Et si on construisait une API qui résume des vidéos ?

Avec Bedrock & Claude

Bonjour !

Zied Ben Tahar

/ **AWS Community builder**
/ **Principal Architect @ AVIV Group**

/ **Blog**: https://zied-ben-tahar.medium.com/
/ **Github** : @ziedbentahar
/ **X** : @wow_sig

aviv_

AWS community builder

Agenda

01 Un aperçu rapide de la Gen AI avec Bedrock et Claude d'Anthropic

02 Comment créer un service qui résume des vidéos avec Claude et Bedrock ?

03 Comment faire mieux ?

04 Demo

01

Un aperçu rapide de la Gen AI avec Bedrock et Claude

Amazon Bedrock

Un service d'IA Generative managé

Plusieurs FM

Accès à plusieurs modèles de fondations issus de plusieurs acteurs en IA

Personnalisable

Possibilité d'adapter les FM de manière privée aux problèmes spécifiques du domaine.

Serverless

Pas de serveurs à gérer, une bonne intégration avec l'écosystème serverless d'AWS

AI21 labs AI21 Labs Jurassic | Amazon Titan | Anthropic's Claude

Cohere Command | Meta Llama 2 | Stability AI SDXL

MISTRAL AI_

Anthropic Claude 3

Un alternative sérieuse à GPT 4

- ➔ Peut gérer des contextes très larges: 200K tokens (150k mots)
- ➔ Capable de maintenir la cohérence factuelle lors du traitement de longs prompts
- ➔ Peut comprendre et générer du code et des données structurées
- ➔ Fonctionnalités de vision

	Claude 3 Opus	Claude 3 Sonnet	Claude 3 Haiku	GPT-4	GPT-3.5	Gemini 1.0 Ultra	Gemini 1.0 Pro
Undergraduate level knowledge *MMLU*	86.8% 5 shot	79.0% 5-shot	75.2% 5-shot	86.4% 5-shot	70.0% 5-shot	83.7% 5-shot	71.8% 5-shot
Graduate level reasoning *GPQA, Diamond*	50.4% 0-shot CoT	40.4% 0-shot CoT	33.3% 0-shot CoT	35.7% 0-shot CoT	28.1% 0-shot CoT	–	–
Grade school math *GSM8K*	95.0% 0-shot CoT	92.3% 0-shot CoT	88.9% 0-shot CoT	92.0% 5-shot CoT	57.1% 5-shot	94.4% Maj1@32	86.5% Maj1@32
Math problem-solving *MATH*	60.1% 0-shot CoT	43.1% 0-shot CoT	38.9% 0-shot CoT	52.9% 4-shot	34.1% 4-shot	53.2% 4-shot	32.6% 4-shot
Multilingual math *MGSM*	90.7% 0-shot	83.5% 0-shot	75.1% 0-shot	74.5% 8-shot	–	79.0% 8-shot	63.5% 8-shot
Code *HumanEval*	84.9% 0-shot	73.0% 0-shot	75.9% 0-shot	67.0% 0-shot	48.1% 0-shot	74.4% 0-shot	67.7% 0-shot
Reasoning over text *DROP, F1 score*	83.1 3-shot	78.9 3-shot	78.4 3-shot	80.9 3-shot	64.1 3-shot	82.4 Variable shots	74.1 Variable shots
Mixed evaluations *BIG-Bench-Hard*	86.8% 3-shot CoT	82.9% 3-shot CoT	73.7% 3-shot CoT	83.1% 3-shot CoT	66.6% 3-shot CoT	83.6% 3-shot CoT	75.0% 3-shot CoT

Claude 3 benchmarks

(source: https://www.anthropic.com/news/claude-3-family)

L’incontournable “Hello World”

Invoker Bedrock avec aws sdk

```
const client = new BedrockRuntimeClient(...);

...

await client.send(
 new InvokeModelCommand({
  modelId: "anthropic.claude-v2:1",
  contentType: "application/json",
  accept: "*/*",
  body: JSON.stringify({
   prompt: "\\n\\nHuman:{{your prompt}}\\n\\nAssistant:",
   max_tokens_to_sample: 300,
   temperature: 0.5,
   top_k: 250,
   top_p: 1,
   stop_sequences: ["\\n\\nHuman:"],
   anthropic_version: "bedrock-2023-05-31",
  }),
 }),
);
```

```
const client = new BedrockRuntimeClient(...);

...

await client.send(
  new InvokeModelCommand({
    modelId: "anthropic.claude-3-haiku-20240307-v1:0",
    contentType: "application/json",
    accept: "application/json",
    body: JSON.stringify({
      system: "...",
      messages: [{ role: "user", content: [{ type: "text", text: "..." }] }],
      anthropic_version: "bedrock-2023-05-31",
      max_tokens: 1000,
      temperature: 1,
    }),
  }),
);
```

~ **“température” plus élevée:** Sortie plus aléatoire et diversifiée
~ **“top-p” plus bas:** Réduit la diversité et fixe une probabilité seuil et sélectionne les meilleurs tokens
~ **“top-k” plus bas:** Limite le nombre d’options puis échantillonne en fonction de probabilités

02

Comment créer un service qui résume des vidéos avec Claude et Bedrock ?

Vue d’ensemble

Une première approche

https://www.youtube.com/watch?v=XRB6IYvoJeY


```
{
  "speakers": [
    "Michael",
    "Ayan",
    "Fahd"
  ],
  "topics": "Making semantic search and RAG real, how to build a prod-ready app",
  "summary": [
    "Michael introduced the topic of making semantic search and RAG real to build produ
    "Ayan talked about Amazon Bedrock which provides foundation models to build generat
    "Michael discussed Elasticsearch capabilities for vector search like approximate ne
    "Fahd shared a use case from Adobe Commerce using lexical expansion with fine tuned
  ],
  "audioUrl": "S3 presigned URL..."
}
```


01 — Extraire ou générer la transcription de la vidéo

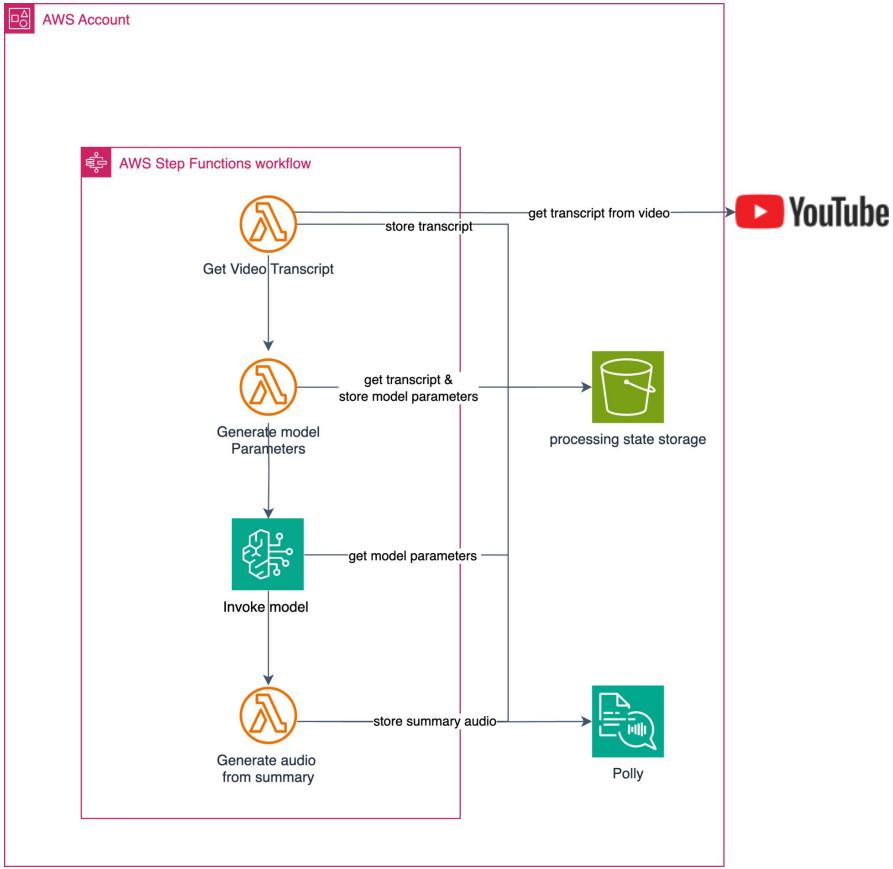
02 — Résumer la transcription dans un format structuré

03 — Générer l’audio à partir du résumé

04 — Enjoy !

Vue d’ensemble

Une première approche

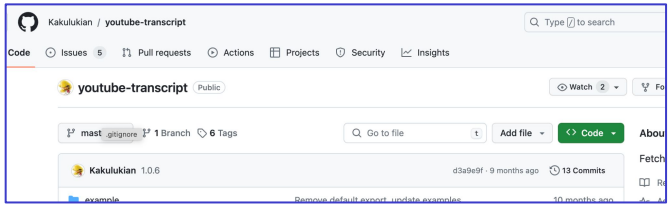


Code source

Récupérer les transcriptions

La version naïve...mais rapide

Le cheat code pour récupérer les transcriptions



🔗 https://github.com/Kakulukian/youtube-transcript

```ts
import { storeTranscript } from "adapters/transcript-repository";
import { YoutubeTranscript } from "youtube-transcript";

export const handler = async (event: {
  youtubeVideoUrl: string;
  requestId: string;
}) => {
  const { youtubeVideoUrl, requestId } = event;
  const transcript = await YoutubeTranscript.fetchTranscript(youtubeVideoUrl);
  const sentences = Array.from(getSentencesFromYoutubeTranscript(transcript));

  await storeTranscript(requestId, sentences.join("\n"));
};
```

~ Rapide: récupère les transcriptions déjà générés par Youtube
~ ...Mais c'est une solution fragile, se basant sur une API de Youtube non officielle, ne supporte pas toutes les langues

Deep dive

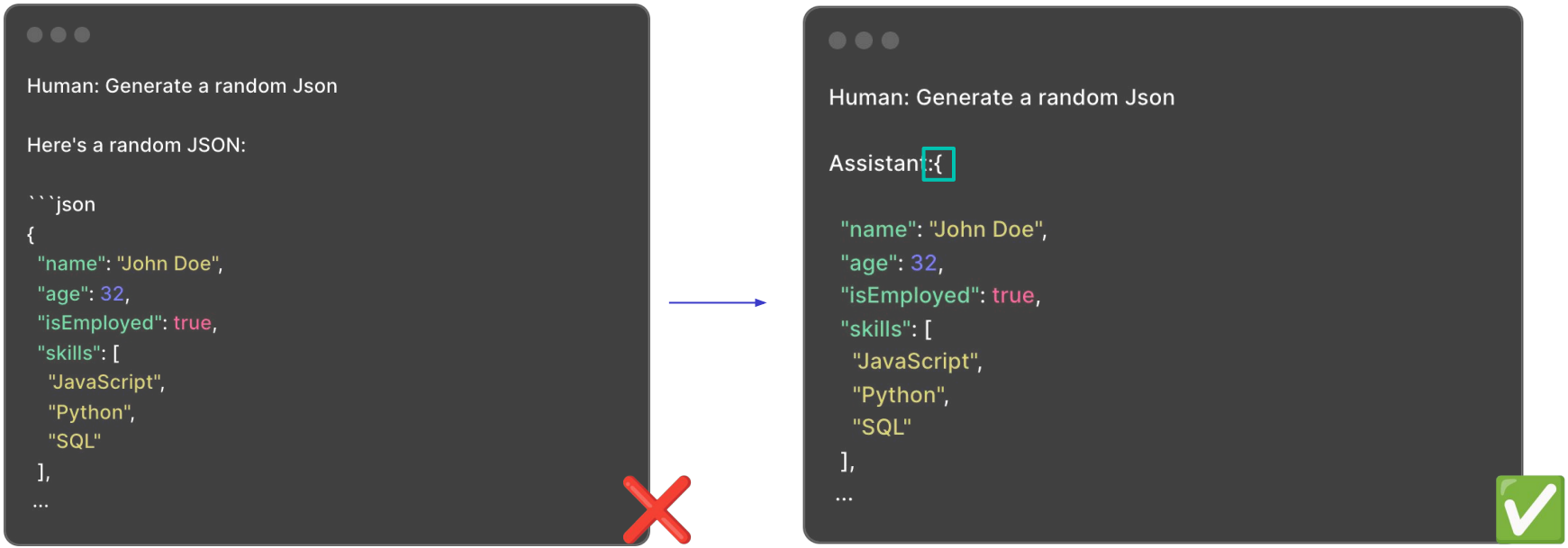
Ecrire des prompts Claude efficaces

```
You are a video transcript summarizer.
Here is a video transcription that you must summarize:

<transcript>{{transcript}}</transcript>

Summarize this transcript in a third person point of view in 10 sentences.
Identify the speakers and the main topics of the transcript and add them in the
output as well.

Do not add or invent speaker names if you are not able to identify them.
You must output the summary JSON format conforming to this JSON schema:

<output-json-schema>
{
  "type": "object",
  "properties": {
    "speakers": {
      "type": "array",
      "items": {
        "type": "string"
      }
    },
    "topics": {
      "type": "string"
    },
    "summary": {
      "type": "array",
      "items": {
        "type": "string"
      }
    }
  }
}
</output-json-schema>
```

- ➔ Clair et sans ambiguïté, ne laissons pas Claude supposer des choses pour nous
- ➔ Avec des contraintes d'output
- ➔ Exemple concret / Json schema pour les résultats structurés

Deep dive

Aider Claude à générer des formats structurés



Deep dive

```
next(
    new LambdaInvoke(this, "generate-inference-parameters", {
      lambdaFunction: generateModelParameters,
      payload: TaskInput.fromObject({
        "requestId.$": "$$.Execution.Name",
      }),
    }).addCatch(failState)
  )
  .next(
    new CustomState(this, "bedrock-invoke-model", {
      stateJson: {
        Type: "Task",
        Resource: "arn:aws:states:::bedrock:invokeModel",
        Parameters: {
          ModelId: "anthropic.claude-3-sonnet-20240229-v1:0",
          Input: {
            "S3Uri.$": `$.Payload.modelParameters`,
          },
          ContentType: "application/json",
        },
        ResultSelector: {
          "requestId.$": "$$.Execution.Name",
          "summaryTaskResult.$":
           "States.StringToJson(States.Format('\\{{}',
$.Body.content[0].text))",
        },
      },
    })
      .addCatch(failState)
      .next(
        new LambdaInvoke(this, "generate-audio-from-summary", {
          lambdaFunction: generateAudioFromSummary,
        }).addCatch(failState)
      )
```

Définition de la machine à état en CDK

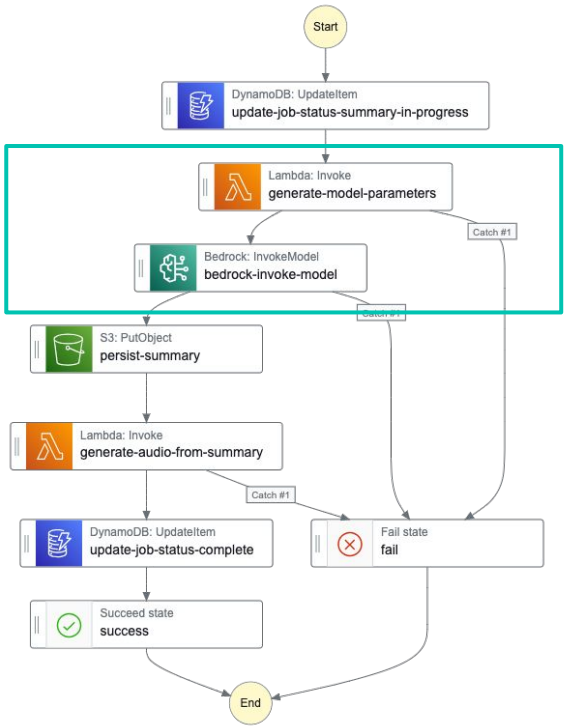
1- Générer les paramètres d'inférence du modèle et écrire dans un bucket S3

2- Intégration directe avec Bedrock

3- “Patch” le résultat de Claude pour avoir un format JSON correct

Deep dive

Invocation du modèle




```
{
  "name": "bedrock-invoke-model",
  "output": {
    "jobId": "37e24860-efdb-4228-a407-63163a2ec051",
    "summaryTaskResult": {
      "speakers": [
        "Michael Hildebrandt",
        "Andre",
        "Fad Sidiki"
      ],
      "topics": "Making semantic search and AI real, building a production-ready app with semantic sea
      "summary": [
        "Michael, Andre and Fad talked about making semantic search and AI real to build production-re
        "They discussed how search is evolving from just keywords to conversational, semantic search."
        "Customers want search to be more natural, not just entering keywords.",
        "To make semantic search real, companies need to choose appropriate AI models, build semantic
        "Andre discussed Amazon Bedrock which provides foundation models to build AI applications.",
        "Companies can customize models using techniques like fine-tuning and retrieval-augmented gene
        "Elasticsearch provides semantic search capabilities to retrieve relevant data and context.",
        "Michael explained Elasticsearch's vector search capabilities for storing, indexing and search
        "He highlighted performance optimizations done in Elasticsearch like parallel segment search."
        "Fad discussed an ecommerce use case of enriching product catalog data to improve search recal
        "He explained using lexical expansion with fine-tuned domain-specific language models to expan
        "This helps match user search queries better than raw product data.",
        "He emphasized the need for a flexible platform like Elasticsearch to run experiments between
      ]
    }
  },
  "outputDetails": {
```

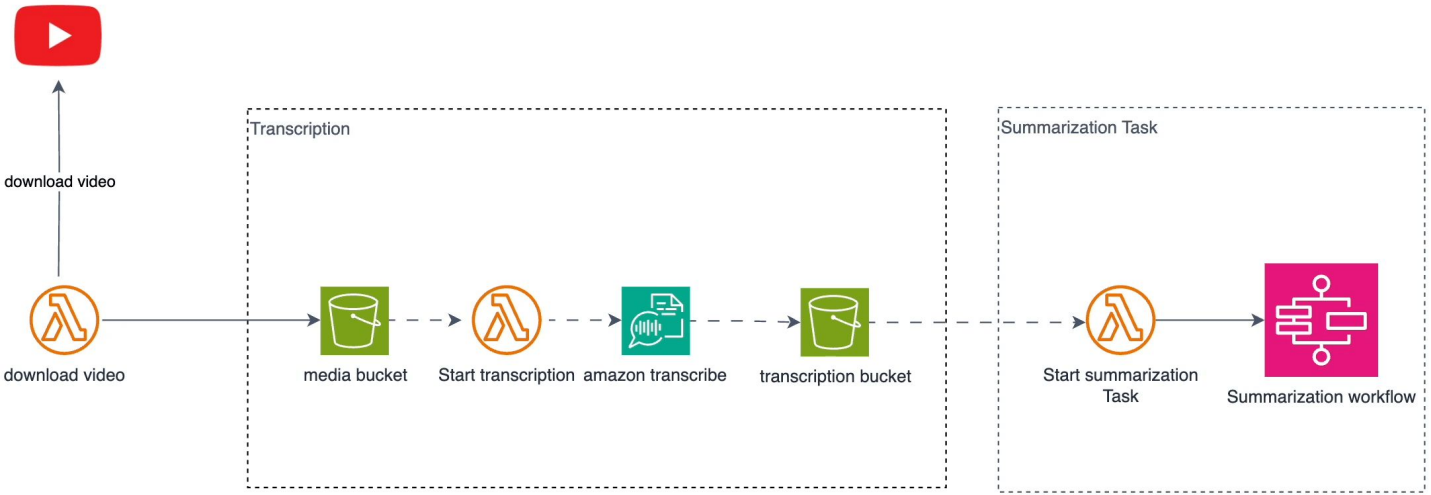
C’est bien !
Mais peut-on mieux faire ?

03

Comment faire mieux ?

Transcription plus stable

Avec Amazon Transcribe



~ Supportant plusieurs types de médias et plusieurs langues

~ Une conséquence architecturale: Le système est maintenant **asynchrone** et **basé sur les événements**

Transcription plus stable

Avec Amazon Transcribe

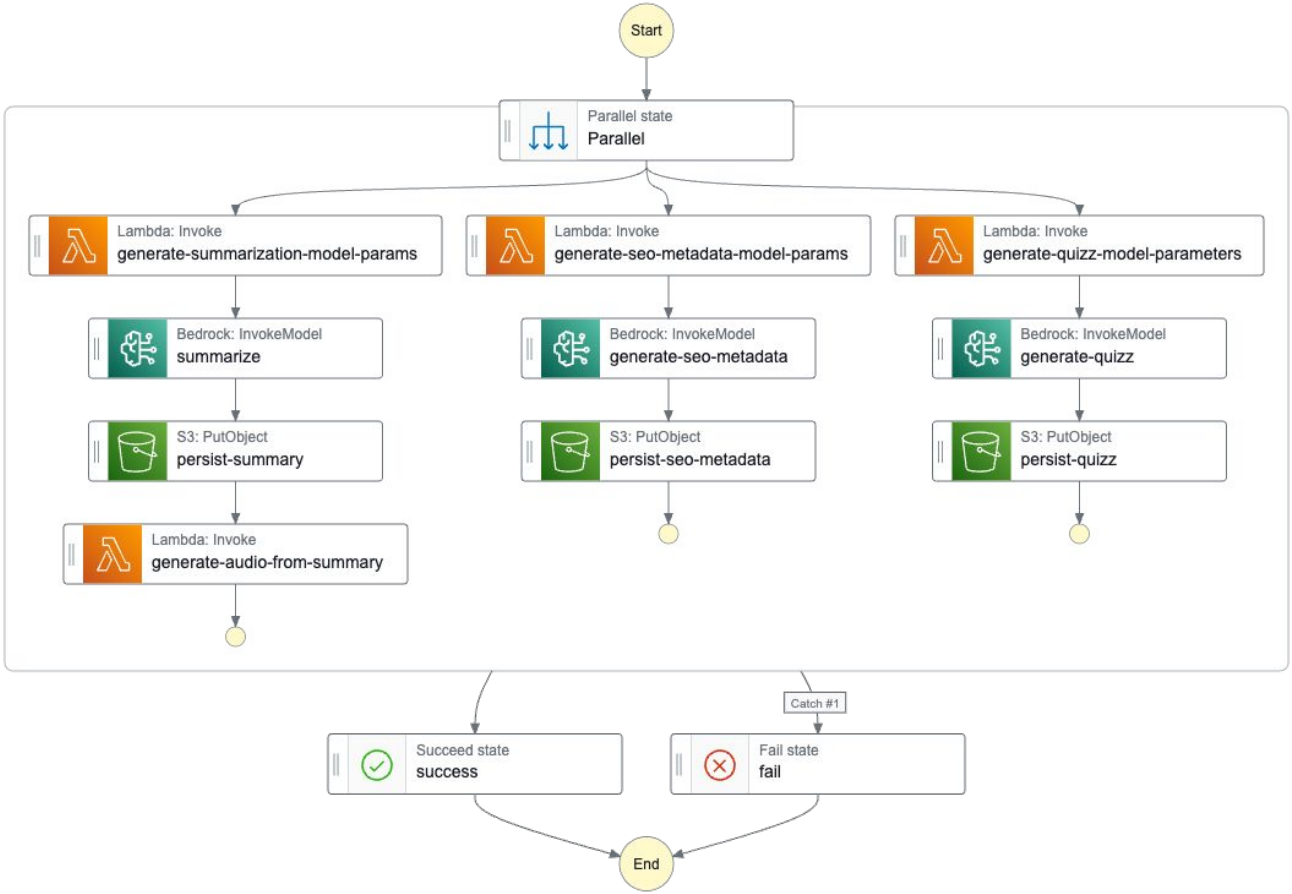
```
export const handler = async (event: S3Event) => {
  for (const record of event.Records) {
    const bucketName = record?.s3?.bucket?.name;
    const objectKey = record?.s3?.object?.key;

    const jobId = Conventions.getJobIdFromObjectKey(objectKey);

    const fileUri = `s3://${bucketName}/${objectKey}`;

    await transcribeClient.send(
      new StartTranscriptionJobCommand({
        Settings: { ShowSpeakerLabels: true, MaxSpeakerLabels: 5 },
        TranscriptionJobName: jobId,
        Media: { MediaFileUri: fileUri },
        IdentifyLanguage: true,
        OutputBucketName: config.getMediaTranscriptBucketName(),
        OutputKey: Conventions.getRawTranscriptKey(jobId),
      })
    );

    await updateJobStatus(jobId, JobStatus.TransciptInProgress);
  }
};
```

Démarrer la transcription

```
export const handler = async (event: S3Event) => {
  for (const record of event.Records) {
    const bucketName = record?.s3?.bucket?.name;
    const objectKey = record?.s3?.object?.key;

    const rawTranscriptObject = await s3Client.send(
      new GetObjectCommand({ Bucket: bucketName, Key: objectKey }));

    const transcriptResult = JSON.parse(
      await rawTranscriptObject.Body.transformToString()
    );

    const jobId = transcriptResult.jobName as string;
    const detectedLanguageCode = transcriptResult.results.language_code;

    await sfnClient.send(
      new StartExecutionCommand({
        stateMachineArn: process.env.STATE_MACHINE_ARN,
        name: jobId,
        input: JSON.stringify({jobId, detectedLanguageCode}),
      }));
  }
};
```

Démarrer la tâche du résumé

Sky’s the limit…



👌Le champ des possibles est immense: Generation de quizz, flashcards, metadonnées seo, images, podcasts, etc.

Sky’s the limit…
(ou presque)

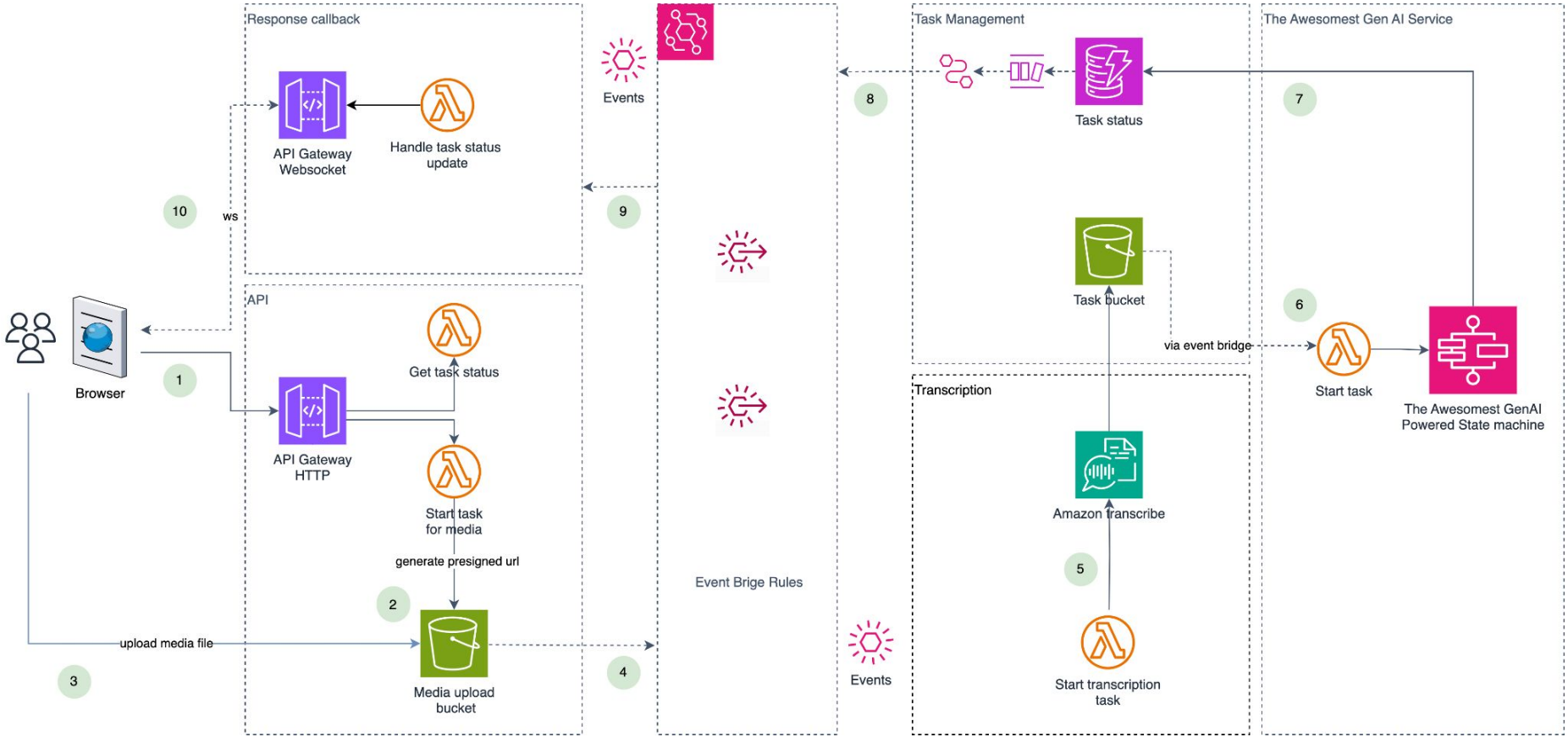
Model	Requests processed per minute	Tokens processed per minute	Adjustable
Anthropic Claude Instant	400	300,000	No
Anthropic Claude 2.x	100	200,000	No
Anthropic Claude 3 Sonnet	100	200,000	No
Anthropic Claude 3 Haiku	400	300,000	No

Garder un oeil sur les quotas de Bedrock & Caude

```
"Catch": [
 {
  "ErrorEquals": [
   "States.ALL"
  ],
  "Next": "fail"
 }
],
"Retry": [
 {
  "ErrorEquals": [
   "States.ALL"
  ],
  "BackoffRate": 2,
  "IntervalSeconds": 5,
  "MaxAttempts": 5
 }
]
```

Ne pas oublier les “retries” lors de vos appels à Bedrock depuis machine à état

The AWSomest GenAI-powered API



04

Demo

New Task

Get the most out of your medias

Get started by selecting a new task type

Import and summarize video files
Import your own video files, get transcripts and generate summary

Import and summarize audio files
Import your own audio files, get transcripts and generate summary

Summarize youtube videos
Extract summary from youtube videos

Exlore other tasks →

Merci !

Questions ?

Code source 👇

Zied Ben Tahar

/ **Blog**: https://zied-ben-tahar.medium.com/
/ **Github :** @ziedbentahar
/ **X** : @wow_sig

Gen AI et LLMs

Quelques défis

Comment résoudre quelques problèmes liés au knowledge cut-off et aux hallucinations ?

- ➔ **Prompt engineering**
  - ◆ Optimiser les prompts pour exécuter des tâches spécifiques et guider le modèle pour produire les réponses attendues.
- ➔ **Retrieval augmented generation (RAG)**
  - ◆ Permettre aux modèles d'accéder à des informations plus à jour ou à des informations internes afin d’améliorer leur réponse
- ➔ **Model fine-tuning**
  - ◆ Ajuster les paramètres du modèle et le spécialiser pour un domaine particulier ou des tâches précises

Cost control

Anthropic

On-Demand and Batch pricing

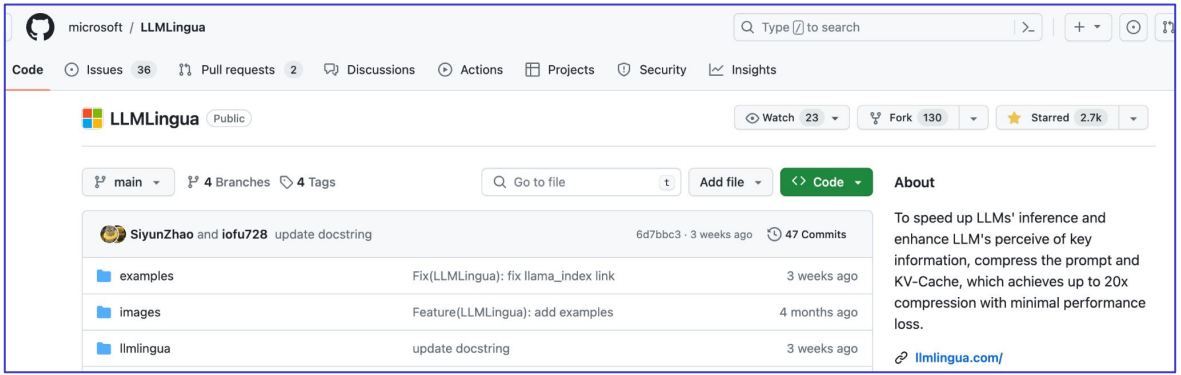
Region: US East (N. Virginia) and US West (Oregon)

Anthropic models	Price per 1,000 input tokens	Price per 1,000 output tokens
Claude Instant	$0.00080	$0.00240
Claude	$0.00800	$0.02400

Cost control

LLMLingua

- ➔ Using a compact, well-trained language model (e.g., GPT2-small) to identify and remove non-essential tokens
- ➔ Compressing prompts up to 20x with minimal performance loss



🔗 https://github.com/microsoft/LLMLingua

Deep dive

Generating audio with Amazon Polly

```
const synthesize = async (data: { topics: string; summary: string[] }) => {
  const audioBuffers = [];
  for (const sentence of data.summary) {
    const sentenceWithBreak = `${sentence} <break strength="x-strong" />`;

    const paragraphBuffers = await Promise.all(
      chunkString(sentenceWithBreak, 1500).map((chunk) => {
        return polly
          .send(
            new SynthesizeSpeechCommand({
              OutputFormat: "mp3",
              TextType: "ssml",
              Text: `<speak>${chunk}</speak>`,
              Engine: "neural",
              VoiceId: "Joanna",
              LanguageCode: "en-US",
            })
          )
          .then((data) => data.AudioStream.transformToByteArray())
          .then((byteArray) => Buffer.from(byteArray));
      })
    );
    audioBuffers.push(...paragraphBuffers);
  }
```

- Processing chunks in “parallel”
- Using **SSML** format to control pauses

Deep diving

Generating audio with language detection

```
const synthesize = async (paragraphs: string[], iso2Lang: string) => {
    const audioBuffers: AudioStream[] = [];
    const langConfig = getLangConfigurationOrDefault(iso2Lang);

    //...
```

```
const iso2LangToPollyParams: {
    [iso2Lang: string]: {
      langCode: string;
      voiceId: string;
      engine: "neural" | "standard";
    };
  } = {
    fr: {
      engine: "neural",
      voiceId: "Lea",
      langCode: "fr-FR",
    },
    en: {
      engine: "neural",
      voiceId: "Joanna",
      langCode: "en-US",
    },
    es: {
      engine: "neural",
      langCode: "es-ES",
      voiceId: "Lucia",
    },
```

~ Language detection can be performed during the summarization task