Technical Report

Parallel, Distributed, Grid, and Cloud Computing

Rounak Nayee and Nischay Mamidi

Revision 1.2

Introduction

SpeakEasy is a serverless computing application that provides transcription services in response to user-submitted audio files received from WhatsApp users. Built on the AWS cloud platform, SpeakEasy leverages a number of technologies to deliver high-quality, accurate transcription and recognition services to its users.

SpeakEasy uses AWS Lambdas to execute its code, providing a serverless architecture that scales dynamically to handle incoming requests. Incoming files are stored in Amazon S3, a highly scalable and durable object storage service that provides easy access to the files for processing. SpeakEasy uses DynamoDB, a fully managed NoSQL database service, to store metadata and other information about the files, providing fast and efficient access to data required for processing.

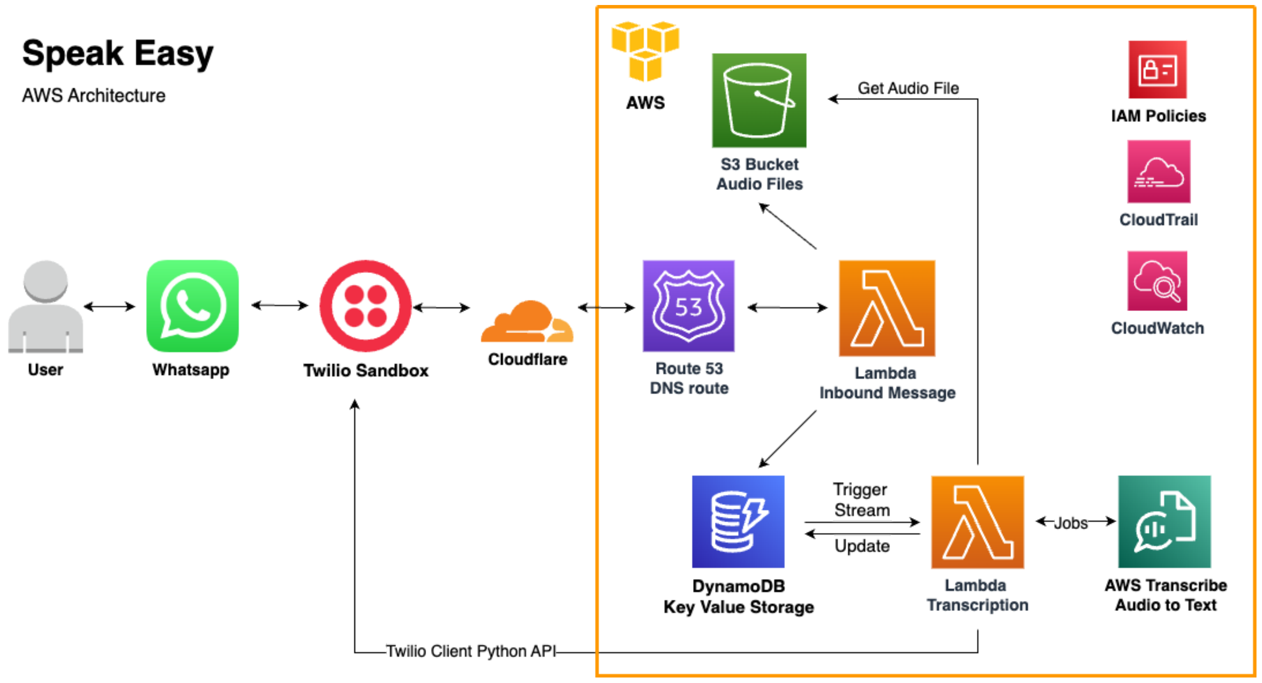
To perform transcription, SpeakEasy uses Amazon Transcribe, a powerful machine learning service provided by AWS. Amazon Transcribe provides accurate speech-to-text conversion. These services are used to provide high-quality transcriptions to users.

To monitor and manage the system, SpeakEasy uses CloudTrail and CloudWatch. CloudTrail provides detailed logs of API activity and events, while CloudWatch provides monitoring and alerting capabilities, allowing SpeakEasy to detect and respond to issues in real-time.

Finally, SpeakEasy uses Cloudflare DNS combined with Amazon Route 53 to manage DNS records and ensure reliable and fast access to the application.

Overall, SpeakEasy provides a powerful and scalable solution for the transcription needs of WhatsApp users. Leveraging the power of AWS, SpeakEasy provides high-quality results in real-time, while remaining cost-effective and scalable.

Architecture Diagram



Design Philosophy

Our serverless application uses Cloudflare, Amazon Web Services, and Twilio for providing transcription services.

- Stateless Functions: The application is designed using stateless functions that are triggered by events. When a user shares an audio file on WhatsApp, a Lambda function is triggered to initiate the transcription process.
- Scalability: The application is designed to scale seamlessly with increasing loads. AWS Lambda functions are used for computing infrastructure, which can be easily scaled up or down based on demand. The application also uses auto-scaling capabilities provided by the AWS platform to handle spikes in traffic.
- Third-Party Services: Twilio is used to provide an interface on WhatsApp where users can share their audio recordings. Cloudflare is used as an Authoritative DNS server, providing reliable and secure domain name resolution. Both services help to reduce the development time and cost.
- Event-Driven Architecture: The SpeakEasy application is designed to be event-driven, where functions are triggered by events rather than being constantly active. This helps to reduce the overall cost of running the application by minimizing idle resources.

- Microservices: The application is broken down into smaller, independent microservices that can be deployed and managed separately. For example, the transcription service is provided by Amazon Transcribe and works independently of the other services.

- Security: The application implements security best practices such as encrypting data in transit and at rest, and implementing access controls to ensure that only authorized users can access the application and its resources. The application also uses AWS IAM to manage access to resources. Additionally, Twilio messages being sent by the user are validated and matched with a signature before they are processed.

- Monitoring and Logging: The SpeakEasy application implements robust monitoring and logging mechanisms to ensure that issues can be tracked and troubleshooted. AWS CloudWatch is used to monitor Lambda functions, and Cloudflare provides analytics and logging capabilities to monitor traffic to the application.

- Cost Optimization: The SpeakEasy application is designed to optimize cost, for example, by using serverless architectures to pay only for the resources used. The application also uses AWS Cost Explorer to monitor and optimize cost.

Implementation

1. The user sends a transcription job (audio file) via WhatsApp to our WhatsApp business account named SpeakEasy
2. Twilio forwards the AWS Lambda endpoints which are protected by Cloudflare CDN
3. The Lambda function verifies the signature of the message received before processing it further.
4. Once verified the function uploads the audio file to an S3 bucket. The entire process implements encryption to protect the user’s privacy.
5. The function also stores any metadata including the user phone number and date the transcription request was received in a DynamoDB table.
6. The DynamoDB table streams the newly added record and launches a new lambda function.
7. The lambda function submits a transcription job to AWS Transcribe and waits for the service to send a callback with the output
8. If the transcription is successful, the lambda function stores the transcribed text in the DynamoDB and sends a reply back to the Twilio server with the text.

The entire process does not use any EC2 instance and is scalable, designed to handle any amount of traffic sent by the users, provided they are within the configured AWS limits.

Since transcription jobs are independent of each other, multiple concurrent requests can be processed at the same time.

Deployment

To deploy the Terraform repo for SpeakEasy onto an AWS platform, follow these steps:

1. Clone the repository onto your local machine using the following command:

```
git clone https://github.com/Rounaknayee/speakeasy.git
```

2. Ensure that Terraform is installed on your machine. If it is not installed, download the latest version from the official website.
3. Navigate to the `speakeasy/terraform` directory in the cloned repository.
4. Open the `terraform.tfvars` file and enter appropriate values for the variables. This includes your AWS access key and secret access key, region, and any other necessary configuration settings.
5. Run the following command to initialize the Terraform configuration:

```
terraform init
```

6. Run the following command to preview the changes that will be made:

```
terraform plan
```

7. If the preview looks good, run the following command to apply the changes:

```
terraform apply
```

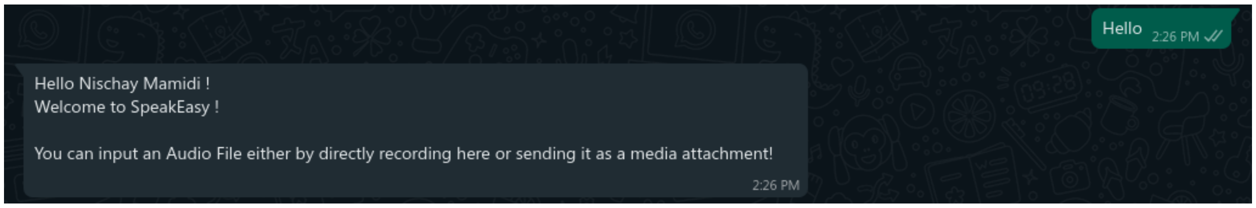
8. Confirm the changes when prompted by typing `yes`.
9. Terraform will create the necessary resources on your AWS platform. Once completed, you will see a message indicating that the deployment was successful.
10. Export the AWS environment keys using the following commands:

```
export AWS_ACCESS_KEY_ID="YOUR_ACCESS_KEY"
export AWS_REGION=”YOUR_AWS_REGION”
export AWS_SECRET_ACCESS_KEY="YOUR_SECRET_ACCESS_KEY"
```

You can now access SpeakEasy on your AWS platform and begin using the transcription service.

Note: Make sure to keep your AWS access key and secret access key secure and not share them with anyone.

Screenshots

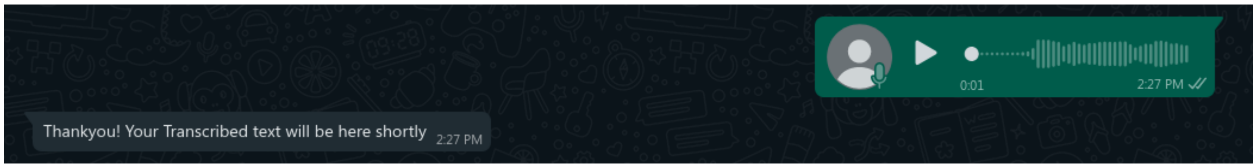


Hello 2:26 PM

Hello Nischay Mamidi !
Welcome to SpeakEasy !

You can input an Audio File either by directly recording here or sending it as a media attachment!
2:26 PM

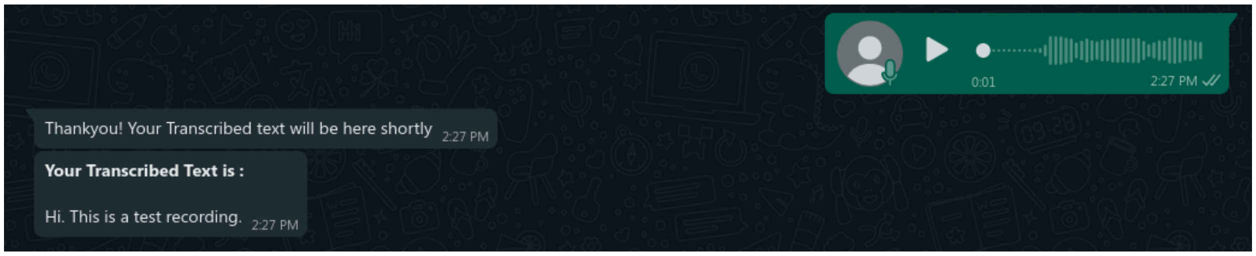
Chatting for the first time



0:01 2:27 PM

Thankyou! Your Transcribed text will be here shortly 2:27 PM

Sending an audio recording for transcription



0:01 2:27 PM

Thankyou! Your Transcribed text will be here shortly 2:27 PM

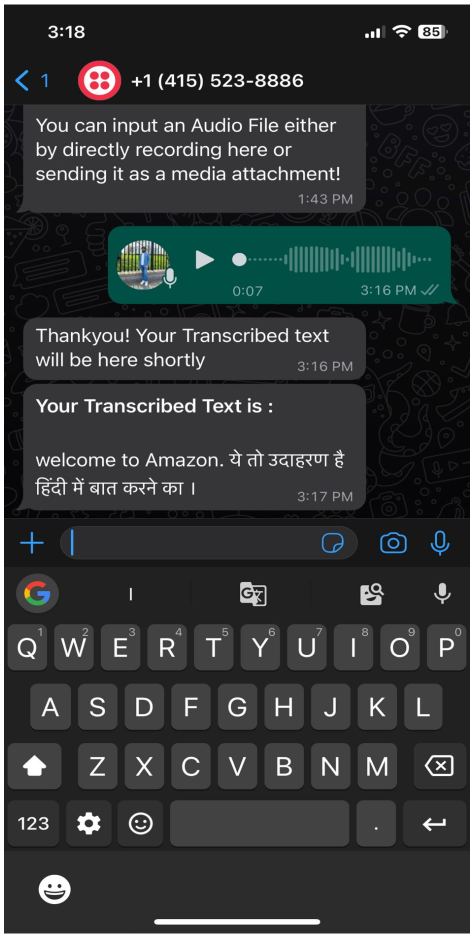
Your Transcribed Text is :

Hi. This is a test recording. 2:27 PM

Response back with the transcribed text.

Source Audio: Hi. This is a test recording

Transcribed Audio: Hi. This is a test recording


3:18

+1 (415) 523-8886

You can input an Audio File either by directly recording here or sending it as a media attachment!
1:43 PM

0:07 3:16 PM

Thankyou! Your Transcribed text will be here shortly 3:16 PM

Your Transcribed Text is :

welcome to Amazon. ये तो उदाहरण है हिंदी में बात करने का ।
3:17 PM

Response back with the transcribed audio showcasing multilingual capability

Source Audio: Welcome to Amazon. [This is an example to speak in Hindi] (Text in Hindi)

Transcribed Audio: Welcome to Amazon. [This is an example to speak in Hindi] (Text in Hindi)

Debugging (SpeakEasy)

SpeakEasy uses CloudWatch and CloudTrail to help in debugging issues related to instance deployment. CloudWatch is a monitoring service that provides log aggregation, metrics, and alarms for AWS resources and applications, while CloudTrail provides a record of all API calls made to AWS resources within an account.

When SpeakEasy deploys an instance, it creates various AWS resources such as AWS Lambda functions, S3 buckets, and DynamoDB tables. CloudWatch is used to monitor the logs generated by these resources. SpeakEasy logs its activities to CloudWatch logs prefixed with `speakeasy-`. This prefix helps to identify the logs related to SpeakEasy activities

To access the CloudWatch logs for SpeakEasy, follow these steps:

1. Log in to the AWS Management Console and navigate to the CloudWatch service.
2. In the CloudWatch console, click on the "Logs" menu item in the left-hand menu.
3. In the "Log groups" section, locate the log group prefixed with `speakeasy-` that corresponds to the instance you want to debug.
4. Click on the log group to open the log stream.
5. In the log stream, you can view the logs generated by the instance.

.

Common Error Messages and Troubleshooting

Media file required

Solution:

1. Send only media messages to the WhatsApp server

Only one Media file allowed!

Solution:

1. Send only one media message. Do not send multiple media files in the same message.

Only Audio File Allowed

Solution:

1. Only audio files of the file type ogg are allowed for transcription. Please use the official WhatsApp audio recorder to send these messages.

Exception in uploading file to s3 and lambda

Solution:

1. Check file permissions: Ensure that the file being uploaded has the necessary read permissions and is not currently in use by another process. Make sure that the user or role has the necessary permissions to upload files to the S3 bucket and invoke the Lambda function.

2. Verify IAM roles: Check that the AWS Identity and Access Management (IAM) roles associated with the S3 bucket and Lambda function are correctly configured. Ensure that the role has the necessary permissions to interact with the S3 bucket and trigger the Lambda function.

3. Check network connectivity: Confirm that there are no network connectivity issues between the S3 bucket and the Lambda function. Check that the relevant security groups and network access control lists (ACLs) are correctly configured to allow traffic between the two services.

4. Check AWS service limits: Ensure that you have not exceeded any service limits, such as the number of requests per second, which could cause the upload to fail. Consider adjusting the limits if necessary or contacting AWS support for assistance.

5. Debug the Lambda function: Check the Lambda function code for errors, as a malfunctioning function can cause exceptions during the upload process. Use CloudWatch logs to troubleshoot any errors and isolate the root cause of the issue.

Error sending transcription job

Solution:

1. Verify AWS service limits: Ensure that you have not exceeded any AWS service limits related to Transcribe, such as the number of concurrent transcription jobs allowed. Consider adjusting the limits if necessary or contacting AWS support for assistance.

2. Check file format and size: Ensure that the file being transcribed is in a supported format, such as OGG, and within the maximum size limit for the Transcribe service.

3. Check the input file path: Confirm that the input file path specified in the AWS Transcribe job configuration is correct and that the file can be accessed by the Transcribe service. Check if the S3 bucket URL upload path is correct and accessible by my Lambda function

4. Debug AWS Transcribe: Use CloudWatch logs to troubleshoot any errors in the Transcribe service and isolate the root cause of the issue.

Error getting transcription result

Solution:

1. Check for errors in the transcription job: Use the Amazon Transcribe console or the AWS CLI to view the status of the transcription job and any related error messages. Debug any issues or inconsistencies found in the transcription process.

2. Monitor CloudWatch logs: Use CloudWatch logs to identify any error messages related to AWS Transcribe and the Lambda function that is invoking the service.

Error saving to DynamoDB

Solution:

1. Check IAM roles: Verify that the IAM role associated with the Lambda function has the necessary permissions to access and write to the DynamoDB table. Ensure that the role has the appropriate permissions to interact with the table and write data.
2. Verify table configuration: Confirm that the DynamoDB table is correctly configured and has the correct schema. Check that the table has the necessary write capacity and is not currently undergoing any operations that may be causing errors.

3. Debug the Lambda function: Check the Lambda function code for any errors that may be causing the "Error saving to DynamoDB" message. Use CloudWatch logs to troubleshoot any errors and isolate the root cause of the issue.

Future Scope

Support for Video Transcription

In the future, we plan on expanding the scope of our application to include support for video transcription services as well. This will enable our users to transcribe and extract meaningful information from the video files they send through WhatsApp.

We plan on leveraging the power of AWS services, such as Amazon Transcribe, to enable video transcription capabilities within SpeakEasy.

With this feature, users can easily transcribe and search the contents of their video files, making it easier to find and analyze critical information. This will also help to improve accessibility for users with hearing impairments or those who prefer written transcripts of audio and video content.

Support for Image OCR

In the future, we plan on expanding SpeakEasy's capabilities to include support for Optical Character Recognition (OCR) and image recognition services when sending image files via WhatsApp. This will allow our users to extract text and information from the images they send through the platform, making it easier to organize and analyze their data.

As part of our plan to add support for OCR and image recognition services in SpeakEasy, we plan on using Amazon Textract as our primary OCR service.

Amazon Textract is a machine learning service that can automatically extract text and data from scanned documents, PDFs, and images. By integrating this service with SpeakEasy, users will be able to extract text from images they send through WhatsApp and convert them into editable text.

Fully Featured User Dashboard

In the future, we plan on expanding SpeakEasy's capabilities by providing users with a dashboard where they can log in and access their previous transcription jobs. This dashboard will allow users to easily view and download their audio files and transcription records, giving them more control over their data and making it easier to track their transcription history

The dashboard will be accessible through a secure login process, where users will need to provide their credentials to access their account. Once logged in, users will be able to view a list of their previous transcription jobs, including the date, time, and status of each job. They will also be able to download the audio file and transcription record for each job, giving them complete access to their data.

By providing users with a dashboard, we aim to make it easier for them to manage and access their transcription data. This will be particularly useful for users who need to keep track of multiple transcription jobs, such as professionals who conduct interviews or record meetings.

Version History

Revision 1.2

- Added detailed description for architecture implementation

Revision 1.1

- Added Future Scope to the document

Revision 1.0

- Initial draft