

Twitter is a popular social network where users share messages called tweets. Twitter allows us to mine the data of any user using Twitter API or Tweepy. The data will be tweets extracted from the user. The first thing to do is get the consumer key, consumer secret, access key and access secret from twitter developer available easily for each user. These keys will help the API for authentication.

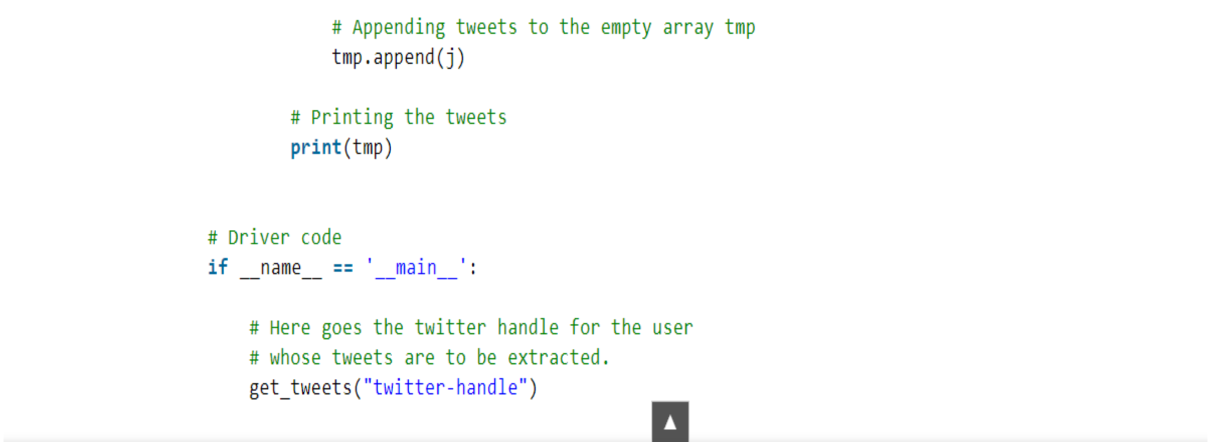
– Login to twitter developer section
– Go to "Create an App"
– Fill the details of the application.
– Click on Create your Twitter Application
– Details of your new app will be shown along with consumer key and consumer secret.
– For access token, click" Create my access token". The page will refresh and generate access token.

Tweepy is one of the library that should be installed using pip. Now in order to authorize our app to access Twitter on our behalf, we need to use the OAuth Interface. Tweepy provides the convenient Cursor interface to iterate through different types of objects. Twitter allows a maximum of 3200 tweets for extraction.

These all are the prerequisite that must be used before getting tweets of a user.



```
import tweepy

# Fill the X's with the credentials obtained by
# following the above mentioned procedure.
consumer_key = "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"
consumer_secret = "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"
access_key = "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"
access_secret = "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"

# Function to extract tweets
def get_tweets(username):

        # Authorization to consumer key and consumer secret
        auth = tweepy.OAuthHandler(consumer_key, consumer_secret)

        # Access to user's access key and access secret
        auth.set_access_token(access_key, access_secret)

        # Calling api
        api = tweepy.API(auth)

        # 200 tweets to be extracted
        number_of_tweets=200
        tweets = api.user_timeline(screen_name=username)

        # Empty Array
        tmp=[]

        # create array of tweet information: username,
        # tweet id, date/time, text
        tweets_for_csv = [tweet.text for tweet in tweets] # CSV file created
        for j in tweets_for_csv:
```



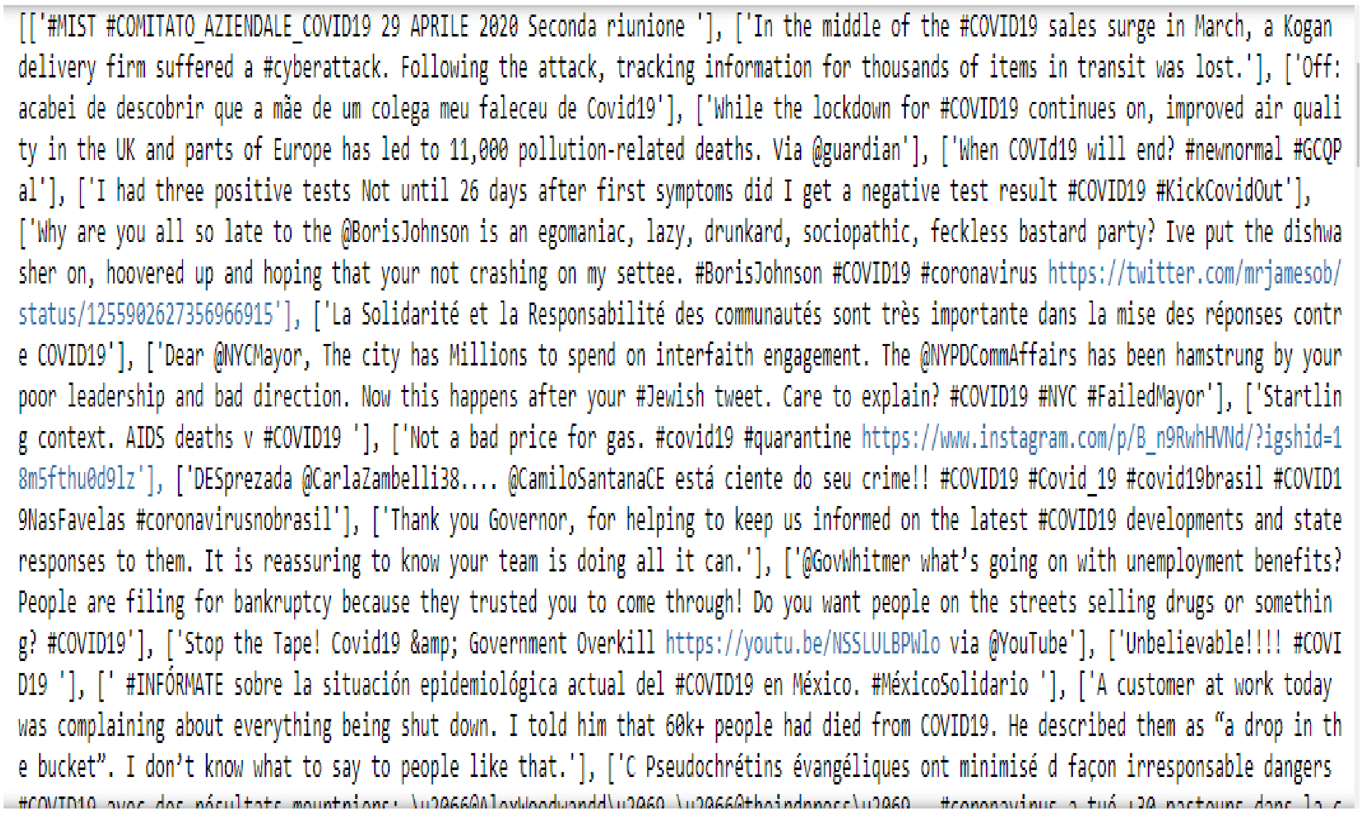
```
            # Appending tweets to the empty array tmp
            tmp.append(j)

        # Printing the tweets
        print(tmp)


# Driver code
if __name__ == '__main__':

    # Here goes the twitter handle for the user
    # whose tweets are to be extracted.
    get_tweets("twitter-handle")
```

Conclusion:

The above script would generate all the tweets of the user and would be appended to the empty array tmp. Here Tweepy is introduced as a tool to access Twitter data in a fairly easy way with Python. There are different types of data we can collect, with the obvious focus on the “tweet” object. Once we have collected some data, the possibilities in terms of analytics applications are endless.

One such application of extracting tweets is sentiment or emotion analysis. The emotion of the user can be obtained from the tweets by tokenizing each word and applying machine learning algorithms on that data. Such emotion or sentiment detection is used worldwide and will be broadly used in the future.

SAMPLE OF EXTRACTED DATA FROM TWITTER BY TWEEPY



```
[['#MIST #COMITATO_AZIENDALE_COVID19 29 APRILE 2020 Seconda riunione '], ['In the middle of the #COVID19 sales surge in March, a Kogan
delivery firm suffered a #cyberattack. Following the attack, tracking information for thousands of items in transit was lost.'], ['Off:
acabei de descobrir que a mãe de um colega meu faleceu de Covid19'], ['While the lockdown for #COVID19 continues on, improved air quali
ty in the UK and parts of Europe has led to 11,000 pollution-related deaths. Via @guardian'], ['When COVId19 will end? #newnormal #GCQP
al'], ['I had three positive tests Not until 26 days after first symptoms did I get a negative test result #COVID19 #KickCovidOut'],
['Why are you all so late to the @BorisJohnson is an egomaniac, lazy, drunkard, sociopathic, feckless bastard party? Ive put the dishwa
sher on, hoovered up and hoping that your not crashing on my settee. #BorisJohnson #COVID19 #coronavirus https://twitter.com/mrjamesob/
status/1255902627356966915'], ['La Solidarité et la Responsabilité des communautés sont très importante dans la mise des réponses contr
e COVID19'], ['Dear @NYCMayor, The city has Millions to spend on interfaith engagement. The @NYPDCommAffairs has been hamstrung by your
poor leadership and bad direction. Now this happens after your #Jewish tweet. Care to explain? #COVID19 #NYC #FailedMayor'], ['Startlin
g context. AIDS deaths v #COVID19 '], ['Not a bad price for gas. #covid19 #quarantine https://www.instagram.com/p/B_n9RwhHVNd/?igshid=1
8m5fthu0d9lz'], ['DESprezada @CarlaZambelli38.... @CamiloSantanaCE está ciente do seu crime!! #COVID19 #Covid_19 #covid19brasil #COVID1
9NasFavelas #coronavirusnobrasil'], ['Thank you Governor, for helping to keep us informed on the latest #COVID19 developments and state
responses to them. It is reassuring to know your team is doing all it can.'], ['@GovWhitmer what’s going on with unemployment benefits?
People are filing for bankruptcy because they trusted you to come through! Do you want people on the streets selling drugs or somethin
g? #COVID19'], ['Stop the Tape! Covid19 &amp; Government Overkill https://youtu.be/NSSLULBPWlo via @YouTube'], ['Unbelievable!!!! #COVI
D19 '], [' #INFÓRMATE sobre la situación epidemiológica actual del #COVID19 en México. #MéxicoSolidario '], ['A customer at work today
was complaining about everything being shut down. I told him that 60k+ people had died from COVID19. He described them as “a drop in th
e bucket”. I don’t know what to say to people like that.'], ['C Pseudochrétins évangéliques ont minimisé d façon irresponsable dangers
#COVID19 avec des résultats meurtriers: \u2066@AlexWoodwardd\u2069 \u2066@theindpress\u2069 . #coronavirus a tué +30 pasteurs dans la c
```

PRACTICAL 4

Understand and Create Core Components of Serverless framework

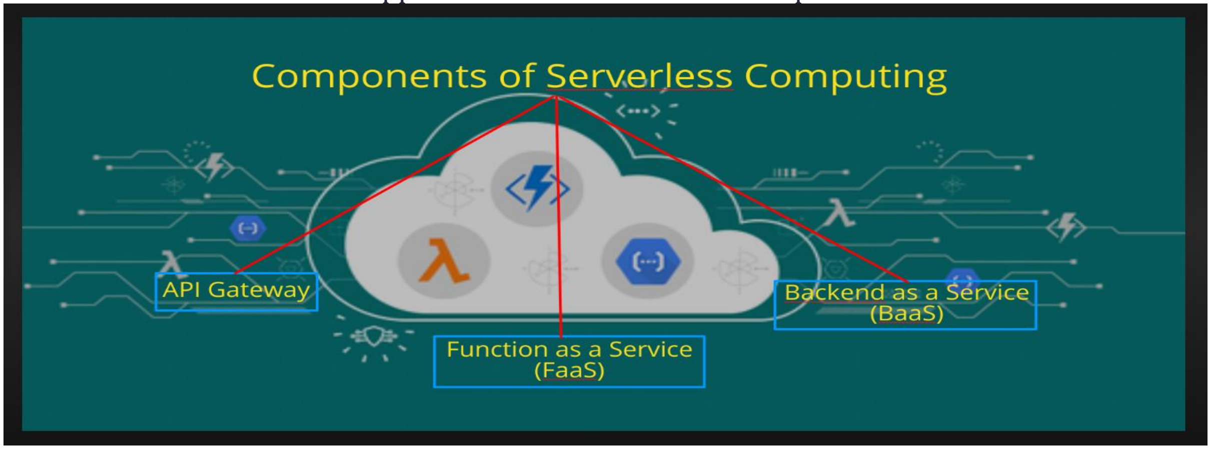
What is serverless?

Serverless is a cloud computing application development and execution model that enables developers to build and run application code without provisioning or managing servers or backend infrastructure.

Serverless lets developers put all their focus into writing the best front-end application code and business logic they can. All developers need to do is write their application code and deploy it to containers managed by a cloud service provider. The cloud provider handles the rest, provisioning the cloud infrastructure required to run the code and scaling the infrastructure up and down on demand as needed. The cloud provider is also responsible for all routine infrastructure management and maintenance such as operating system updates and patches, security management, capacity planning, system monitoring and more.

Core Components of Serverless Framework

- **Client application**: The UI of your application is best-rendered client side in JavaScript which allows you to use a simple, static web server.
- **Web server**: Amazon S3 provides a robust and simple web server. All of the static HTML, CSS and is files for your application can be served from S3.
- **FAAs solution**: It is the key enabler in serverless architecture. Some popular examples of FAAs are AWS Lambda, Google Cloud Functions, and Microsoft Azure Functions. AWS Lambda is used in this framework. The application services for logging in and accessing data will be built as Lambda functions. These functions will read and write from your database and provide JSON responses.
- **Security Token Service (STS)**: This will generate temporary AWS credentials (API key and secret key) for users of the application. These temporary credentials are used by the client application to invoke the AWS API (and thus invoke Lambda).
- **User authentication**: AWS Cognito is an identity service which is integrated with AWS Lambda. With Amazon Cognito, you can easily add user Sign-up and sign-in to your mobile and web apps. It also has the options to authenticate users through social identity providers such as Facebook, Twitter, or Amazon, with SAML identity solutions, or by using your own identity system.
- **Database**: AWS DynamoDB provides a fully managed NoSQL database. DynamoDB is not essential for a serverless application but is used as an example here.



Components of Serverless Computing
API Gateway
Function as a Service (FaaS)
Backend as a Service (BaaS)

Setting Up Serverless Framework With AWS

Get started with Serverless Framework's open-source CLI and AWS in minutes.

Installation

Install the `serverless` CLI via NPM:

```
npm install -g serverless
```

Note: If you don't already have Node on your machine, install it first. If you don't want to install Node or NPM, you can install `serverless` as a standalone binary.

Upgrade

You can upgrade the CLI later by running the same command: `npm install -g serverless`.

To upgrade to a specific major version, specify it like this: `npm install -g serverless@2`. If you installed `serverless` as a standalone binary, read this documentation instead.

Installation
Upgrade
Getting started
Deploying
Invoking function
Fetching function logs
Monitoring
Remove your service
What's next

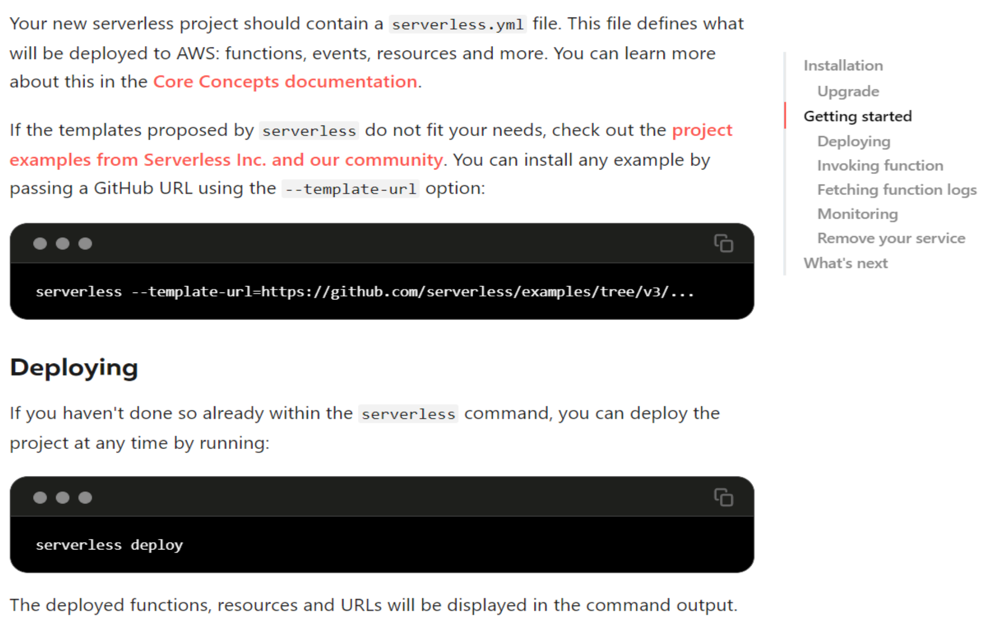
Getting started

To create your first project, run the command below and follow the prompts:

```
# Create a new serverless project
serverless

# Move into the newly created directory
cd your-service-name
```

The `serverless` command will guide you to:

1. create a new project
2. configure AWS credentials
3. optionally set up a free Serverless Dashboard account to monitor and troubleshoot your project



Your new serverless project should contain a `serverless.yml` file. This file defines what will be deployed to AWS: functions, events, resources and more. You can learn more about this in the Core Concepts documentation.

If the templates proposed by `serverless` do not fit your needs, check out the project examples from Serverless Inc. and our community. You can install any example by passing a GitHub URL using the `--template-url` option:

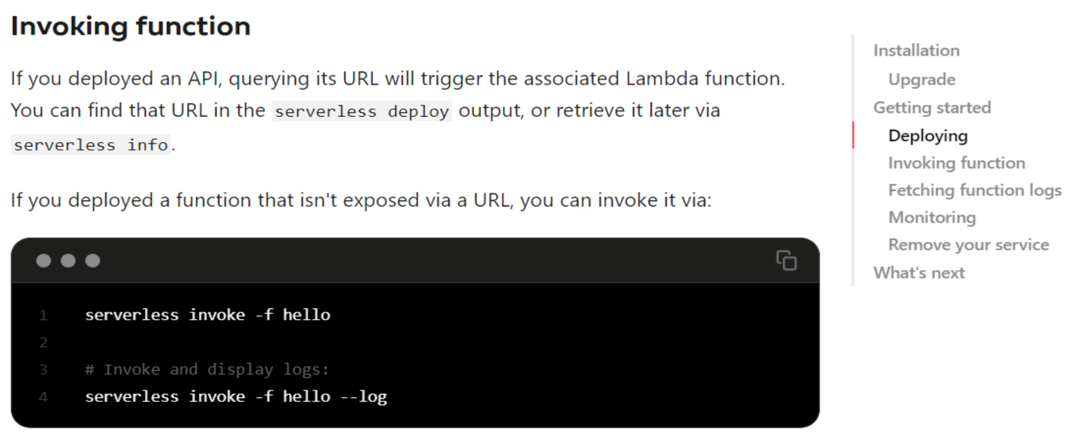
```
serverless --template-url=https://github.com/serverless/examples/tree/v3/...
```

Deploying

If you haven't done so already within the `serverless` command, you can deploy the project at any time by running:

```
serverless deploy
```

The deployed functions, resources and URLs will be displayed in the command output.

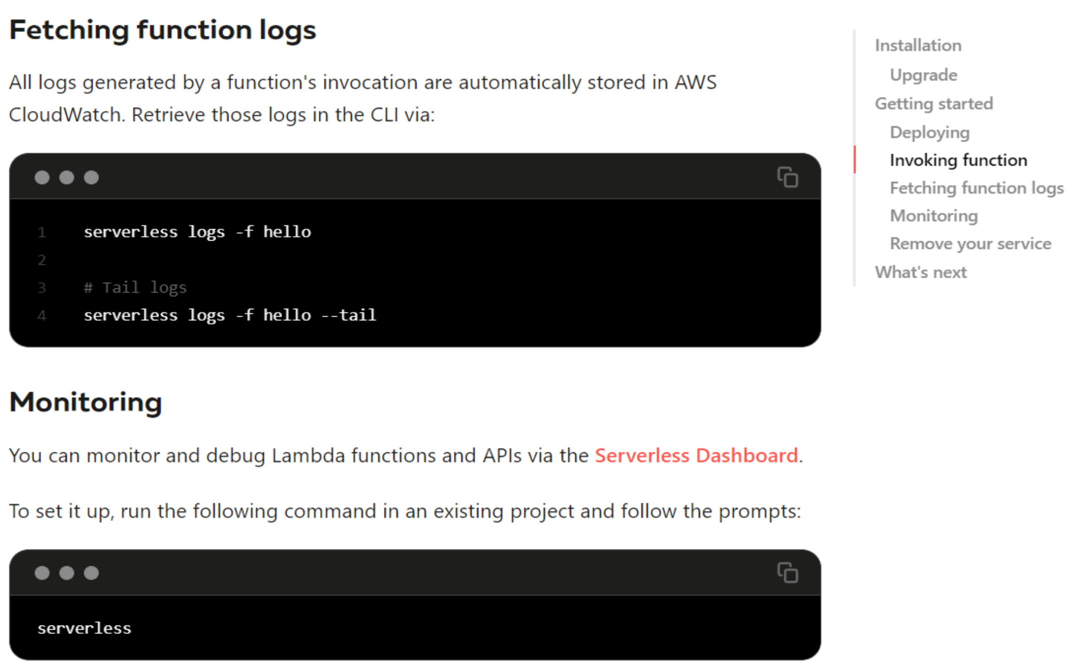


Invoking function

If you deployed an API, querying its URL will trigger the associated Lambda function. You can find that URL in the `serverless deploy` output, or retrieve it later via `serverless info`.

If you deployed a function that isn't exposed via a URL, you can invoke it via:

```
serverless invoke -f hello

# Invoke and display logs:
serverless invoke -f hello --log
```



Fetching function logs

All logs generated by a function's invocation are automatically stored in AWS CloudWatch. Retrieve those logs in the CLI via:

```
serverless logs -f hello

# Tail logs
serverless logs -f hello --tail
```

Monitoring

You can monitor and debug Lambda functions and APIs via the Serverless Dashboard.

To set it up, run the following command in an existing project and follow the prompts:

```
serverless
```

Remove your service

If you want to delete your service, run `serverless remove`. This will delete all the AWS resources created by your project and ensure that you don't incur any unexpected charges. It will also remove the service from Serverless Dashboard.

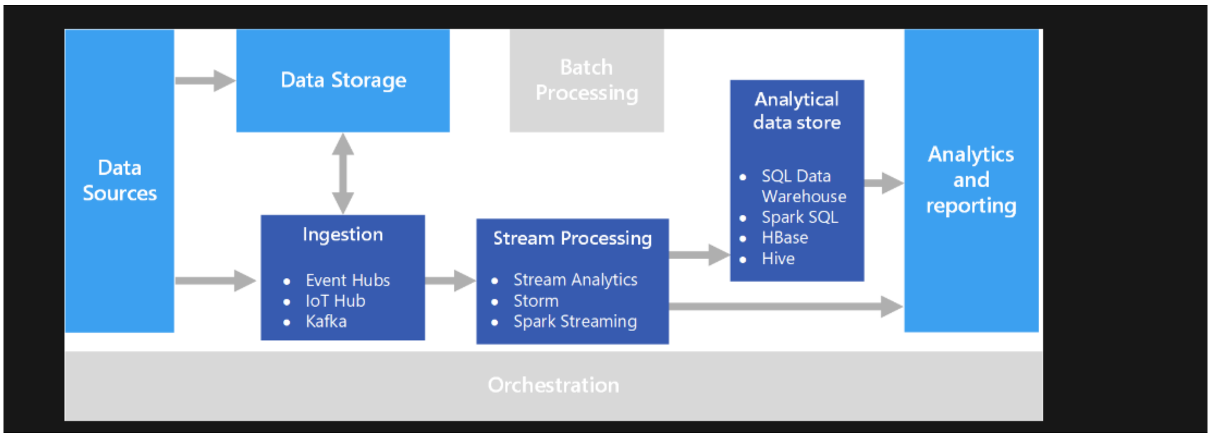
```
serverless remove
```

PRACTICAL 5

Application of event processing to real-time streaming data

Real-time processing is defined as the processing of unbounded stream of input data, with very short latency requirements for processing — measured in milliseconds or seconds. This incoming data typically arrives in an unstructured or semi-structured format, such as JSON, and has the same processing requirements as batch processing, but with shorter turnaround times to support real-time consumption.

Processed data is often written to an analytical data store, which is optimized for analytics and visualization. The processed data can also be ingested directly into the analytics and reporting layer for analysis, business intelligence, and real-time dashboard visualization.



Architecture

A real-time processing architecture has the following logical components.

1. **Real-time message ingestion**. The architecture must include a way to capture and store real-time messages to be consumed by a stream processing consumer. In simple cases, this service could be implemented as a simple data store in which new messages are deposited in a folder. But often the solution requires a message broker, such as Azure Event Hubs, that acts as a buffer for the messages. The message broker should support scale-out processing and reliable delivery.
2. **Stream processing**. After capturing real-time messages, the solution must process them by filtering, aggregating, and otherwise preparing the data for analysis.
3. **Analytical data store**. Many big data solutions are designed to prepare data for analysis and then serve the processed data in a structured format that can be queried using analytical tools.
4. **Analysis and reporting.** The goal of most big data solutions is to provide insights into the data through analysis and reporting.

Stream processing

1. **Azure Stream Analytics**. Azure Stream Analytics can run perpetual queries against an unbounded stream of data. These queries consume streams of data from storage or message brokers, filter and aggregate the data based on temporal windows, and write the results to sinks such as storage, databases, or directly to reports in Power BI. Stream

Analytics uses a SQL-based query language that supports temporal and geospatial constructs, and can be extended using JavaScript.

2. **Storm**. Apache Storm is an open-source framework for stream processing that uses a topology of spouts and bolts to consume, process, and output the results from real-time streaming data sources. You can provision Storm in an Azure HDInsight cluster, and implement a topology in Java or C#.
3. **Spark Streaming**. Apache Spark is an open-source distributed platform for general data processing. Spark provides the Spark Streaming API, in which you can write code in any supported Spark language, including Java, Scala, and Python. Spark 2.0 introduced the Spark Structured Streaming API, which provides a simpler and more consistent programming model. Spark 2.0 is available in an Azure HDInsight cluster.

Data storage

Azure Storage Blob Containers or Azure Data Lake Store. Incoming real-time data is usually captured in a message broker (see above), but in some scenarios, it can make sense to monitor a folder for new files and process them as they are created or updated. Additionally, many real-time processing solutions combine streaming data with static reference data, which can be stored in a file store. Finally, file storage may be used as an output destination for captured real-time data for archiving, or for further batch processing in a lambda architecture.

Applications of Real-time System:

Real-time System has applications in various fields of the technology. Here we will discuss the important applications of real-time system.

1. Industrial application:

Real-time system has a vast and prominent role in modern industries. Systems are made real time based so that maximum and accurate output can be obtained. In order to such things real -time systems are used in maximum industrial organizations. This system somehow led to the better performance and high productivity in less time. Some of the examples of industrial applications are: Automated Car Assembly Plant, Chemical Plant etc.

2. Medical Science application:

In the field of medical science, real-time system has a huge impact on the human health and treatment. Due to the introduction of real-time system in medical science, many lives are saved and treatment of complex diseases has been turned down to easier ways. People specially related to medical, now feel more relaxed due to these systems. Some of the examples of medical science applications are: Robot, MRI Scan, Radiation therapy etc.

3. Peripheral Equipment applications:

Real-time system has made the printing of large banners and such things very easier. Once these systems came into use, the technology world became stronger. Peripheral equipment is used for various purposes. These systems are embedded with microchips and perform accurately in order to get the desired response. Some of the examples of peripheral equipment applications are: Laser printer, fax machine, digital camera etc.

4. Telecommunication applications:

Real-time system map the world in such a way that it can be connected within a short time. Real-time systems have enabled the whole world to connect via a medium across internet. These systems make the people connect with each other in no time and feel the real environment of togetherness. Some examples of telecommunication applications of real-time systems are: Video Conferencing, Cellular system etc.

5. Defence applications:

In the new era of atomic world, defence can produce the missiles which have the dangerous powers and have the great destroying ability. All these systems are real-time system and it provides the system to attack and a system to defend. Some of the applications of defence using real time systems are: Missile guidance system, anti-missile system, Satellite missile system etc.

Real-time message ingestion

Azure Event Hubs. Azure Event Hubs is a messaging solution for ingesting millions of event messages per second. The captured event data can be processed by multiple consumers in parallel. While Event Hubs natively supports AMQP (Advanced Message Queuing Protocol 1.0), it also provides a binary compatibility layer that allows applications using the Kafka protocol (Kafka 1.0 and above) to process events using Event Hubs with no application changes.
Azure IoT Hub. Azure IoT Hub provides bi-directional communication between Internet-connected devices, and a scalable message queue that can handle millions of simultaneously connected devices.
Apache Kafka. Kafka is an open-source message queuing and stream processing application that can scale to handle millions of messages per second from multiple message producers, and route them to multiple consumers. Kafka is available in Azure as an HDInsight cluster type, with Azure Events for Kafka, and available via Confluent Cloud through our partnership with Confluent.

PRACTICAL 6

Write basic Python Programs for Variables, Datatypes, Numbers, Strings etc

INTRODUCTION OF PYTHON

Python is a popular programming language. It was created by Guido van Rossum, and released in 1991.

It is used for:

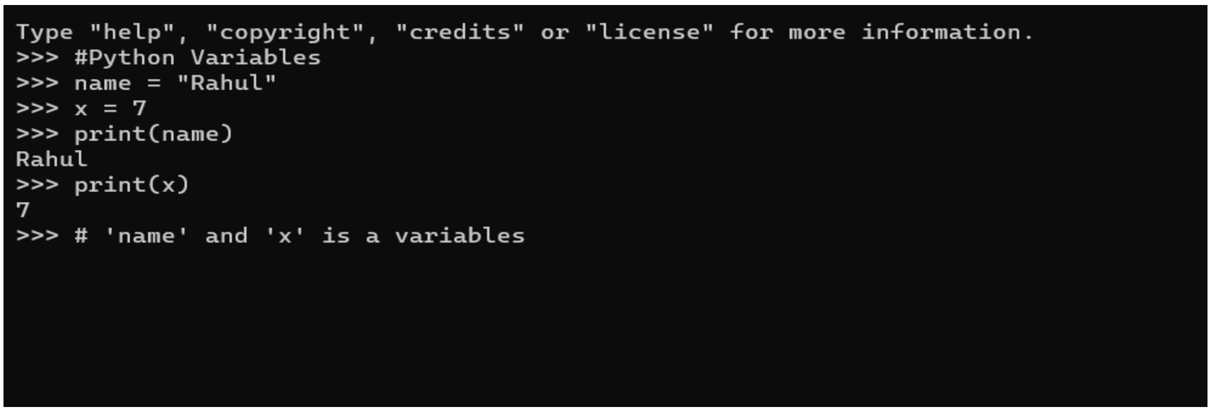
- Web development (server-side)
- Software development
- Mathematics
- System scripting

PYTHON VARIABLES

Variables are containers for storing data values.

Python has no command for declaring a variable.

A variable is created the moment you first assign a value to it.

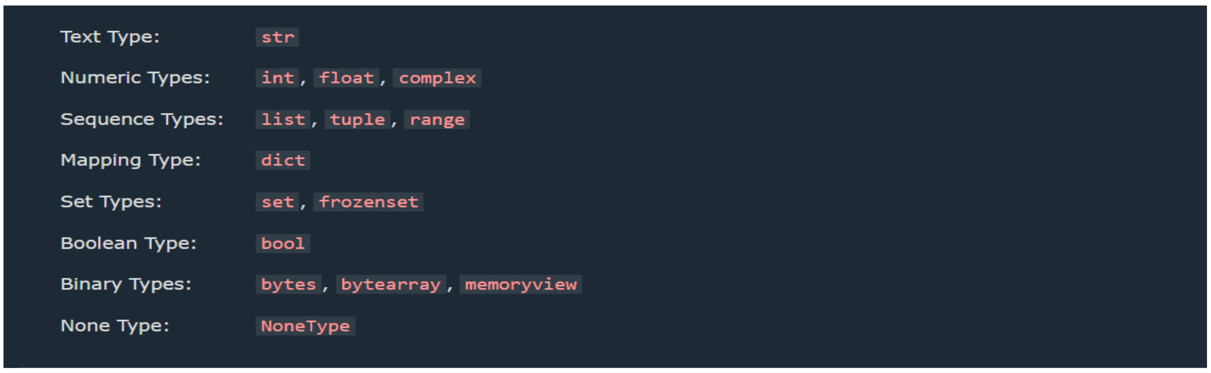


```
Type "help", "copyright", "credits" or "license" for more information.
>>> #Python Variables
>>> name = "Rahul"
>>> x = 7
>>> print(name)
Rahul
>>> print(x)
7
>>> # 'name' and 'x' is a variables
```

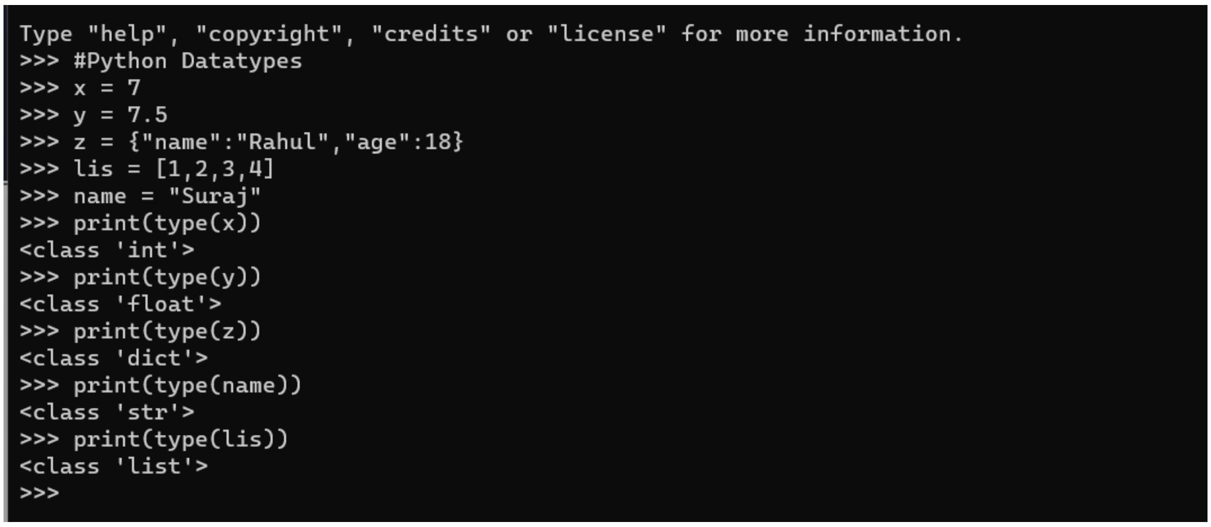
PYTHON DATATYPES

Variables can store data of different types, and different types can do different things.

Python has the following data types built-in by default, in these categories:



Text Type:	`str`
Numeric Types:	`int`, `float`, `complex`
Sequence Types:	`list`, `tuple`, `range`
Mapping Type:	`dict`
Set Types:	`set`, `frozenset`
Boolean Type:	`bool`
Binary Types:	`bytes`, `bytearray`, `memoryview`
None Type:	`NoneType`

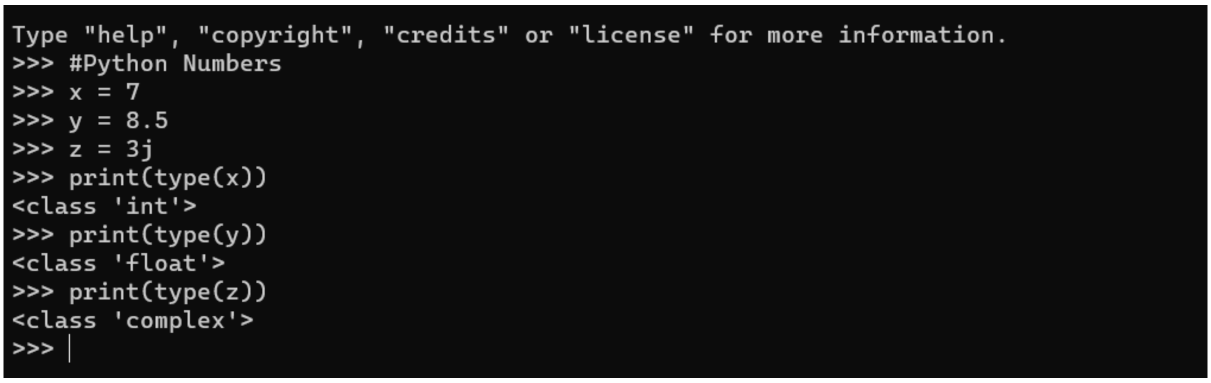


```
Type "help", "copyright", "credits" or "license" for more information.
>>> #Python Datatypes
>>> x = 7
>>> y = 7.5
>>> z = {"name":"Rahul","age":18}
>>> lis = [1,2,3,4]
>>> name = "Suraj"
>>> print(type(x))
<class 'int'>
>>> print(type(y))
<class 'float'>
>>> print(type(z))
<class 'dict'>
>>> print(type(name))
<class 'str'>
>>> print(type(lis))
<class 'list'>
>>>
```

PYTHON NUMBERS

There are three numeric types in Python.

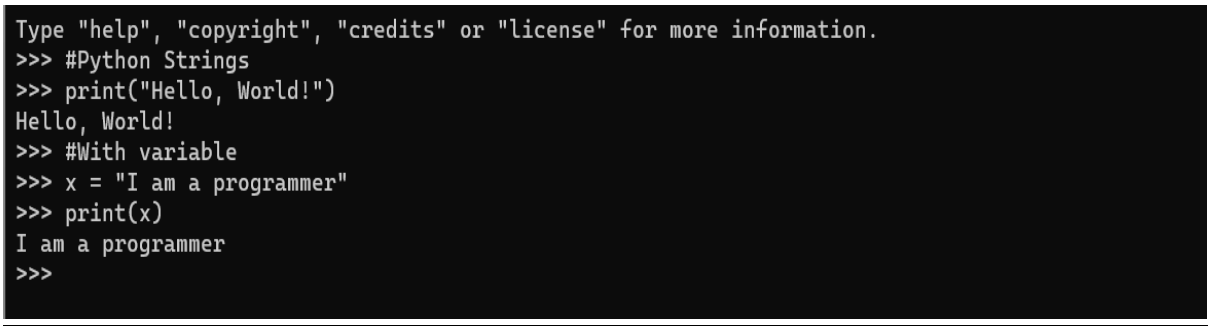
- Integer (int)
- Float
- Complex



```
Type "help", "copyright", "credits" or "license" for more information.
>>> #Python Numbers
>>> x = 7
>>> y = 8.5
>>> z = 3j
>>> print(type(x))
<class 'int'>
>>> print(type(y))
<class 'float'>
>>> print(type(z))
<class 'complex'>
>>> |
```

PYTHON STRINGS

Strings in python are surrounded by either single quotation marks, or double quotation marks.
‘hello’ is the same as “hello”.
You can display a string literal with the print() function:



```
Type "help", "copyright", "credits" or "license" for more information.
>>> #Python Strings
>>> print("Hello, World!")
Hello, World!
>>> #With variable
>>> x = "I am a programmer"
>>> print(x)
I am a programmer
>>>
```

PRACTICAL 7

Build and Deploy an Authenticated Microservices

What is Microservices Architecture?

Microservices architecture (often shortened to microservices) refers to an architectural style for developing applications. Microservices allow a large application to be separated into smaller independent parts, with each part having its own realm of responsibility. To serve a single user request, a microservices-based application can call on many internal microservices to compose its response.

What is microservices architecture used for?

Typically, microservices are used to speed up application development. Microservices architectures built using Java are common, especially Spring Boot ones. It is also common to compare microservices versus service-oriented architecture. Both have the same objective, which is to break up monolithic applications into smaller components, but they have different approaches. Here are some microservices architecture examples:

Website migration

A complex website that is hosted on a monolithic platform can be migrated to a cloud-based and container-based microservices platform.

Media content

Using microservices architecture, images and video assets can be stored in a scalable object storage system and served directly to web or mobile.

Transactions and invoices

Payment processing and ordering can be separated as independent units of services so payments continue to be accepted if invoicing is not working.

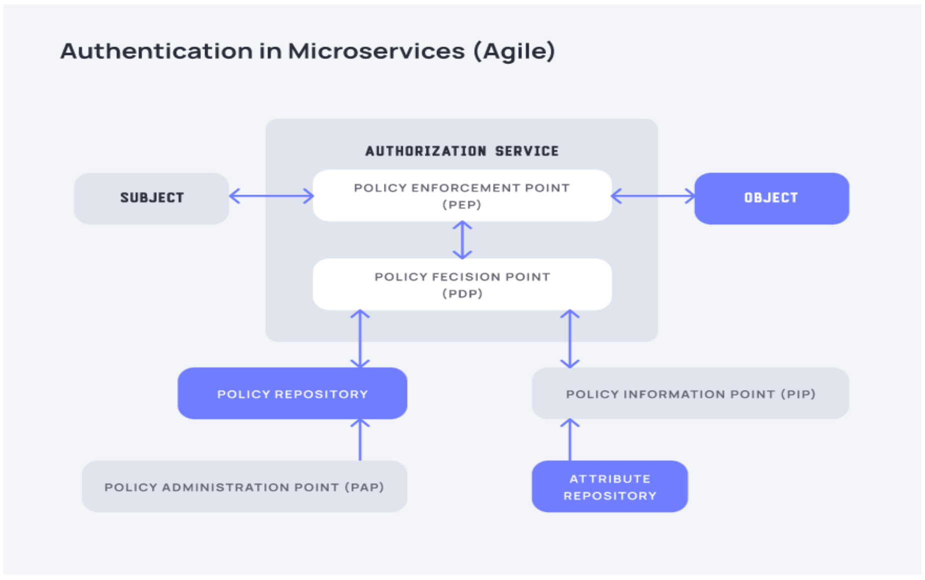
Data processing

A microservices platform can extend cloud support for existing modular data processing services.

How to build microservices

Step 1: Start with a monolith

The first best practice of microservices is that you probably do not need them. If you do not have any users for your application chances are that the business requirements are going to rapidly change while you are building your MVP. This is simply due to the nature of software development and the feedback cycle that needs to happen while you are identifying the key business capabilities that your system needs to provide. Microservices add exponential overhead and management complexity. For this reason, it is much less overhead for new projects to keep all the code and logic within a single codebase since it makes it easier to move the boundaries of the different modules of your application.


Authentication in Microservices (Agile)
AUTHORIZATION SERVICE
SUBJECT
POLICY ENFORCEMENT POINT (PEP)
OBJECT
POLICY FECISION POINT (PDP)
POLICY REPOSITORY
POLICY INFORMATION POINT (PIP)
POLICY ADMINISTRATION POINT (PAP)
ATTRIBUTE REPOSITORY

Step 2: Organize your teams the right way

Up until now, it might seem that building microservices is mostly a technical affair. You need to split a codebase into multiple services, implement the right patterns to fail gracefully and recover from network issues, deal with data consistency, monitor service load, etc. There will be numerous new concepts to grasp. But arguably the most but one thing that must not be ignored is that you'll need to restructure the way your teams are organized.

Step 3: Split the monolith to build a microservices architecture

When you have identified the boundaries of your services and you've figured out how to restructure your teams, you can start splitting your monolith to build microservices. The following are the key points to think about at that time.

- Keep communication between services simple with a RESTful AP
- Divide data into bounded contexts or data domains

- Build your microservices architecture for failure
- Emphasize monitoring to ease microservices testing
- Embrace continuous delivery to reduce deployment friction

Running microservices is not a sprint

Microservices are a popular and widely adopted industry best practice. For complex projects, they offer greater flexibility for building and deploying software. They also help identify and formalize the business components of your system, which comes in handy when you have several teams working on the same application. But there are also some clear drawbacks to managing distributed systems, and splitting a monolithic architecture should only be done when there's a clear understanding of the service boundaries.

Building microservices should be seen as a journey rather than the immediate goal for a team. Start small to understand the technical requirements of a distributed system, how to fail gracefully, and scale individual components. Then you can gradually extract more services as you gain experience and knowledge.

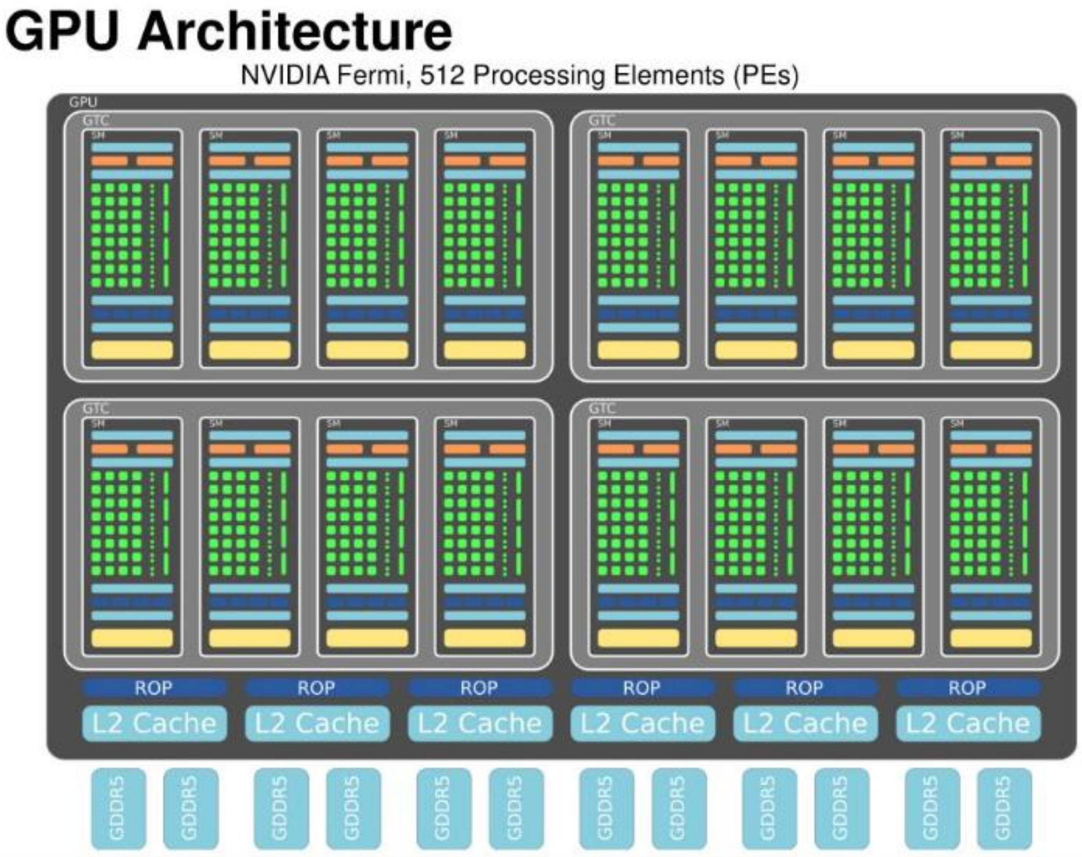
The migration to a microservices architecture does not need to be accomplished in one holistic effort. An iterative strategy to sequentially migrate smaller components to microservices is a safer bet. Identify the most well-defined service boundaries within an established monolith application and iteratively work to decouple them into their own microservice.

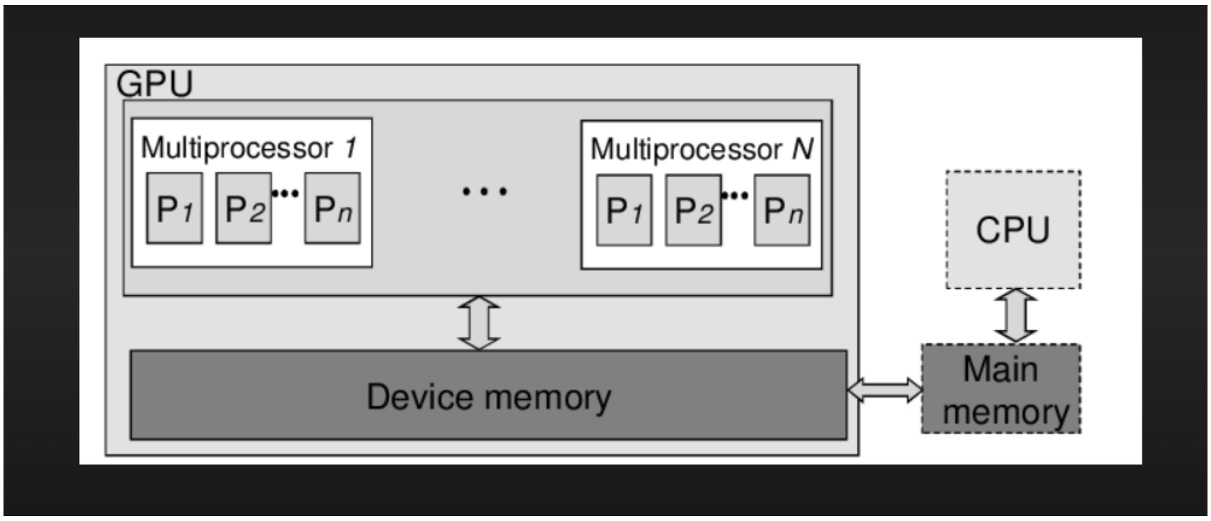
Conclusions

Microservices is a strategy that is beneficial to both the raw technical code development process and overall business organization strategy. Microservices help organize teams into units that focus on developing and owning specific business functions. This granular focus improves overall business communication and efficiency. There are trade-offs for the benefits of microservices. It is important that service boundaries are clearly defined before migrating to a microservice architecture.

PRACTICAL 8

Understand the core architectural components of GPU





What is GPU?

A **graphics processing unit** (**GPU**) is a specialized electronic circuit designed to manipulate and alter memory to accelerate the creation of images in a frame buffer intended for output to a display device. GPUs are used in embedded systems, mobile phones, personal computers, workstations, and game consoles.

What Does a GPU Do?

The graphics processing unit, or GPU, has become one of the most important types of computing technology, both for personal and business computing. Designed for parallel processing, the GPU is used in a wide range of applications, including graphics and video rendering. Although they're best known for their capabilities in gaming, GPUs are becoming more popular for use in creative production and artificial intelligence (AI).

GPUs were originally designed to accelerate the rendering of 3D graphics. Over time, they became more flexible and programmable, enhancing their capabilities. This allowed graphics programmers to create more interesting visual effects and realistic scenes with advanced lighting and shadowing techniques. Other developers also began to tap the power of GPUs to dramatically accelerate additional workloads in high performance computing (HPC), deep learning, and more.

What Are GPUs Used For?

Two decades ago, GPUs were used primarily to accelerate real-time 3D graphics applications, such as games. However, as the 21st century began, computer scientists realized that GPUs had the potential to solve some of the world's most difficult computing problems.

GPUs for Gaming

Video games have become more computationally intensive, with hyper realistic graphics and vast, complicated in-game worlds. With advanced display technologies, such as 4K screens and high refresh rates, along with the rise of virtual reality gaming, demands on graphics processing are growing fast. GPUs are capable of rendering graphics in both 2D and 3D. With better graphics performance, games can be played at higher resolution, at faster frame rates, or both.

GPUs for Video Editing and Content Creation

For years, video editors, graphic designers, and other creative professionals have struggled with long rendering times that tied up computing resources and stifled creative flow. Now, the parallel processing offered by GPUs makes it faster and easier to render video and graphics in higher-definition formats.

GPU for Machine Learning

Some of the most exciting applications for GPU technology involve AI and machine learning. Because GPUs incorporate an extraordinary amount of computational capability, they can deliver incredible acceleration in workloads that take advantage of the highly parallel nature of GPUs, such as image recognition. Many of today's deep learning technologies rely on GPUs working in conjunction with CPUs.

GPU system requirements (Linux, Windows only)

Hardware requirements

- Support for GPU processing is available on x86-64 systems and IBM® POWER® 8 systems that support NVIDIA Compute Unified Device Architecture (CUDA).
- GPU adapters must have a minimum compute capability of 3.0. To determine the compute capability of a device, see https://developer.nvidia.com/cuda-gpus. The CUDA toolkit

includes the sample programs *device Query* and *device Query Drv* that output the compute capability of devices in a system.

Software requirements

Your operating system must be supported by J9 and the CUDA Toolkit:

- GPU devices require the CUDA Toolkit 7.5, which provides a compiler, math libraries, and tools for debugging and tuning application performance. For a list of supported operating systems, and to download the toolkit, see https://developer.nvidia.com/cuda-toolkit.

Just-In-Time Compiler (JIT) based GPU support requires the NVIDIA Virtual Machine (NVVM) library, and the NVIDIA Management Library (NVML). NVVM is part of the CUDA Toolkit, and the runtime version of NVML is packaged with NVIDIA display drivers. You can download the latest GPU drivers from NVIDIA Driver Downloads.

You must include a path to the CUDA run time in the appropriate environment variable:

- Linux®: **LD_LIBRARY_PATH**
- Windows: **PATH**

What are the advantages of GPU?

1. Richer Gaming Experience.
2. Increase in Computer Performance.
3. Performance Increase in 3D Applications & Software's.
4. Better Video and HD Experience.
5. Better Driver Support.
6. Final Words & Advice.

PRACTICAL 9

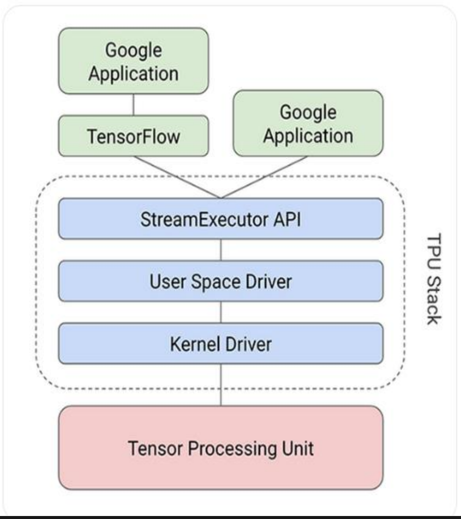
Understand core architectural components of TPU.

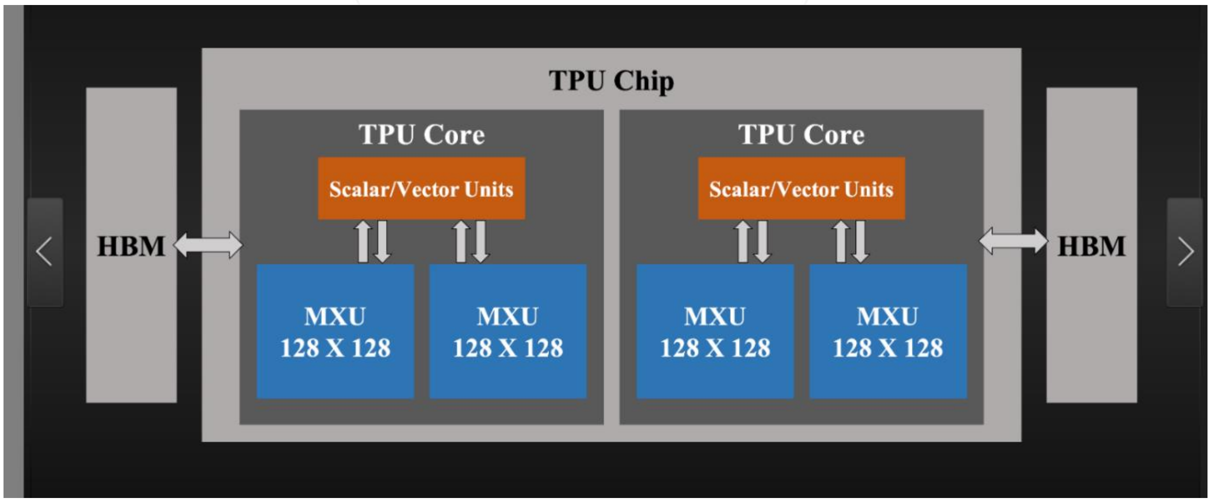
What is a Tensor Processing Unit?

With machine learning gaining its relevance and importance every day, the conventional microprocessors have proven to be unable to effectively handle it, be it training or neural network processing. GPUs, with their highly parallel architecture designed for fast graphic processing proved to be way more useful than CPUs for the purpose, but were somewhat lacking. Therefore, in order to combat this situation, Google developed an AI accelerator integrated circuit which would be used by its TensorFlow AI framework. This device has been named TPU (Tensor Processing Unit). The chip has been designed for TensorFlow.

The TPU includes the following computational resources:

- **Matrix Multiplier Unit (MXU)**: 65, 536 8-bit multiply-and-add units for matrix operations.
- **Unified Buffer (UB)**: 24MB of SRAM that work as registers
- **Activation Unit (AU):** Hardwired activation functions.





Advantages of TPU

The following are some notable advantages of TPUs:

1. Accelerates the performance of linear algebra computation, which is used heavily in machine learning applications.
2. Minimizes the time-to-accuracy when you train large, complex neural network models.
3. Models that previously took weeks to train on other hardware platforms can converge in hours on TPUs.

When to use a TPU

The following are the cases where TPUs are best suited in machine learning:

- Models dominated by matrix computations.
- Models with no custom TensorFlow operations inside the main training loop.
- Models that train for weeks or months
- Larger and very large models with very large effective batch sizes.

Cloud TPU programming model

Cloud TPUs are very fast at performing dense vector and matrix computations. Transferring data between Cloud TPU and host memory is slow compared to the speed of computation—the speed of the PCIe bus is *much* slower than both the Cloud TPU interconnect and the on-chip high bandwidth memory (HBM). Partial compilation of a model, where execution passes back and forth between host and device causes the TPU to be idle most of the time, waiting for data to arrive over the PCIe bus. To alleviate this situation, the programming model for Cloud TPU is designed to execute much of the training on the TPU—ideally the entire training loop.

Following are some salient features of the TPU programming model:

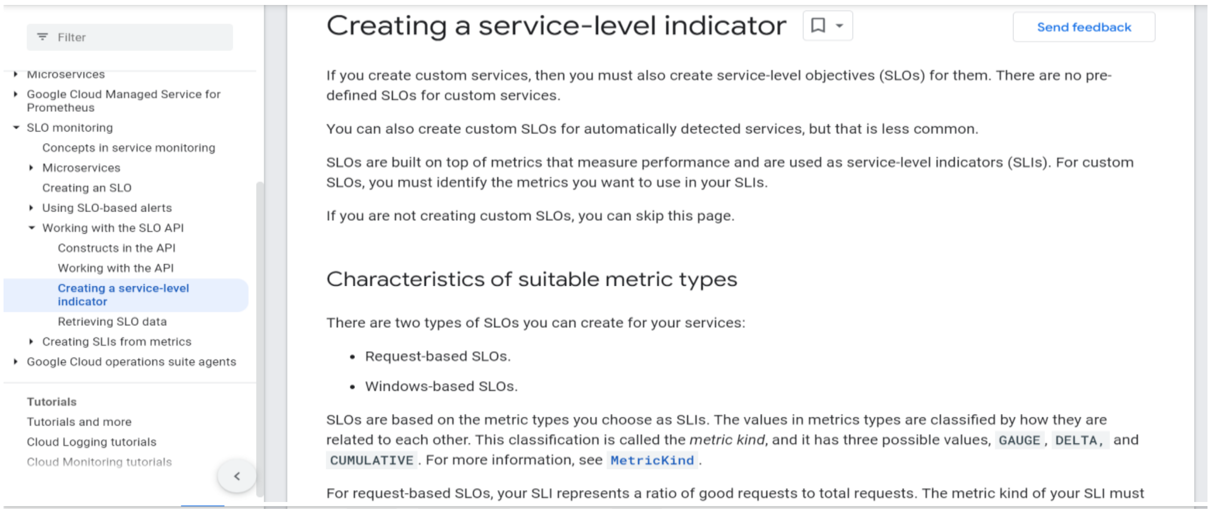
- All model parameters are kept in on-chip high bandwidth memory.
- The cost of launching computations on Cloud TPU is amortized by executing many training steps in a loop.
- Input training data is streamed to an "infeed" queue on the Cloud TPU. A program running on Cloud TPU retrieves batches from these queues during each training step.
- The TensorFlow server running on the host machine (the CPU attached to the Cloud TPU device) fetches data and pre-processes it before "in feeding" to the Cloud TPU hardware.
- Data parallelism: Cores on a Cloud TPU execute an identical program residing in their own respective HBM in a synchronous manner. A reduction operation is performed at the end of each neural network step across all the cores.

PRACTICAL 10

Develop service level indicator and service level Objective compliance

What is Service Level Indicator?

A service level indicator (SLI) is a measure of the service level provided by a service provider to a customer. SLIs form the basis of service level objectives (SLOs), which in turn form the basis of service level agreements (SLAs); an SLI is thus also called an SLA metric.



Creating a service-level indicator

If you create custom services, then you must also create service-level objectives (SLOs) for them. There are no pre-defined SLOs for custom services.

You can also create custom SLOs for automatically detected services, but that is less common.

SLOs are built on top of metrics that measure performance and are used as service-level indicators (SLIs). For custom SLOs, you must identify the metrics you want to use in your SLIs.

If you are not creating custom SLOs, you can skip this page.

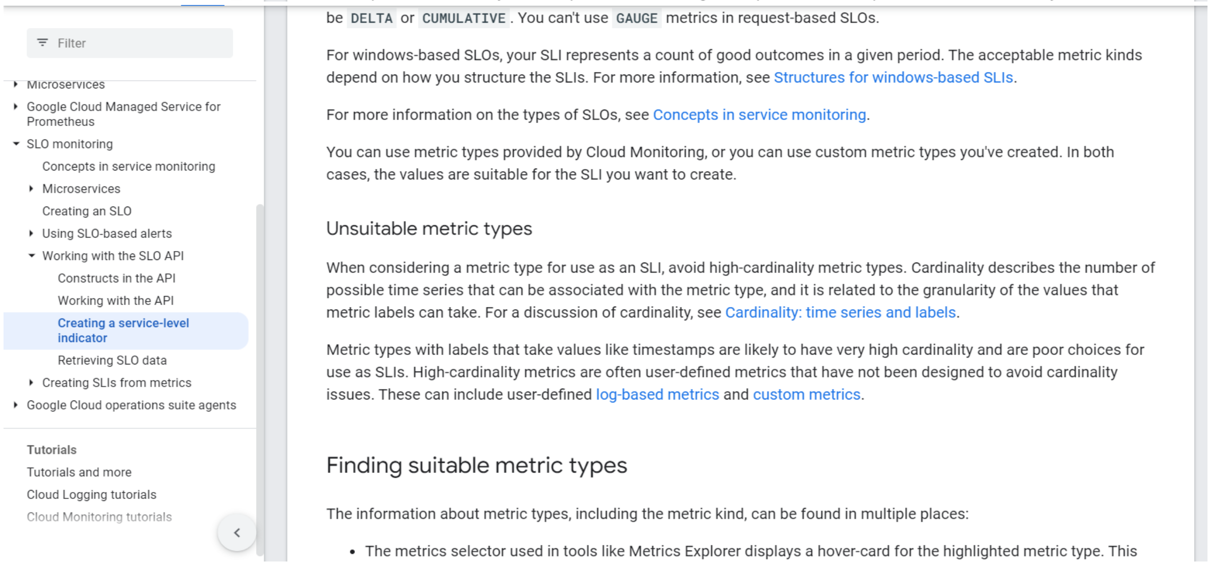
Characteristics of suitable metric types

There are two types of SLOs you can create for your services:

- Request-based SLOs.
- Windows-based SLOs.

SLOs are based on the metric types you choose as SLIs. The values in metrics types are classified by how they are related to each other. This classification is called the *metric kind*, and it has three possible values, `GAUGE`, `DELTA`, and `CUMULATIVE`. For more information, see `MetricKind`.

For request-based SLOs, your SLI represents a ratio of good requests to total requests. The metric kind of your SLI must



be `DELTA` or `CUMULATIVE`. You can't use `GAUGE` metrics in request-based SLOs.

For windows-based SLOs, your SLI represents a count of good outcomes in a given period. The acceptable metric kinds depend on how you structure the SLIs. For more information, see Structures for windows-based SLIs.

For more information on the types of SLOs, see Concepts in service monitoring.

You can use metric types provided by Cloud Monitoring, or you can use custom metric types you've created. In both cases, the values are suitable for the SLI you want to create.

Unsuitable metric types

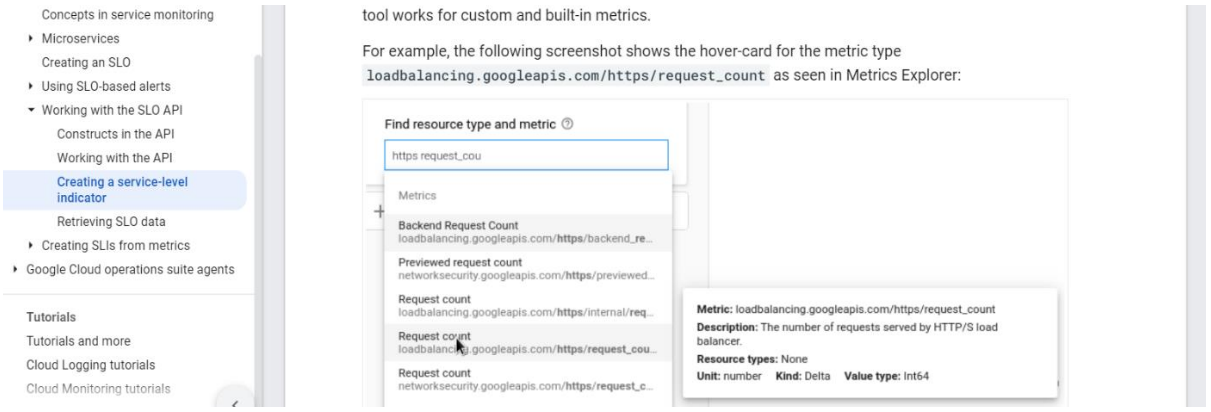
When considering a metric type for use as an SLI, avoid high-cardinality metric types. Cardinality describes the number of possible time series that can be associated with the metric type, and it is related to the granularity of the values that metric labels can take. For a discussion of cardinality, see Cardinality: time series and labels.

Metric types with labels that take values like timestamps are likely to have very high cardinality and are poor choices for use as SLIs. High-cardinality metrics are often user-defined metrics that have not been designed to avoid cardinality issues. These can include user-defined log-based metrics and custom metrics.

Finding suitable metric types

The information about metric types, including the metric kind, can be found in multiple places:

- The metrics selector used in tools like Metrics Explorer displays a hover-card for the highlighted metric type. This



tool works for custom and built-in metrics.

For example, the following screenshot shows the hover-card for the metric type `loadbalancing.googleapis.com/https/request_count` as seen in Metrics Explorer:

https request_cou

Backend Request Count
loadbalancing.googleapis.com/**https**/backend_**re**...

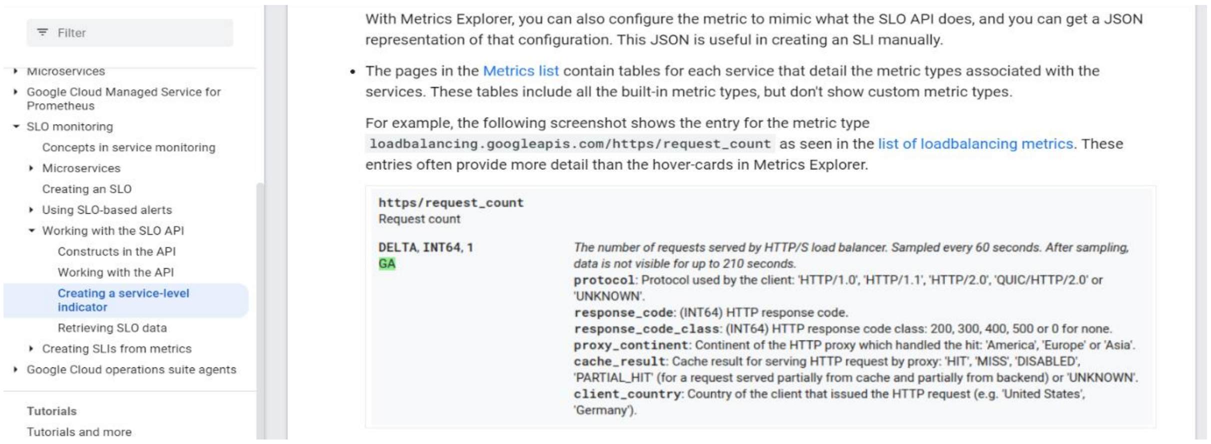
Previewed request count
networksecurity.googleapis.com/**https**/previewed...

Request count
loadbalancing.googleapis.com/**https**/internal/**req**...

Request count
loadbalancing.googleapis.com/**https/request_cou**...

Request count
networksecurity.googleapis.com/**https/request_c**...

Metric: loadbalancing.googleapis.com/https/request_count
Description: The number of requests served by HTTP/S load balancer.
Resource types: None
Unit: number **Kind:** Delta **Value type:** Int64



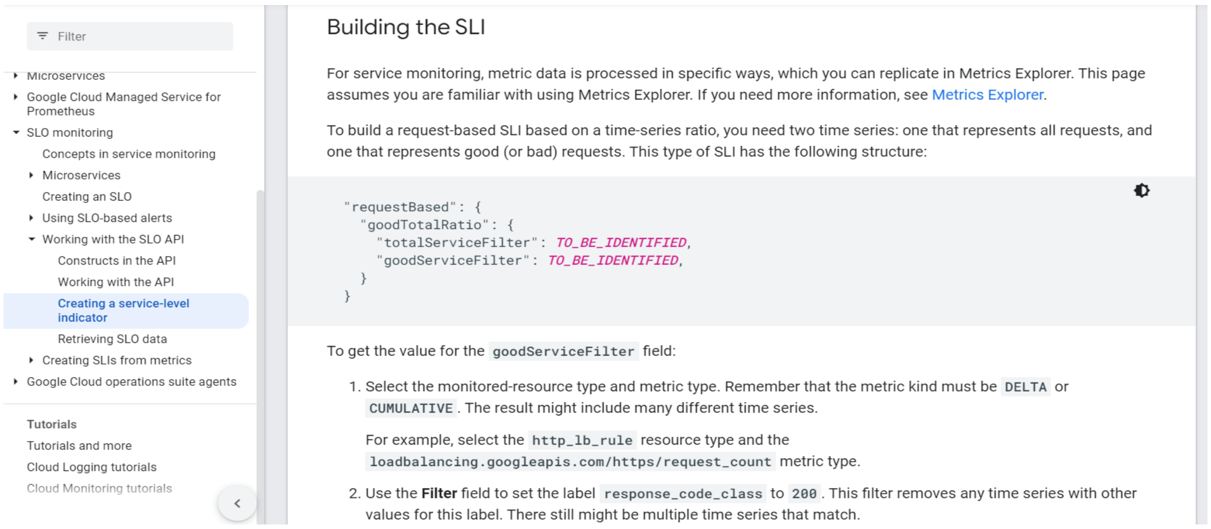
With Metrics Explorer, you can also configure the metric to mimic what the SLO API does, and you can get a JSON representation of that configuration. This JSON is useful in creating an SLI manually.

- The pages in the Metrics list contain tables for each service that detail the metric types associated with the services. These tables include all the built-in metric types, but don't show custom metric types.

  For example, the following screenshot shows the entry for the metric type `loadbalancing.googleapis.com/https/request_count` as seen in the list of loadbalancing metrics. These entries often provide more detail than the hover-cards in Metrics Explorer.

`https/request_count`
Request count

DELTA, INT64, 1 GA	*The number of requests served by HTTP/S load balancer. Sampled every 60 seconds. After sampling, data is not visible for up to 210 seconds.* **`protocol`**: Protocol used by the client: 'HTTP/1.0', 'HTTP/1.1', 'HTTP/2.0', 'QUIC/HTTP/2.0' or 'UNKNOWN'. **`response_code`**: (INT64) HTTP response code. **`response_code_class`**: (INT64) HTTP response code class: 200, 300, 400, 500 or 0 for none. **`proxy_continent`**: Continent of the HTTP proxy which handled the hit: 'America', 'Europe' or 'Asia'. **`cache_result`**: Cache result for serving HTTP request by proxy: 'HIT', 'MISS', 'DISABLED', 'PARTIAL_HIT' (for a request served partially from cache and partially from backend) or 'UNKNOWN'. **`client_country`**: Country of the client that issued the HTTP request (e.g. 'United States', 'Germany').


Filter
- Microservices
- Google Cloud Managed Service for Prometheus
- SLO monitoring
  - Concepts in service monitoring
  - Microservices
  - Creating an SLO
  - Using SLO-based alerts
  - Working with the SLO API
    - Constructs in the API
    - Working with the API
    - Creating a service-level indicator
    - Retrieving SLO data
  - Creating SLIs from metrics
- Google Cloud operations suite agents

Tutorials
Tutorials and more
Cloud Logging tutorials
Cloud Monitoring tutorials

Building the SLI

For service monitoring, metric data is processed in specific ways, which you can replicate in Metrics Explorer. This page assumes you are familiar with using Metrics Explorer. If you need more information, see Metrics Explorer.

To build a request-based SLI based on a time-series ratio, you need two time series: one that represents all requests, and one that represents good (or bad) requests. This type of SLI has the following structure:

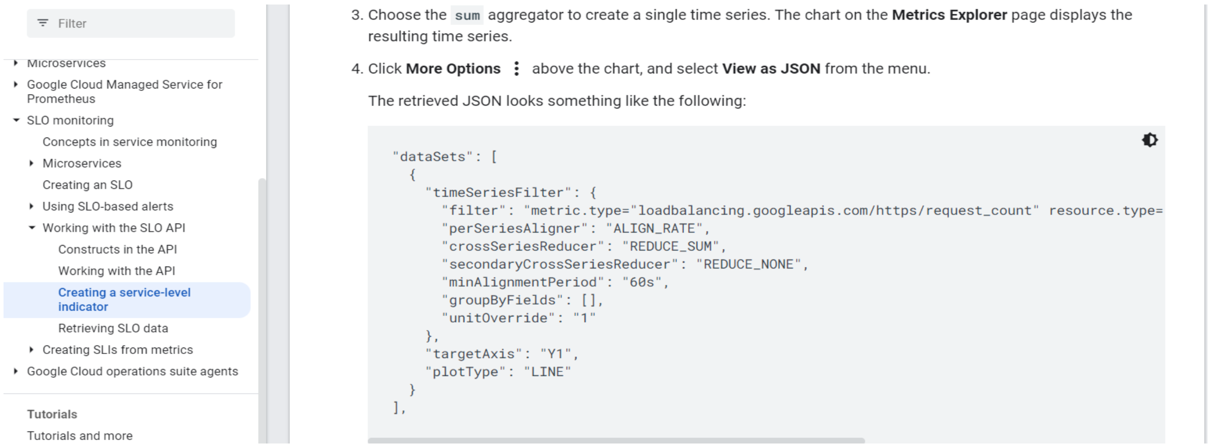
```
"requestBased": {
  "goodTotalRatio": {
    "totalServiceFilter": TO_BE_IDENTIFIED,
    "goodServiceFilter": TO_BE_IDENTIFIED,
  }
}
```

To get the value for the `goodServiceFilter` field:

1. Select the monitored-resource type and metric type. Remember that the metric kind must be `DELTA` or `CUMULATIVE`. The result might include many different time series.

   For example, select the `http_lb_rule` resource type and the `loadbalancing.googleapis.com/https/request_count` metric type.

2. Use the **Filter** field to set the label `response_code_class` to `200`. This filter removes any time series with other values for this label. There still might be multiple time series that match.


Filter
- Microservices
- Google Cloud Managed Service for Prometheus
- SLO monitoring
  - Concepts in service monitoring
  - Microservices
  - Creating an SLO
  - Using SLO-based alerts
  - Working with the SLO API
    - Constructs in the API
    - Working with the API
    - Creating a service-level indicator
    - Retrieving SLO data
  - Creating SLIs from metrics
- Google Cloud operations suite agents

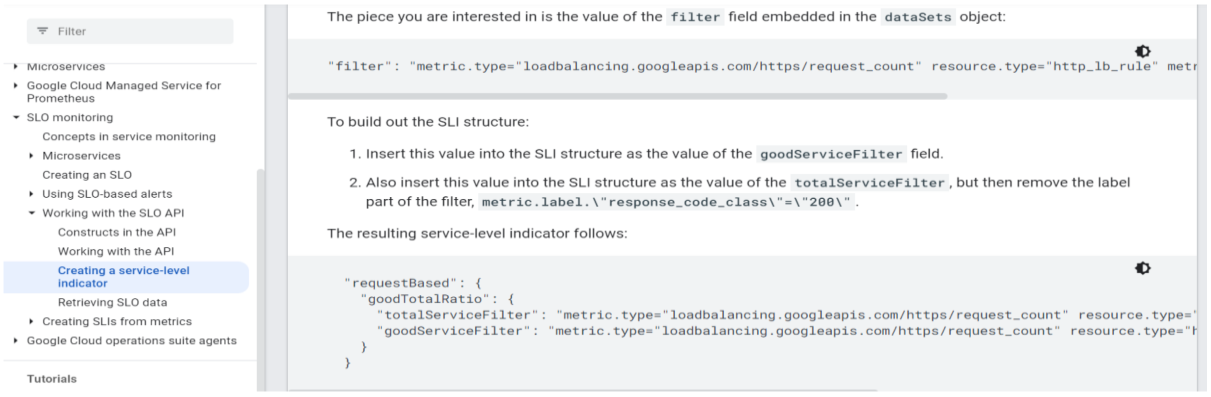
Tutorials
Tutorials and more

3. Choose the `sum` aggregator to create a single time series. The chart on the **Metrics Explorer** page displays the resulting time series.

4. Click **More Options** ⋮ above the chart, and select **View as JSON** from the menu.

   The retrieved JSON looks something like the following:

```
"dataSets": [
  {
    "timeSeriesFilter": {
      "filter": "metric.type="loadbalancing.googleapis.com/https/request_count" resource.type=
      "perSeriesAligner": "ALIGN_RATE",
      "crossSeriesReducer": "REDUCE_SUM",
      "secondaryCrossSeriesReducer": "REDUCE_NONE",
      "minAlignmentPeriod": "60s",
      "groupByFields": [],
      "unitOverride": "1"
    },
    "targetAxis": "Y1",
    "plotType": "LINE"
  }
],
```


Filter
- Microservices
- Google Cloud Managed Service for Prometheus
- SLO monitoring
  - Concepts in service monitoring
  - Microservices
  - Creating an SLO
  - Using SLO-based alerts
  - Working with the SLO API
    - Constructs in the API
    - Working with the API
    - Creating a service-level indicator
    - Retrieving SLO data
  - Creating SLIs from metrics
- Google Cloud operations suite agents

Tutorials

The piece you are interested in is the value of the `filter` field embedded in the `dataSets` object:

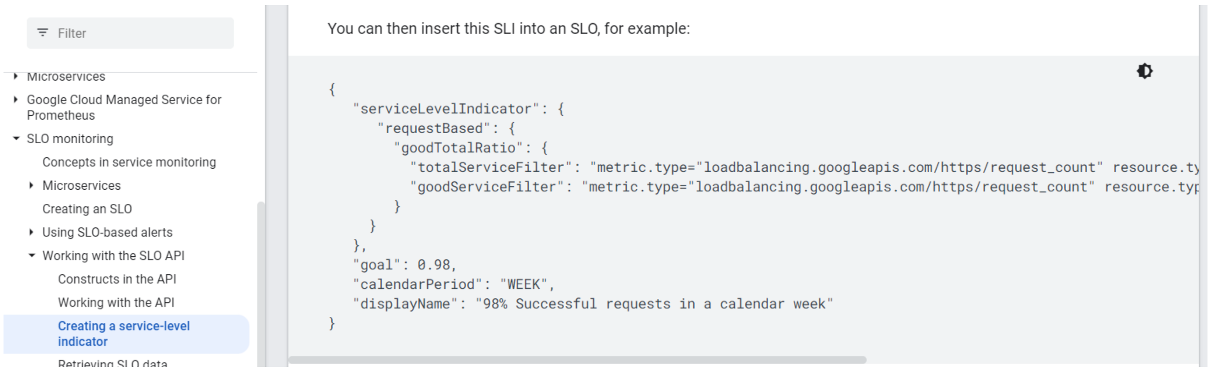
```
"filter": "metric.type="loadbalancing.googleapis.com/https/request_count" resource.type="http_lb_rule" metr
```

To build out the SLI structure:

1. Insert this value into the SLI structure as the value of the `goodServiceFilter` field.
2. Also insert this value into the SLI structure as the value of the `totalServiceFilter`, but then remove the label part of the filter, `metric.label.\"response_code_class\"=\"200\"`.

The resulting service-level indicator follows:

```
"requestBased": {
  "goodTotalRatio": {
    "totalServiceFilter": "metric.type="loadbalancing.googleapis.com/https/request_count" resource.type='
    "goodServiceFilter": "metric.type="loadbalancing.googleapis.com/https/request_count" resource.type="h
  }
}
```

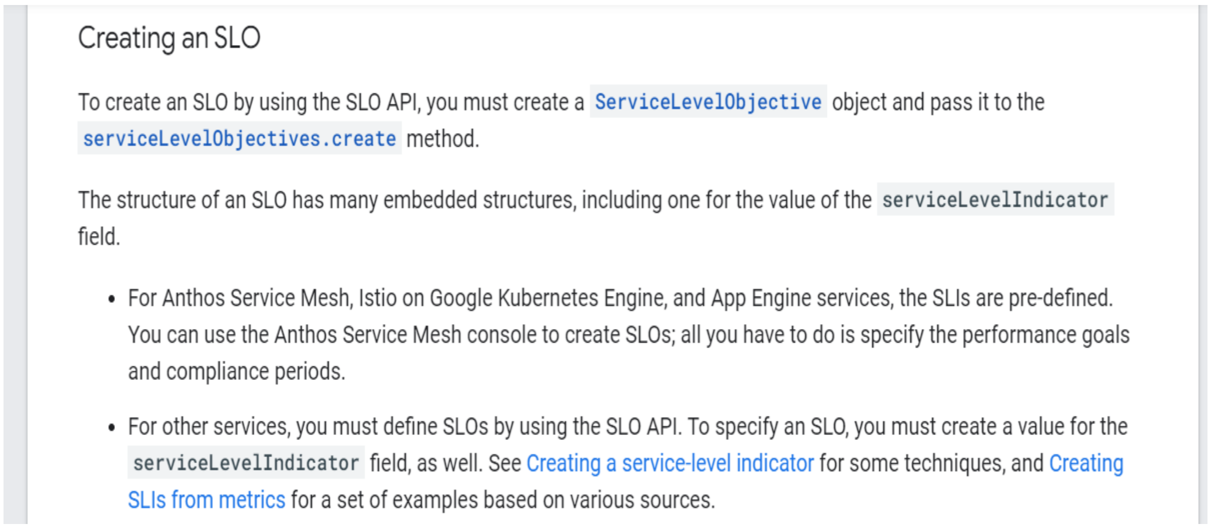


You can then insert this SLI into an SLO, for example:

```
{
   "serviceLevelIndicator": {
      "requestBased": {
        "goodTotalRatio": {
          "totalServiceFilter": "metric.type="loadbalancing.googleapis.com/https/request_count" resource.ty
          "goodServiceFilter": "metric.type="loadbalancing.googleapis.com/https/request_count" resource.typ
        }
      }
   },
   "goal": 0.98,
   "calendarPeriod": "WEEK",
   "displayName": "98% Successful requests in a calendar week"
}
```

What are Service Level Objectives?

A Service Level Objective (SLO) serves as a benchmark for indicators, parameters, or metrics defined with specific service level targets. The objectives may be an optimal range or a specific value for each service function or process that constitutes a cloud service.

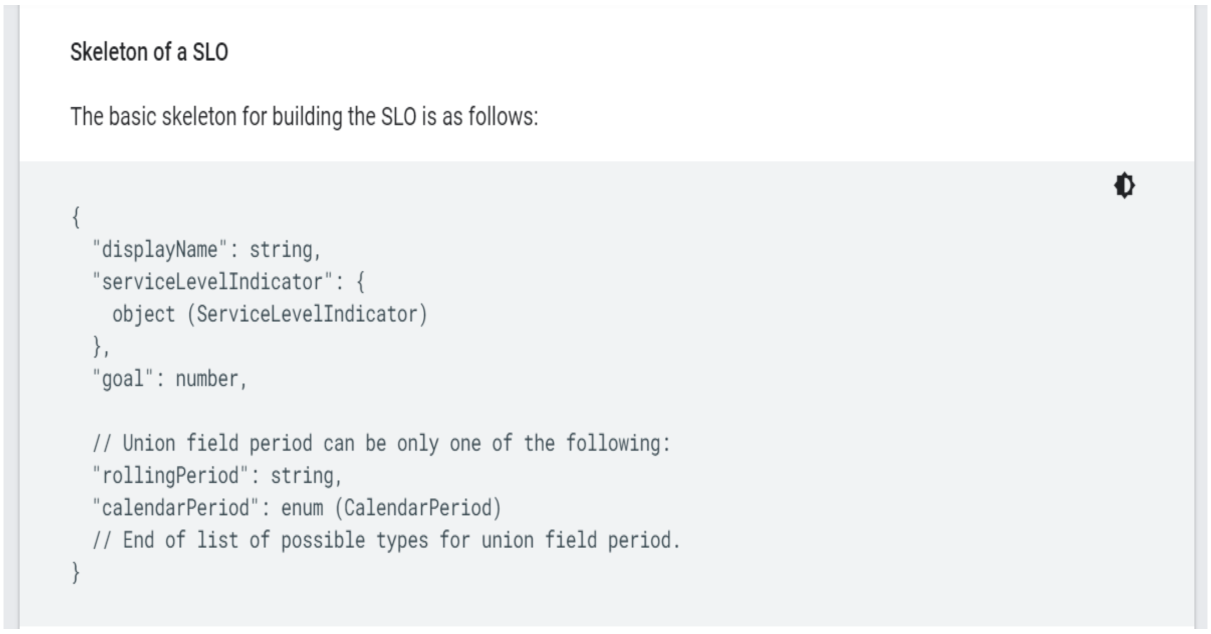


Creating an SLO

To create an SLO by using the SLO API, you must create a `ServiceLevelObjective` object and pass it to the `serviceLevelObjectives.create` method.

The structure of an SLO has many embedded structures, including one for the value of the `serviceLevelIndicator` field.

- For Anthos Service Mesh, Istio on Google Kubernetes Engine, and App Engine services, the SLIs are pre-defined. You can use the Anthos Service Mesh console to create SLOs; all you have to do is specify the performance goals and compliance periods.
- For other services, you must define SLOs by using the SLO API. To specify an SLO, you must create a value for the `serviceLevelIndicator` field, as well. See Creating a service-level indicator for some techniques, and Creating SLIs from metrics for a set of examples based on various sources.



Skeleton of a SLO

The basic skeleton for building the SLO is as follows:

```
{
  "displayName": string,
  "serviceLevelIndicator": {
    object (ServiceLevelIndicator)
  },
  "goal": number,

  // Union field period can be only one of the following:
  "rollingPeriod": string,
  "calendarPeriod": enum (CalendarPeriod)
  // End of list of possible types for union field period.
}
```

You must specify the following:

- Display name: a description of the SLO
- A service-level indicator, which is one of the three types:
  - `BasicSli`
  - `RequestBasedSli`
  - `WindowsBasedSli`
- The performance goal (a percentage)
- The compliance period, which one of two types:
  - A rolling period of some length (in seconds)
  - A calendar period

For more information about SLIs, performance goals, and compliance periods, see Concepts in service monitoring.

For Anthos Service Mesh, Istio on Google Kubernetes Engine, and App Engine services, the SLI type is the basic SLI.

For other services, you have to create a request-based SLI or a windows-based SLI. See Creating a service-level indicator for some techniques.

Example

After you have the SLI, you can build the SLO. The following example defines an SLO for a service that uses a basic SLI. You might have several SLOs on a single SLI, for example, one for 95% availability and one for 99% availability. The following example is an SLO for 95% availability over a calendar week:

```
{
  "serviceLevelIndicator": {
    "basicSli": {
      "availability": {},
      "location": [
        "us-central1-c"
      ]
    }
  },
  "goal": 0.95,
  "calendarPeriod": "WEEK",
  "displayName": "95% Availability in Calendar Week"
}
```

This example specifies an SLO for 75% availability over a rolling 3-day period:

```
{
  "serviceLevelIndicator": {
    "basicSli": {
      "availability": {},
      "location": [
        "us-central1-c"
      ]
    }
  },
  "goal": 0.75,
  "rollingPeriod": "259200s",
  "displayName": "75% Availability in Rolling 3 Days"
}
```

Protocol

To create an SLO by using `curl`, send a `POST` message to the `https://monitoring.googleapis.com/v3/projects/[PROJECT_ID]/services/SERVICE_ID/serviceLevelObjectives` endpoint:

1. Create a variable for the service ID:

```
SERVICE_ID=custom:my-test-service
```

2. Create a variable to hold the request body, an instance of the `ServiceLevelObjective` object. This example uses one of the SLOs described previously:

```
CREATE_SLO_POST_BODY=$(cat <<EOF
{
  "serviceLevelIndicator": {
    "basicSli": {
      "availability": {},
      "location": [
        "us-central1-c"
      ]
    }
  },
  "goal": 0.75,
  "rollingPeriod": "259200s",
  "displayName": "75% Availability in Rolling 3 Days"
}
EOF
)
```

3. Post the request body to the endpoint:

```
curl  --http1.1 --header "Authorization: Bearer ${ACCESS_TOKEN}" --header "Content-Type:
```

Optionally, you can also specify a desired ID for the SLO as the value of the `service_level_objective_id` query parameter:

```
SLO_ID=test-slo-id

curl  --http1.1 --header "Authorization: Bearer ${ACCESS_TOKEN}" --header "Content-Type:
```