**GCP version 2 Assignment 1**

Q1.What is Google Cloud Platform Document AI? Ans:- Document AI in Google Cloud Platform

Document AI

Document AI, in beta, offers a scalable, serverless platform to automatically classify, extract, and enrich data from your scanned documents. It converts unstructured data into structured data.

Internally it uses the same deep machine learning technology that powers Google Search, Google Assistant, Natural Language Processing API to derive valuable insights from your unstructured documents.

**Use Case**

The organisation has multiple field offices and the head office runs the HR / Payroll system. New joiner’s details are filled in a form by field managers and then forms are being sent to head office. The operator enters the details manually to the system and then Employee’s email, training, access card, laptop, and other formalities get sorted.

This end-2-end process takes days and till then employee sits idle. In my opinion new joiners (be it fresher or experienced) have always eager to demonstrate his/her talent or skill so they should not idle in their early days.

how to automate document processing and end-to-end new joiner process:

* Field managers fill the form and upload the scanned form
* The application extracts the data from document and store into database
* Then alerts to other processes, new joiners employee\_id
* Other processes like ID Cards, Laptop, Desk, etc will fetch the details from the database using employee\_id

**Components**

The following ***Serverless*** components are used in this architecture. This means that you will pay per use, without any up-front costs. Also, no servers need to be configured or maintained.

1. Front-end app to upload the scanned document on **Cloud Run**.
2. The document is stored in **Google Cloud Storage**.
3. This triggers the **Cloud Function**.
4. The Cloud Function calls **Document AI** to fetch the entities.
5. The Cloud Function reads the response generates employee\_id, email, and stores data to **Cloud Firestore.**
6. The new-joiner notification is sent to **Cloud Pub/Sub**topic.
7. This topic has multiple subscribers: Desk Service, Laptop Service, ID Card Service. These services will fetch the details from Firestore.
8. The Service deployed using **Cloud Run** which has end-point to GET the details of an employee from Cloud Firestore.
9. All components are logging data to **Stackdriver**.

Q2. What are the database services offered by Google's cloud platform?

Ans:-

1. **Cloud Storage :**  
Google Storage is a service offered by Google for storing the data objects into Google Cloud. A data object can be termed as an immutable entity that consists of data of any file irrespective of the format. It is best for the applications containing structured data objects, for example, large media files and images. Unstructured type data objects are also supported which is used for backups.

1. **Cloud Spanner :**  
   Google Spanner is the first of its kind data storage option, which supports relational database structure with a non-relational horizontal scale. It is highly scalable which consistent SQL support and is available at a high rate. It is best for applications having a large database for example e-commerce, trading, etc.
2. **Cloud SQL :**  
   Cloud SQL is a bundled service that is offered by Google Cloud. This supports relational database services for [MySQL](https://www.geeksforgeeks.org/mysql-common-mysql-queries/" \t "_blank), SQL Server, and PostgreSQL. This is used to store the client’s credentials and orders which are placed by the customers.
3. **Google Big Table :**  
   Google Big Table is Google’s Big Data services. It is used in many known google applications such as Maps, Search, and Gmail. it is also used where the analysis work and heavy read and write cases.
4. **Cloud Data Store :**  
   Google Datastore is the best suited for the applications where the data is structured and applications process that type of data.his data store is highly used by software developers who are intended to work with structured data.

**Difference between Services :**

| **Basis of** | **Cloud Datastore** | **Google Bigtable** | **Cloud Storage** | **Cloud SQL** | **Cloud Spanner** |
| --- | --- | --- | --- | --- | --- |
| Type | Uses NoSQL document | NoSQL wide column | Blobstore | Relational SQL for OLTP | Relational SQL for OLTP |
| Transactions Support | No | No | No | Supported | Supported |
| Supports Complex queries | No | No | No | Yes it supports | Yes it supports |
| Capacity | More than Terabytes | More than Petabytes | More than Petabytes | Terabytes | Petabytes |
| Entity Size | 1 MB/entity | 10 MB/row | 5 TB/object | Decided by database engine | 10, 240 MiB/row |

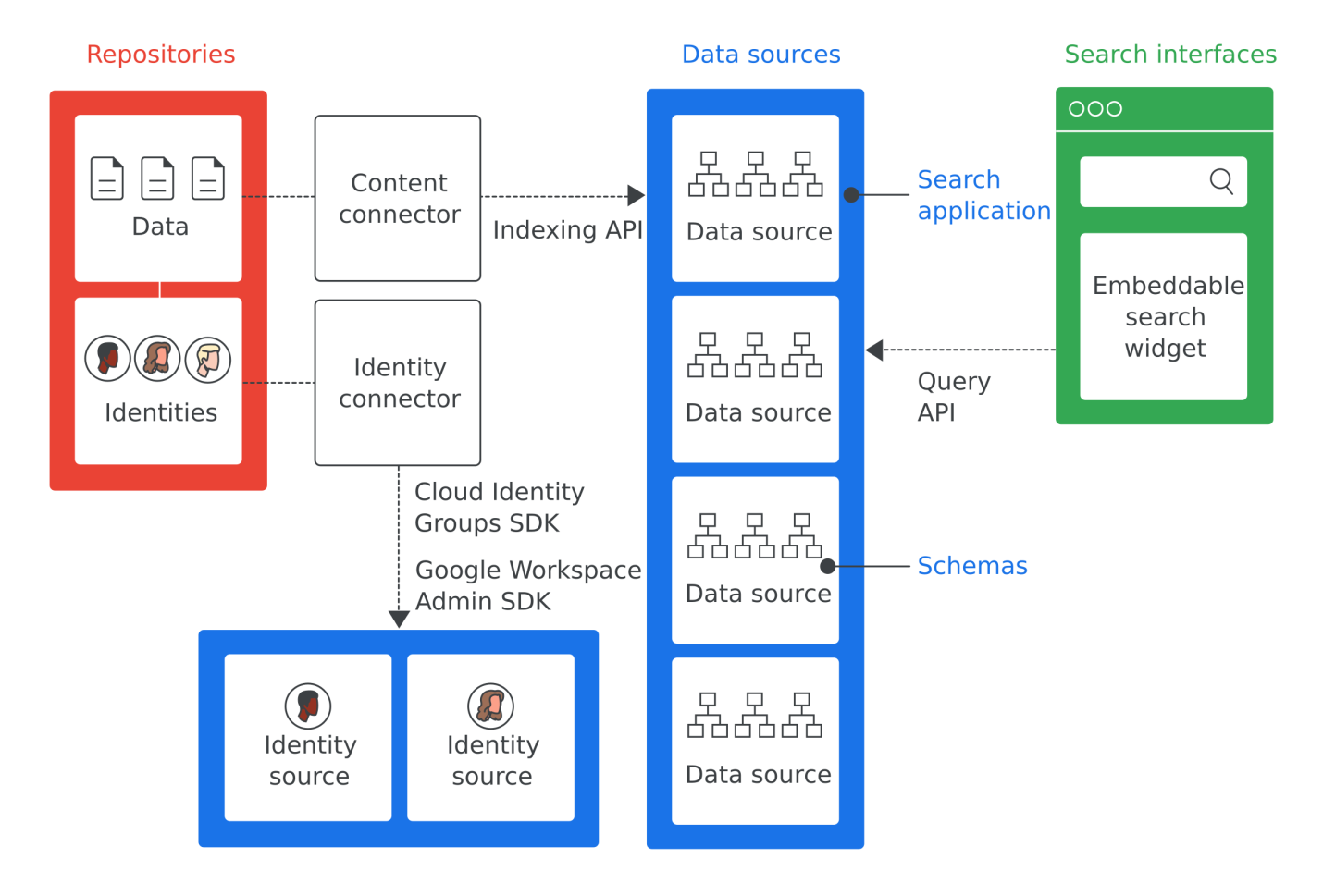
Q3. What is the difference between cloud search and cloud identity? Prepare a list of the applications for cloud search and cloud identity.

Ans:- Google Cloud Search

Google Cloud Search allows employees of a company to search and retrieve information, such as internal documents, database fields, and CRM data, from the company's internal data repositories.

## Architectural overview

Figure 1 shows all the key components of a Google Cloud Search implementation:



**Figure 1.** Key components of Google Cloud Search

Here are the definitions of the most important terms from Figure 1:

***Repository***

Software used by an enterprise to store its data, such as database used to store employee information.

***Data source***

Data from a repository that has been indexed and stored in Google Cloud Search.

***Search interface***

The user interface used by employees to search a data source. A search interface can be developed for use on any device, from a mobile phone to a desktop computer. The Google-provided search widget can also be deployed to enable search within your internal web sites. The search application ID is included with every search to ensure that the context of that search, such as within a customer service tool, is known. The site cloudsearch.google.com contains a search interface.

***Search application***

A group of settings that, when associated with a search interface, provide contextual information about searches. Contextual information includes the data sources and search rankings that should be used for a search using that interface. Search applications also include mechanisms for filtering results and enable reporting on data sources, such as number of queries made over a given time period.

***Schema***

A data structure outlining how the data in a enterprise’s repository should be represented for Google Cloud Search. A schema defines the employee Google Cloud Search experience, such as how things are filtered and displayed.

***Content connector***

A software program used to traverse the data in an enterprise's repository and populate a data source.

***Identity connector***

A software program used to sync enterprise identities (users and groups) to the identities required by Google Cloud Search.

**Cloud Identity**

Cloud Identity is an Identity as a Service (IDaaS) solution that centrally manages users and groups. You can configure Cloud Identity to federate identities between Google and other identity providers, such as [Active Directory](https://cloud.google.com/architecture/identity/federating-gcp-with-active-directory-introduction) and [Azure Active Directory](https://cloud.google.com/architecture/identity/federating-gcp-with-azure-active-directory).

Cloud Identity also gives you more control over the accounts that are used in your organization. For example, if developers in your organization use personal accounts, such as Gmail accounts, those accounts are outside of your control. When you adopt Cloud Identity, you can manage access and compliance across all users in your domain.

When you adopt Cloud Identity, you create a Cloud Identity account for each of your users and groups. You can then use Identity and Access Management (IAM) to manage access to Google Cloud resources for each Cloud Identity account.

**The list of applications for cloud search are:-**

* Google Docs:- With Google Docs, your documents are stored in the cloud, and you can access them from any computer or device. You create and edit documents right in your web browser; no special software is required.
* Google Sheets:- Google Sheets is an online spreadsheet app that lets users create and format spreadsheets while simultaneously working with other people. Businesses can use Google Sheets to maintain data consistency across departments and to ensure that every member of their team is on the same page.
* Gmail:- Google Cloud is a suite of cloud computing services that runs on the same infrastructure that Google uses internally for their own consumer products, such as Google Search, Gmail, and YouTube.

**Gmail** is a free service provided by [Google](https://www.computerhope.com/jargon/g/google.htm) that enables users to send and receive [e-mail](https://www.computerhope.com/jargon/e/email.htm) over the [Internet](https://www.computerhope.com/jargon/i/internet.htm).

* Google Drive:- Google Drive is a cloud-based storage service that enables users to store and access files online.

**List of applications for cloud identity**

You can use Cloud Identity accounts with other Google services, such as Google Cloud, Chrome, Android enterprise, and many third-party applications.

So the list of applications for cloud identity are:-

* Google cloud
* Chrome
* Android
* Enterprise
* Many third-party applications

Q4. What is conversational AI, and how does it work? List and Explain various GCP Conversation AI services.

### Ans:-

### Conversational AI

Conversational AI is a type of artificial intelligence (AI) that can simulate human conversation.

It is made possible by natural language processing (NLP), a field of AI that allows computers to understand and process human language. NLP is used to analyze the meaning of text and speech, and to generate responses that are appropriate and relevant to the conversation.

### conversational AI working

Conversational AI works by using a combination of [natural language processing](https://cloud.google.com/natural-language) (NLP) and machine learning (ML).

Conversational AI systems are trained on large amounts of data, such as text and speech. This data is used to teach the system how to understand and process human language. The system then uses this knowledge to interact with humans in a natural way. It’s constantly learning from its interactions and improving its response quality over time.

### GCP conversational AI services

A few examples include:

[Dialogflow CX](https://cloud.google.com/dialogflow): developer platform for building conversational agents

[Contact Center AI Platform](https://cloud.google.com/solutions/contact-center-ai-platform): a Contact Center as a Service (CCaaS) solution that offers security and privacy, along with unified data

[Agent Assist](https://cloud.google.com/agent-assist): real-time human-agent assistance

[CCAI Insights](https://cloud.google.com/solutions/ccai-insights): customer interactions insights for better service

[Natural Language API](https://cloud.google.com/natural-language#section-2): natural language understanding and processing

[Text-to-Speech API](https://cloud.google.com/text-to-speech#section-2): convert text into speech

[Speech-to-Text API](https://cloud.google.com/speech-to-text#section-2): convert speech into text

Q5. Give an example of GCP's Media Translation service.

Ans:- Media Translation has only one method so far to perform speech translation:

* **Streaming Translation** (gRPC only) performs translation on audio data provided within a [gRPC bi-directional stream](http://www.grpc.io/docs/guides/concepts.html" \l "bidirectional-streaming-rpc). Streaming requests are designed for real-time translation purposes, such as capturing live audio from a microphone. Streaming translation provides interim results while audio is being captured, allowing result to appear, for example, while a user is still speaking. Streaming translation requests are limited to audio data of 5 minutes or less in duration.

Requests contain either configuration parameters or audio data. The following sections describe these type of speech translation requests, the responses they generate, and how to handle those responses in more detail.

### Streaming speech translation requests

A streaming Media Translation API request can be either a speech translation configuration, or audio data. A sample configuration request is shown below:

{  
    "audio\_config": {  
        "audio\_encoding": "linear16",  
        "sample\_rate\_hertz": 16000,  
        "source\_language\_code": "en-US",  
        "target\_language\_code": "zh",  
        "model" : "google-provided-model/video",  
    }  
    "single\_utterance" : False  
}

A sample audio data request is shown below:

{  
    "audio\_content " : "\366c\256\375jQ\r\312\205j\271\243%/u\216z\330\354\221\360\253KJ\005\"  
}

The first StreamingTranslateSpeechRequest must contain a configuration of type [StreamingTranslateSpeechConfig](https://cloud.google.com/translate/media/docs/reference/rpc/google.cloud.mediatranslation.v1beta1" \l "StreamingTranslateSpeechConfig) without any accompanying audio. Subsequent StreamingTranslateSpeechRequests sent over the same stream will then consist of consecutive frames of raw audio bytes.

A StreamingTranslateSpeechConfig contains the following fields:

* audio\_config - (required) contains configuration information for the audio, of type [TranslateSpeechConfig](https://cloud.google.com/translate/media/docs/reference/rpc/google.cloud.mediatranslation.v1beta1" \l "TranslateSpeechConfig).
* single\_utterance - (optional, defaults to false) indicates whether this request should automatically end after speech is no longer detected. If set, Media Translation will detect pauses, silence, or non-speech audio to determine when to end translation. If not set, the stream will continue to listen and process audio until either the stream is closed directly, or the stream's limit length has been exceeded. Setting single\_utterance to true is useful for processing voice commands.

A TranslateSpeechConfig contains the following sub-fields:

* audio\_encoding - (required) specifies the encoding scheme of the supplied audio (of type AudioEncoding). If you have a choice in codec, prefer a lossless encoding such as FLAC or LINEAR16 for best performance. (For more information, see [Audio Encodings](https://cloud.google.com/translate/media/docs/encoding#audio-encodings).)
* sample\_rate\_hertz - (required) specifies the sample rate (in Hertz) of the supplied audio. (For more information on sample rates, see [Sample Rates](https://cloud.google.com/translate/media/docs/basics#sample-rates) below.)
* source\_language\_code - (required) contains the language + region/locale to use for speech recognition of the supplied audio. The language code must be a [BCP-47](https://tools.ietf.org/html/bcp47) identifier. Note that language codes typically consist of primary language tags and secondary region subtags to indicate dialects (for example, 'en' for English and 'US' for the United States in the above example.) (For a list of supported languages, see [Supported Languages](https://cloud.google.com/translate/media/docs/languages).)
* 'target\_language\_code' - (required) contains the language to use for text translation of the supplied audio. The language code must be a [BCP-47](https://tools.ietf.org/html/bcp47) identifier. Note that language codes typically only consist of primary language tags, since translation text will not' consider dialects. However, "zh-CN" and "zh-TW" will be different translation texts. (For a list of supported languages, see [Supported Languages](https://cloud.google.com/translate/media/docs/languages).)

Audio is supplied to Media Translation through the audio\_content field of type [StreamingTranslateSpeechRequest](https://cloud.google.com/translate/media/docs/reference/rpc/google.cloud.mediatranslation.v1beta1" \l "StreamingTranslateSpeechRequest). The audio\_content field contains the audio to evaluate, embedded within the request. See [Embedding Audio Content](https://cloud.google.com/translate/media/docs/basics#embedded-audio) below for more information.

### Streaming speech translation response

Streaming speech translation results are returned within a series of responses of type [StreamingTranslateSpeechResponse](https://cloud.google.com/translate/media/docs/reference/rpc/google.cloud.mediatranslation.v1beta1" \l "StreamingTranslateSpeechResponse). Such a response consists of the following fields:

* speech\_event\_type contains events of type [SpeechEventType](https://cloud.google.com/translate/media/docs/reference/rpc/google.cloud.mediatranslation.v1beta1" \l "SpeechEventType). The value of these events will indicate when a single utterance has been determined to have been completed. The speech events serve as markers within your stream's response. When receive END\_OF\_SINGLE\_UTTERANCE, user need to stop sending requests, while waiting to receive remaining translation responses.
* results contains the list of results, which may be either interim or final results, of type [StreamingTranslateSpeechResult](https://cloud.google.com/translate/media/docs/reference/rpc/google.cloud.mediatranslation.v1beta1" \l "StreamingTranslateSpeechResult). The results list contains following the sub-fields:
  + translation contains translation text.
  + isFinal indicates whether the results obtained within this list entry are interim or are final.

More information on these request and response parameters appears below.

### Sample rates

You specify the sample rate of your audio in the sample\_rate\_hertz field of the configuration request, and it must match the sample rate of the associated audio content or stream. Sample rates between 8000 Hz and 48000 Hz are supported within Media Translation.

If you have a choice when encoding the source material, capture audio using a sample rate of 16000 Hz. Values lower than this may impair speech recognition accuracy, as consequence, translation quality will be reduced. And higher sample rate have no appreciable effect on speech recognition quality, and it may increase latency.

However, if your audio data has already been recorded at an existing sample rate other than 16000 Hz, do not resample your audio to 16000 Hz. Most legacy telephony audio, for example, use sample rates of 8000 Hz, which may give less accurate results. If you must use such audio, provide the audio to the Media Translation API at its native sample rate.

### Languages

Media Translation's recognition/translation engine supports a variety of language/dialect pairs. You specify the language (and national or regional dialect) of your audio within the request configuration's source\_language\_code and target\_language\_code field, using a [BCP-47](https://tools.ietf.org/html/bcp47) identifier.

A full list of supported languages for each feature is available on the [Language Support](https://cloud.google.com/translate/media/docs/languages) page.

### Selecting models

Media Translation can use one of several machine learning models to translation your audio file. Google has trained these models for specific audio types and sources.

When you send an audio translation request to Media Translation, you can improve the results that you receive by specifying the model. This allows the Media Translation API to process your audio files using a machine learning model trained to recognize speech audio from that particular type of source.

To specify a model for speech translation, include the model field in the [TranslateSpeechConfig](https://cloud.google.com/translate/media/docs/reference/rpc/google.cloud.mediatranslation.v1beta1" \l "TranslateSpeechConfig) object for your request, specifying the model that you want to use.

Media Translation can use the following types of machine learning models for translating your audio files.

**Note:** See the [supported languages](https://cloud.google.com/translate/media/docs/languages) page to see which models are available for your language.

|  |  |  |
| --- | --- | --- |
| **Type** | **Name String** | **Description** |
| Video | google-provided-model/video | Use this model for transcribing audio in video clips or that includes multiple speakers. For best results, provide audio recorded at 16,000Hz or greater sampling rate.  **Note:** This is a premium model that costs more than the standard rate. |
| Phone call | google-provided-model/(enhanced-)phone-call | Use this model for transcribing audio from a phone call. Typically, phone audio is recorded at 8,000Hz sampling rate.  **Note:** The enhanced phone model is a premium model that costs more than the standard rate. |
| media translation: Default | google-provided-model/default | Use this model if your audio does not fit one of the previously described models. For example, you can use this for long-form audio recordings that feature a single speaker only. Ideally, the audio is high-fidelity, recorded at 16,000Hz or greater sampling rate. |

### Embedding audio content

Embedded audio is included in the streaming speech translation request when passing an audio\_content field within the streaming request. For embedded audio provided as content within a gRPC request, that audio must be compatible for [Proto3](https://developers.google.com/protocol-buffers/docs/reference/proto3-spec) serialization, and provided as binary data.

When constructing a request using a [Google Cloud client library](https://cloud.google.com/sdk/cloud-client-libraries), you generally will write out this binary data directly within the audio\_content field.

To see code samples, see [Translating streaming audio](https://cloud.google.com/translate/media/docs/streaming).

## Code samples

### Translating speech from an audio file

To learn how to install and use the client library for Media Translation, see [Media Translation client libraries](https://cloud.google.com/translate/media/docs/libraries/client-libraries). For more information, see the [Media Translation Java API reference documentation](https://cloud.google.com/java/docs/reference/google-cloud-mediatranslation/latest/overview).

To authenticate to Media Translation, set up Application Default Credentials. For more information, see [Set up authentication for a local development environment](https://cloud.google.com/docs/authentication/provide-credentials-adc#local-dev).

import com.google.api.gax.rpc.BidiStream;  
import com.google.cloud.mediatranslation.v1beta1.SpeechTranslationServiceClient;  
import com.google.cloud.mediatranslation.v1beta1.StreamingTranslateSpeechConfig;  
import com.google.cloud.mediatranslation.v1beta1.StreamingTranslateSpeechRequest;  
import com.google.cloud.mediatranslation.v1beta1.StreamingTranslateSpeechResponse;  
import com.google.cloud.mediatranslation.v1beta1.StreamingTranslateSpeechResult;  
import com.google.cloud.mediatranslation.v1beta1.TranslateSpeechConfig;  
import com.google.protobuf.ByteString;  
import java.io.IOException;  
import java.nio.file.Files;  
import java.nio.file.Path;  
import java.nio.file.Paths;  
  
public class TranslateFromFile {  
  
  public static void translateFromFile() throws IOException {  
    // TODO(developer): Replace these variables before running the sample.  
    String filePath = "path/to/audio.raw";  
    translateFromFile(filePath);  
  }  
  
  public static void translateFromFile(String filePath) throws IOException {  
    // Initialize client that will be used to send requests. This client only needs to be created  
    // once, and can be reused for multiple requests. After completing all of your requests, call  
    // the "close" method on the client to safely clean up any remaining background resources.  
    try (SpeechTranslationServiceClient client = SpeechTranslationServiceClient.create()) {  
      Path path = Paths.get(filePath);  
      byte[] content = Files.readAllBytes(path);  
      ByteString audioContent = ByteString.copyFrom(content);  
  
      TranslateSpeechConfig audioConfig =  
          TranslateSpeechConfig.newBuilder()  
              .setAudioEncoding("linear16")  
              .setSampleRateHertz(16000)  
              .setSourceLanguageCode("en-US")  
              .setTargetLanguageCode("fr-FR")  
              .build();  
  
      StreamingTranslateSpeechConfig config =  
          StreamingTranslateSpeechConfig.newBuilder()  
              .setAudioConfig(audioConfig)  
              .setSingleUtterance(true)  
              .build();  
  
      BidiStream<StreamingTranslateSpeechRequest, StreamingTranslateSpeechResponse> bidiStream =  
          client.streamingTranslateSpeechCallable().call();  
  
      // The first request contains the configuration.  
      StreamingTranslateSpeechRequest requestConfig =  
          StreamingTranslateSpeechRequest.newBuilder().setStreamingConfig(config).build();  
  
      // The second request contains the audio  
      StreamingTranslateSpeechRequest request =  
          StreamingTranslateSpeechRequest.newBuilder().setAudioContent(audioContent).build();  
  
      bidiStream.send(requestConfig);  
      bidiStream.send(request);  
  
      for (StreamingTranslateSpeechResponse response : bidiStream) {  
        // Once the transcription settles, the response contains the  
        // is\_final result. The other results will be for subsequent portions of  
        // the audio.  
        StreamingTranslateSpeechResult res = response.getResult();  
        String translation = res.getTextTranslationResult().getTranslation();  
  
        if (res.getTextTranslationResult().getIsFinal()) {  
          System.out.println(String.format("\nFinal translation: %s", translation));  
          break;  
        }  
        System.out.println(String.format("\nPartial translation: %s", translation));  
      }  
    }  
  }  
}

### Translating speech from a microphone

[Java](https://cloud.google.com/translate/media/docs/streaming#java)[Node.js](https://cloud.google.com/translate/media/docs/streaming#node.js)[Python](https://cloud.google.com/translate/media/docs/streaming#python)

To learn how to install and use the client library for Media Translation, see [Media Translation client libraries](https://cloud.google.com/translate/media/docs/libraries/client-libraries). For more information, see the [Media Translation Java API reference documentation](https://cloud.google.com/java/docs/reference/google-cloud-mediatranslation/latest/overview).

To authenticate to Media Translation, set up Application Default Credentials. For more information, see [Set up authentication for a local development environment](https://cloud.google.com/docs/authentication/provide-credentials-adc#local-dev).

[View on GitHub](https://github.com/GoogleCloudPlatform/java-docs-samples/blob/HEAD/mediatranslation/src/main/java/com/example/mediatranslation/TranslateFromMic.java)

import com.google.api.gax.rpc.ClientStream;  
import com.google.api.gax.rpc.ResponseObserver;  
import com.google.api.gax.rpc.StreamController;  
import com.google.cloud.mediatranslation.v1beta1.SpeechTranslationServiceClient;  
import com.google.cloud.mediatranslation.v1beta1.StreamingTranslateSpeechConfig;  
import com.google.cloud.mediatranslation.v1beta1.StreamingTranslateSpeechRequest;  
import com.google.cloud.mediatranslation.v1beta1.StreamingTranslateSpeechResponse;  
import com.google.cloud.mediatranslation.v1beta1.StreamingTranslateSpeechResult;  
import com.google.cloud.mediatranslation.v1beta1.TranslateSpeechConfig;  
import com.google.protobuf.ByteString;  
import java.io.IOException;  
import javax.sound.sampled.AudioFormat;  
import javax.sound.sampled.AudioInputStream;  
import javax.sound.sampled.AudioSystem;  
import javax.sound.sampled.DataLine;  
import javax.sound.sampled.LineUnavailableException;  
import javax.sound.sampled.TargetDataLine;  
  
public class TranslateFromMic {  
  
  public static void main(String[] args) throws IOException, LineUnavailableException {  
    translateFromMic();  
  }  
  
  public static void translateFromMic() throws IOException, LineUnavailableException {  
  
    ResponseObserver<StreamingTranslateSpeechResponse> responseObserver = null;  
  
    // Initialize client that will be used to send requests. This client only needs to be created  
    // once, and can be reused for multiple requests. After completing all of your requests, call  
    // the "close" method on the client to safely clean up any remaining background resources.  
    try (SpeechTranslationServiceClient client = SpeechTranslationServiceClient.create()) {  
      responseObserver =  
          new ResponseObserver<StreamingTranslateSpeechResponse>() {  
  
            @Override  
            public void onStart(StreamController controller) {}  
  
            @Override  
            public void onResponse(StreamingTranslateSpeechResponse response) {  
              StreamingTranslateSpeechResult res = response.getResult();  
              String translation = res.getTextTranslationResult().getTranslation();  
  
              if (res.getTextTranslationResult().getIsFinal()) {  
                System.out.println(String.format("\nFinal translation: %s", translation));  
              } else {  
                System.out.println(String.format("\nPartial translation: %s", translation));  
              }  
            }  
  
            @Override  
            public void onComplete() {}  
  
            public void onError(Throwable t) {  
              System.out.println(t);  
            }  
          };  
  
      ClientStream<StreamingTranslateSpeechRequest> clientStream =  
          client.streamingTranslateSpeechCallable().splitCall(responseObserver);  
  
      TranslateSpeechConfig audioConfig =  
          TranslateSpeechConfig.newBuilder()  
              .setAudioEncoding("linear16")  
              .setSourceLanguageCode("en-US")  
              .setTargetLanguageCode("es-ES")  
              .setSampleRateHertz(16000)  
              .build();  
  
      StreamingTranslateSpeechConfig streamingRecognitionConfig =  
          StreamingTranslateSpeechConfig.newBuilder().setAudioConfig(audioConfig).build();  
  
      StreamingTranslateSpeechRequest request =  
          StreamingTranslateSpeechRequest.newBuilder()  
              .setStreamingConfig(streamingRecognitionConfig)  
              .build(); // The first request in a streaming call has to be a config  
  
      clientStream.send(request);  
      // SampleRate:16000Hz, SampleSizeInBits: 16, Number of channels: 1, Signed: true,  
      // bigEndian: false  
      AudioFormat audioFormat = new AudioFormat(16000, 16, 1, true, false);  
      DataLine.Info targetInfo =  
          new DataLine.Info(  
              TargetDataLine.class,  
              audioFormat); // Set the system information to read from the microphone audio stream  
  
      if (!AudioSystem.isLineSupported(targetInfo)) {  
        System.out.println("Microphone not supported");  
        System.exit(0);  
      }  
      // Target data line captures the audio stream the microphone produces.  
      TargetDataLine targetDataLine = (TargetDataLine) AudioSystem.getLine(targetInfo);  
      targetDataLine.open(audioFormat);  
      targetDataLine.start();  
      System.out.println("Start speaking... Press Ctrl-C to stop");  
      long startTime = System.currentTimeMillis();  
      // Audio Input Stream  
      AudioInputStream audio = new AudioInputStream(targetDataLine);  
  
      while (true) {  
        byte[] data = new byte[6400];  
        audio.read(data);  
        request =  
            StreamingTranslateSpeechRequest.newBuilder()  
                .setAudioContent(ByteString.copyFrom(data))  
                .build();  
        clientStream.send(request);  
      }  
    }  
  }  
}

Q6. Explain how to use Google Cloud Platform's cloud logging and monitoring features.

Ans:- [Cloud Logging](https://cloud.google.com/logging) is a fully managed service that performs at scale and can ingest application and platform log data, as well as custom log data from GKE environments, VMs, and other services inside and outside of Google Cloud. Get advanced performance, troubleshooting, security, and business insights with [Log Analytics](https://cloud.google.com/logging/docs/log-analytics), integrating the power of BigQuery into Cloud Logging.

[Cloud Monitoring](https://cloud.google.com/monitoring) provides visibility into the performance, uptime, and overall health of cloud-powered applications. Collect metrics, events, and metadata from Google Cloud services, hosted uptime probes, application instrumentation, and a variety of common application components. Visualize this data on charts and dashboards and create alerts so you are notified when metrics are outside of expected ranges.

## ****Logging and Monitoring in the Google Cloud****

Cloud logging and monitoring is the process of storing logs across all Cloud products with the possibility to search, monitor, and alert based on various metrics. You can store, search, analyze, monitor, and alert on log data from services such as Compute Engine instances, Cloud Storage services, Cloud Big Data analytical services, Cloud Functions, AI infrastructure, Cloud CDN, and more. In this article, we will discuss what opportunities Google Cloud offers around [Cloud Logging](https://cloud.google.com/logging) and [Monitoring](https://cloud.google.com/monitoring/quickstart-aws), and the possibilities for integration with third-party tools.

Google Cloud Logging and Monitoring not only allows you to log and monitor your services running on Google Cloud but also on [Amazon Web Services](https://cloud.google.com/monitoring/quickstart-aws). With Google Cloud Logging and Monitoring (GCLM) being a fully managed service, you do not have to worry about setting up, maintaining, or scaling servers. It can ingest application and system log data from thousands of sources concurrently. Another impressive feature that GCLM has is the ability to analyze this incoming data in real-time while it synchronizes server instances and manages different time zones.

Log entries are data chunks generated by a service running in Google Cloud, third-party applications, or even your own code. Examples of log entries include a user accessing your storage bucket, details of a Compute Engine instance starting up, a call made to the AI infrastructure, an action to query Cloud SQL or BigQuery, your application code writing to the standard or error output files, and more. GCLM allows you to query these log entries based on parameters such as timestamps, log levels, resource names, namespaces, or any custom parameters you define.

Google Cloud provides powerful in-house tools such as the [Logs Explorer](https://cloud.google.com/logging/docs/view/logs-viewer-preview) and the [Logging API](https://cloud.google.com/logging/docs/api) to help in querying and analyzing log entries. If you like, you have the option to export your logs to other Google services such as[BigQuery](https://www.happtiq.com/data-warehousing-with-bigquery/), [Cloud Storage](https://cloud.google.com/storage), [Pub/Sub](https://cloud.google.com/pubsub) after which you can export to any storage service of your choice for more advanced analytics, archival purposes, or any other reason.

Q7. How to use Cloud Identity to generate and manage user IDs in the cloud?

# Ans:- Cloud Identity: A Smart Way to Manage Gmail Users

## Cloud Identity is a platform that helps organisations to get the flexibility of management according to their requirement over the Google server.

# Create Cloud Identity user accounts

Before you can manage users in Cloud Identity, you must create accounts for each user to be managed by Cloud Identity. An account provides users with a name and password for signing in to their Google services. Optionally, you can add more information to your users’ profiles.

## Create one or more user accounts

* [Add users individually](https://support.google.com/cloudidentity/answer/33310) using your Google Admin console. https://www.google.com/images/icons/feature/checkmark-lb16.png **Easiest**
* [Add several users at once](https://support.google.com/cloudidentity/answer/40057) by uploading their names in a CSV file.

## Options for large organisations

* Use [Google Cloud Directory Sync](https://support.google.com/cloudidentity/answer/106368) to synchronize user data in your existing LDAP directory with your Google account (syncs groups, contacts, and organizations, too).
* Use the [Admin SDK Directory API](https://developers.google.com/admin-sdk/directory/) to provision a large number of users with data from your existing LDAP directory, such as Microsoft® Active Directory®. This API provides more flexibility than Google Cloud Directory Sync, but requires programming.

## Guidelines when creating accounts

**Important**: We recommend that you resolve any potential [conflicting accounts](https://support.google.com/cloudidentity/answer/7062710) before you add users to your organization's new Cloud Identity account.

Other guidelines:

* Each account's username becomes that person's login name and the first part of their email address. If your domain is **solarmora.com**, a user whose email is **jsmith@solarmora.com** has the username **jsmith**. If your organization's Cloud Identity account has multiple domain names associated with it, you specify which domain name to use when you create the Cloud Identity user account.
* It can take up to 24 hours for a new user account to appear in the [searchable G Suite Directory](https://support.google.com/a/answer/1628008).

See also: [Name and password guidelines](https://support.google.com/cloudidentity/answer/9193374).

## Do your users have existing Google accounts?

If a user created a personal Google Account using the domain name of your organization, it may result in a [conflicting account](https://support.google.com/cloudidentity/answer/7062710). If you create user accounts with the same username as existing personal Google Accounts, then add them to your organization, they'll have the same address for their personal and Cloud Identity accounts. Two accounts can't share the same username.

For help on resolving conflicting accounts, see [Resolve conflict accounts](https://support.google.com/cloudidentity/answer/185186).