Automatic Translations

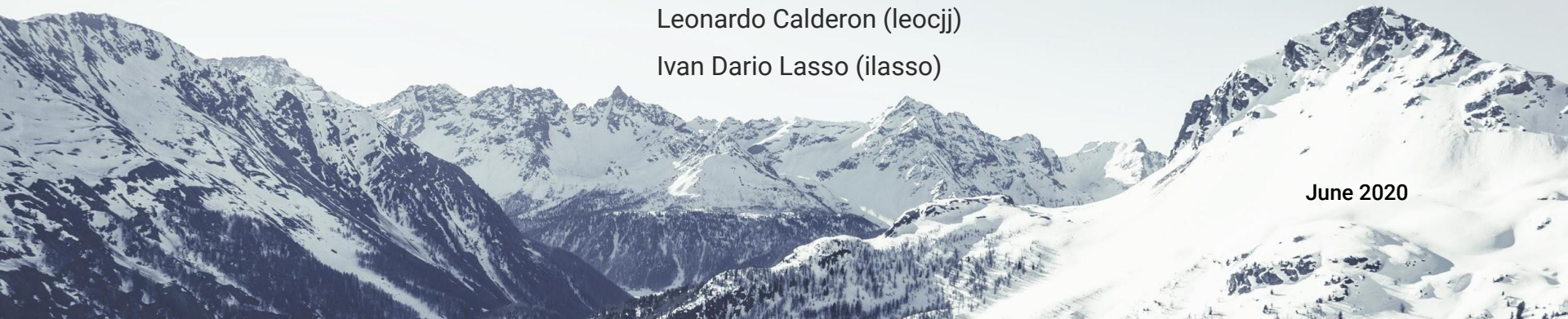
Skillshare

Zombieland CLI Team (cohort 10)

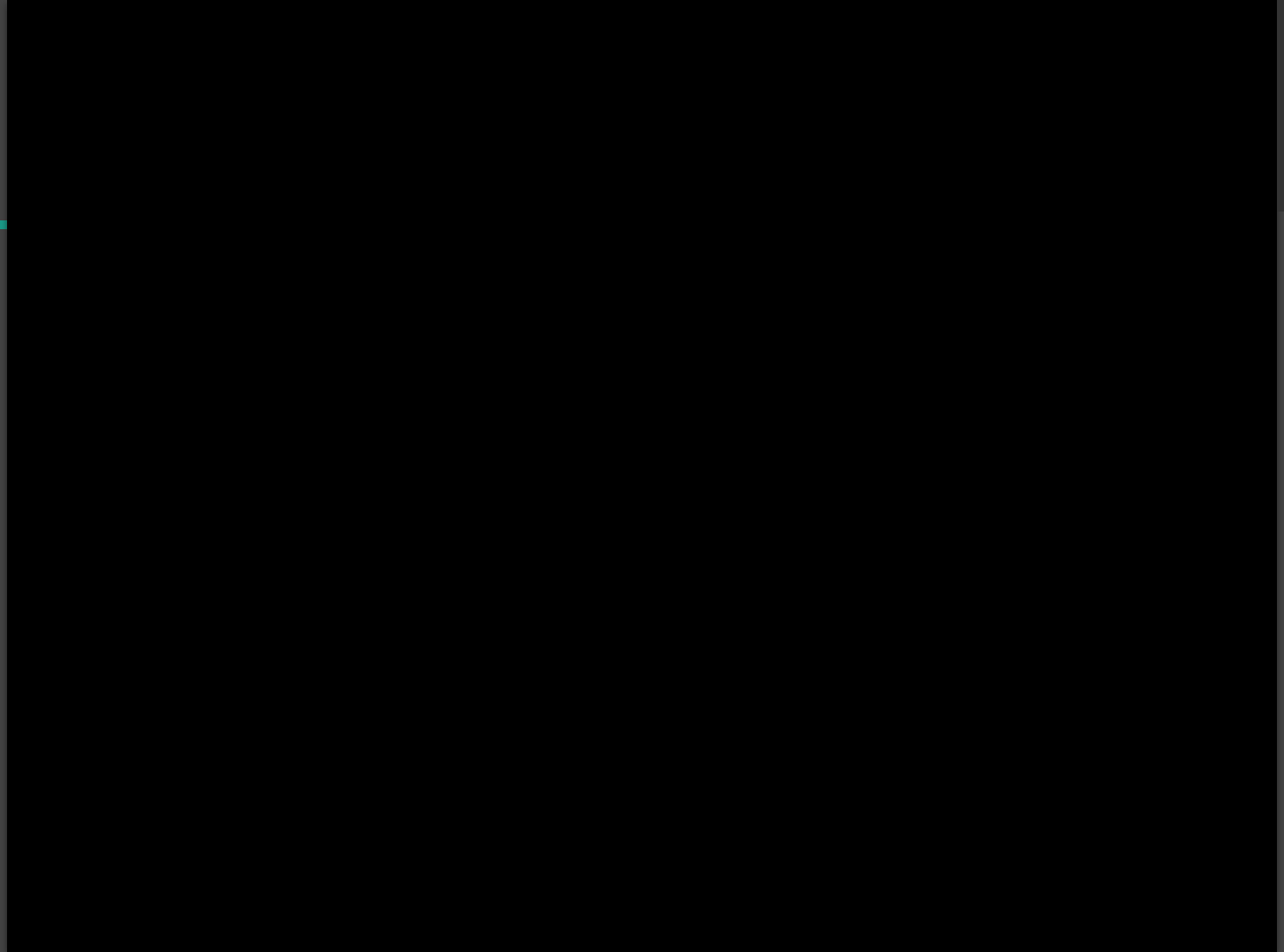
Carlos Andres Garcia (agzsoftsi)

Leonardo Calderon (leocjj)

Ivan Dario Lasso (ilasso)



June 2020


ZCLI
Translate
transcription

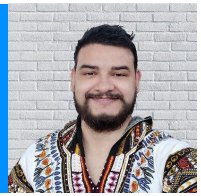
Overview

0. Team members/roles
1. Company
2. The inspiration for the project
3. Technologies & Architecture
4. Core algorithms/code snippet
5. Process, collaboration, and timeline
6. Main challenge overcome
7. Live Demo (front-end, back-end)
8. Learning lessons
9. Questions session

Team and roles

Carlos A. García M., SWE
@karlgarmor agzsoftsi

FrontEnd and Design:
HTML, CSS, Javascript,
AWS CloudFront,
AWS S3



Leonardo Calderon J., SWE
@leocj leocjj

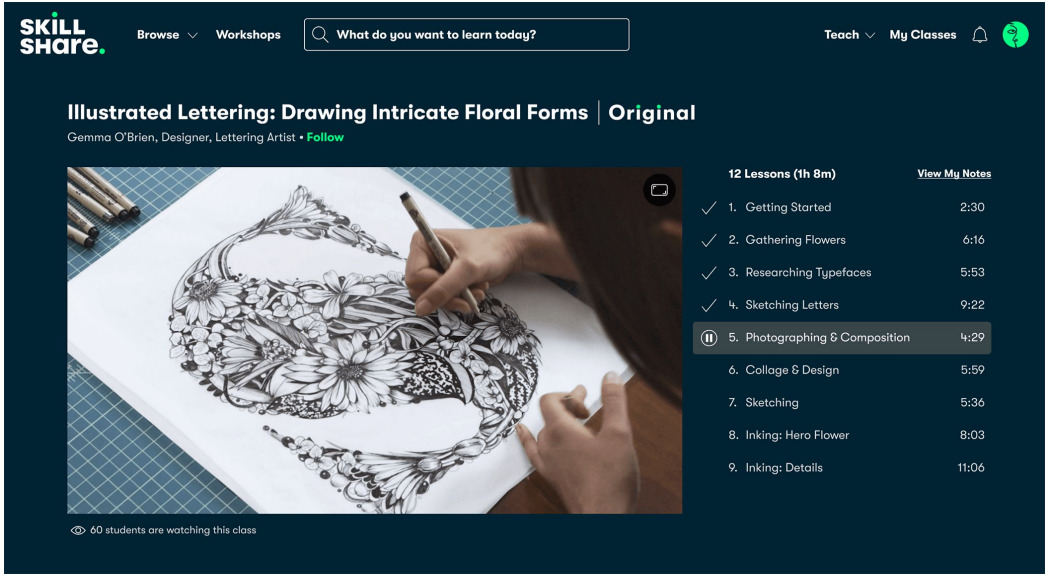
BackEnd:
Python, AWS Lambda,
AWS API Gateway,
AWS S3
AWS Translate

Ivan D. Lasso G., SWE
@ilasso ilasso

FrontEnd and Testing:
HTML, CSS, Javascript,
AWS CloudFront,
AWS S3

Skillshare

- Online learning community
- **Learning from educational videos**
- Available through subscription
- Focus on interaction,
  with the primary goal of
  learning by completing a project.



The inspiration for the project

1. Skillshare wants to go beyond English spoken users.
2. Videos and transcriptions are in English.

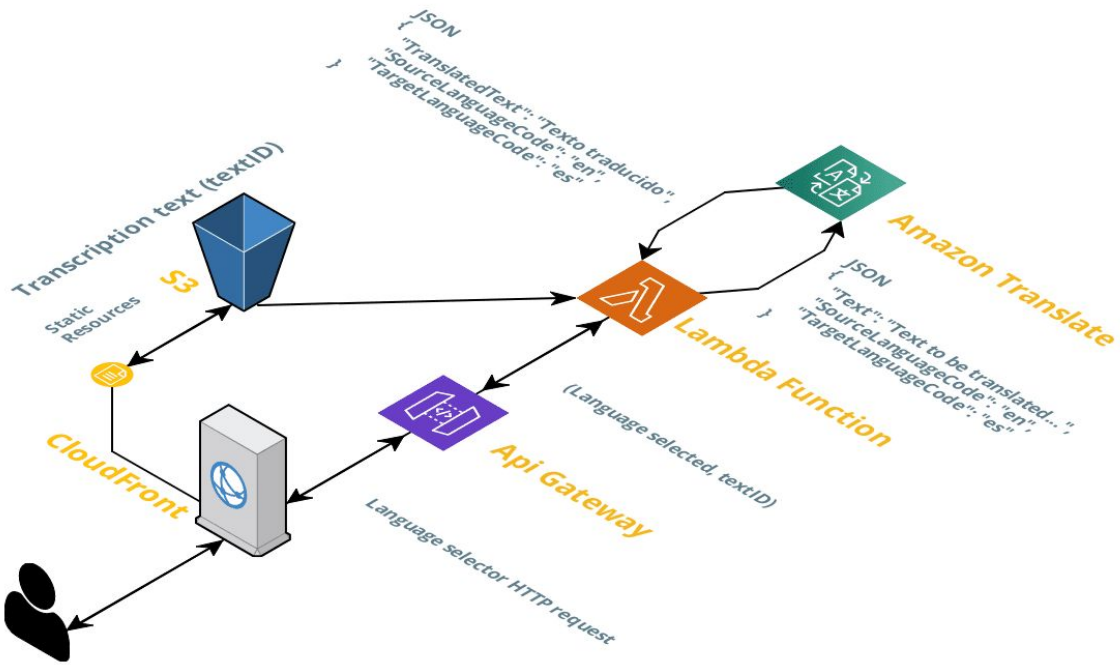
1. Online translation for transcriptions in several languages.
2. Automatic process without affecting user experience.

Technologies & Architecture

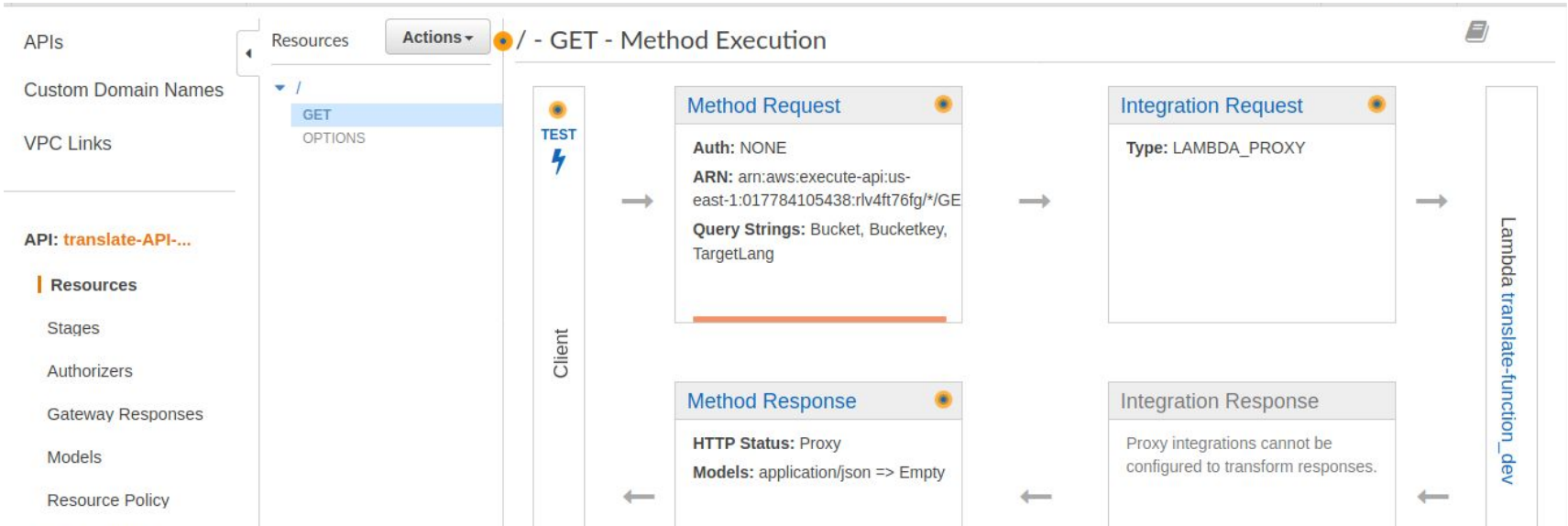
AWS - Amazon Web Services:

- Lambda function (Python)
- S3 objects
- API Gateway
- Translate
- Comprehend

JSON format



Core algorithms/code snippet



lambda_function.py

```
from boto3 import client
from json import dumps
from os import environ

def lambda_handler(event, context):
    '''
    This function is called from an API Gateway (translate-API),
    receive three parameters inside an array to get a translation.
    '''

    key = event['queryStringParameters']['Bucketkey']
    targetLang = event['queryStringParameters']['TargetLang']

    return lambdaFunction(key, targetLang)
```

```
def lambdaFunction(key, targetLang):
    resp['headers']['targetKey'] = key[:key.rfind('.')] + '_' + targetLang + '.srt'


    ''' IF ORIGINAL TRANSCRIPTION STORED IN S3 IS NOT FOUND, RETURN 404'''
    if not keyExist(key):
        resp['body'] = dumps('Video transcription not found')
        return resp

    '''GET STORED TRANSLATION, IF NOT, TRANSLATE TRANSCRIPTION AND STORE IT'''
    if keyExist(resp['headers']['targetKey']):
        translationFile = s3Client.get_object(Bucket=bucket, Key=resp['headers']['targetKey'])
        resp['body'] = translationFile['Body'].read().decode("utf-8")
        resp['statusCode'] = 200  # Ok
        return resp
    else:
        # To get S3 object with original transcription
        response = s3Client.get_object(Bucket=bucket, Key=key)
        text = str(response['Body'].read().decode('utf-8'))

        # Detect source language by sending first 100 characters to check
        responseLanguage = comprehendClient.detect_dominant_language(Text=text[:100])
        if responseLanguage['Languages'][0]['Score'] <= languageThreshold:
            resp['statusCode'] = 412     # Precondition Failed
            resp['body'] = dumps('Video language not soported')
            return resp
        sourceLang = responseLanguage['Languages'][0]['LanguageCode']
```

lambda_function.py

```
def translateBatches(sourceLang, targetLang, text):
        ''' Translate text in batches of step size but with cutting rows of file '''
        lenn = len(text)
        if lenn <= step:
            try:
                translatedText = translateClient.translate_text(  Text = text,
                                                                  SourceLanguageCode = sourceLang,
                                                                  TargetLanguageCode = targetLang
                                                                )
        except translateClient.exceptions.ClientError as e:
            if e.response['Error']['Code'] == "500":
                resp['statusCode'] = 500   # Internal error
            return None
        return translatedText.get('TranslatedText')
```

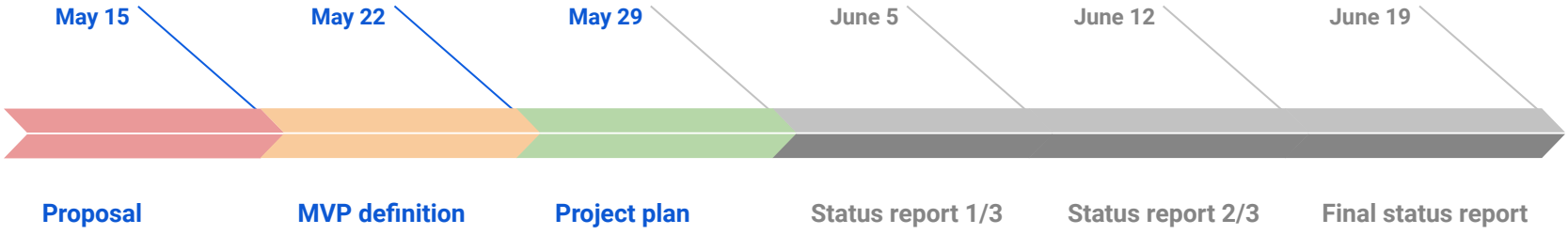
Scrum based process

- The Sprint: weekly, Friday to Thursday
- Sprint Planning: each friday
- Daily Scrum Mo-Th, 11 am.
- Sprint Review: each thursday
- Sprint Retrospective: each friday

Collaboration

- All the teamwork was performed remotely
- Individual accounts for Skillshare and AWS
- Development repository in GitHub
- Internal team rules:
  - Daily meeting with the Client representative.
  - Two daily team meetings
    - To share individual research
    - To do pair programming to integrate codes.
  - Special rule to avoid migration to production code without the approval of all the team.

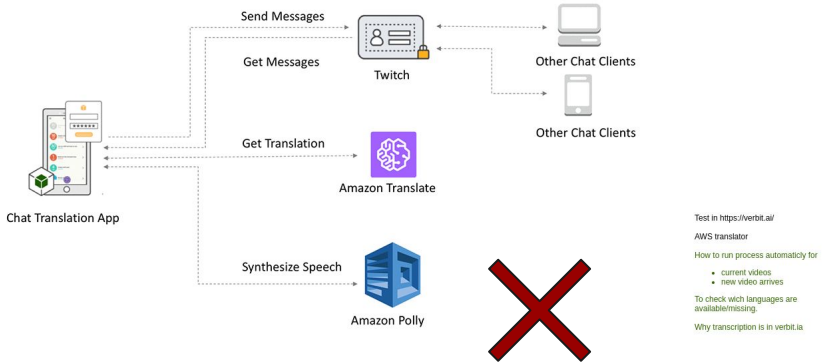
Timeline

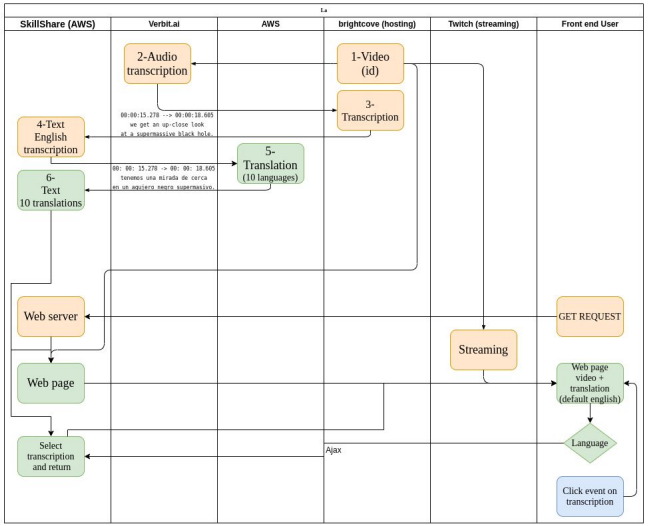


Main challenge overcome

Initial architecture definition has struggled because of technical details not well known by the client representative.

However, his trust in our decisions helped.





Live demo

Landing page:

https://transcriptions-translations.s3.amazonaws.com/landing/index.html

MVP:

https://transcriptions-translations.s3.amazonaws.com/index.html

API test:

https://ytuln8zsz1.execute-api.us-east-1.amazonaws.com/translatetext?TargetLang=es&Bucketkey=formats/video1t.srt

Learning lessons

01 | To define architecture with an extended team.
02 | Scrum helps to organize, prioritize, and develop.
03 | Well defined team rules help during the journey.
04 | To use the right technology rather than use “the best”.
05 | We are capable to learn what we need.

Thank you.

Holberton School
Silvia Guzmán
Johanna Alzate
Laura C. Sánchez
Andrés Barreto
Cohort 10 Cali peers

We made it, keep moving.