Software Requirements Specification

for

Text-to-Speech Converter

Version 1.0 approved

Prepared by <author>

<organization>

<date created>

.

Table of Contents

Revision History

Name	Date	Reason For Changes	Version

1. Introduction

TTS converter is a serverless application hosted on amazon public cloud that uses Amazon Polly to convert text to speech. The application has a simple user interface that accepts text in many different languages and then converts it to audio files which can be played from a web browser.

1.1 Purpose

The purpose of this document is to describe the technical architecture and the functional component of TTS converter application system The document also describes the options/approaches considered during design along with reasoning on the selected option wherever applicable. The key purposes fulfilled by this document are :

- To provide an insight into the architecture of the TTS system.
- To capture key architectural decisions that were made in order to arrive at the final system architecture.

1.2 Intended Audience and Reading Suggestions

Developers working in cloud technologies and NoSQL databases.

1.3 Project Scope

Scope of the document is to represent the application architecture and design implementation. Regulatory requirements concerning process, data, and security/auditing for various geographies/verticals/domains will be incorporated outside the scope of core product development. These requirements are not considered for TTS architecture.

2. Overall Description

2.1 Product Features

Amazon Polly provides speech synthesis functionality that overcomes those challenges, allowing you to focus on building applications that use text-to-speech instead of addressing interpretation challenges.

Amazon Polly turns text into lifelike speech. It lets you create applications that talk naturally, enabling you to build entirely new categories of speech-enabled products. Amazon Polly is an Amazon AI service that uses advanced deep learning technologies to synthesize speech that sounds like a human voice.

2.2 Operating Environment

Software	Critical Drivers
Simple Storage Service	Data Object Storage for storing large files
Simple Notification Service	To send notification
Lambda	To execute adhoc function
DynamoDB	NoSql Database
API gateway	Hosting restful services

.

2.3 Design Implementation and assumptions

Following assumptions are made while specifying the architecture of TTS converter:

- TTS stores the voices data in DynamoDB database managed by amazon cloud. The architecture does not factor in use of an external repository for storing speech data.
- Application connects to database through lambda function.
- No servers to maintain or patch, etc. By default, TTS Converter application is highly available because AWS Lambda, Amazon API Gateway, Amazon S3, and Amazon DynamoDB use multiple Available Zones..

3. System Features

The application provides two methods – one for sending information about a new post, which should be converted into an MP3 file, and one for retrieving information about the post (including a link to the MP3 file stored in an S3 bucket). Both methods are exposed as RESTful web services through Amazon API Gateway.

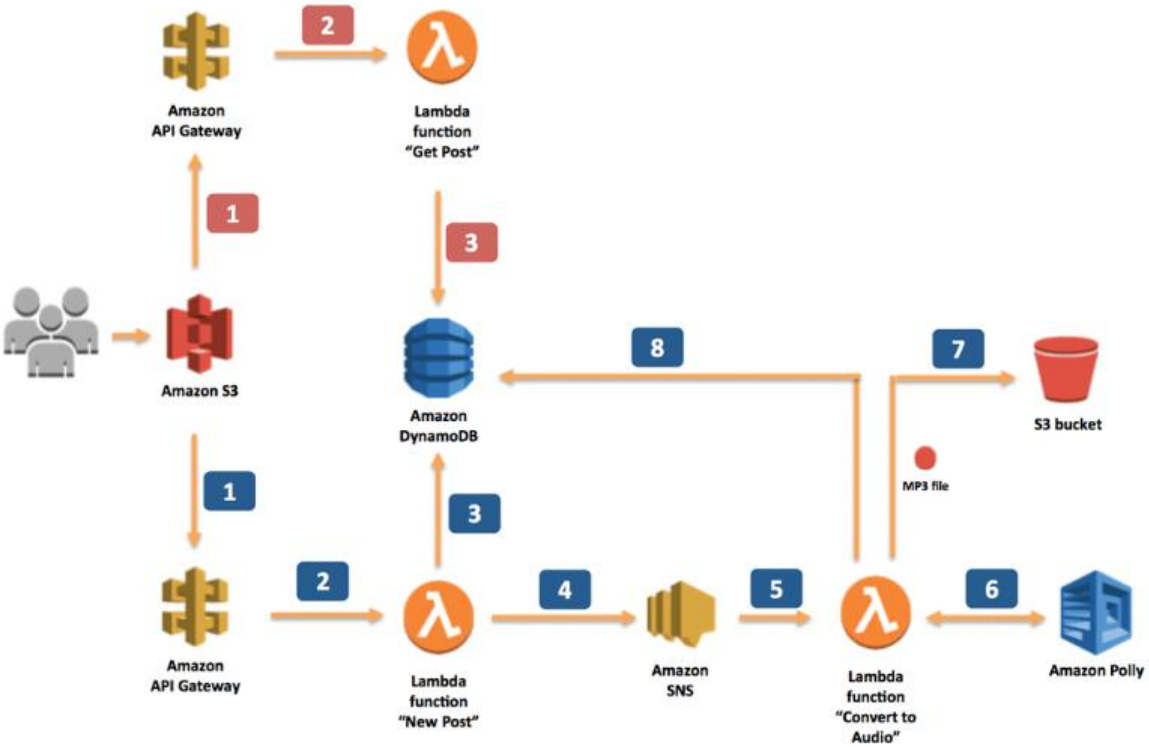
When the application sends information about new posts:

1. The information is received by the RESTful web service exposed by Amazon API Gateway. This web service is invoked by a static webpage hosted on Amazon Simple Storage Service (Amazon S3).
2. Amazon API Gateway sets off a dedicated Lambda function, “New Post,” which is responsible for initializing the process of generating MP3 files.
3. The Lambda function inserts information about the post into a DynamoDB table, where information about all posts is stored.
4. To run the whole process asynchronously, we use Amazon SNS to decouple the process of receiving information about new posts and starting their conversion.

5. Another Lambda function, “Convert to Speech,” is subscribed to SNS topic whenever a new message appears (which means that a new post should be converted into an audio file). This is the trigger.
6. The “Convert to Speech” Lambda function uses Amazon Polly to convert the text into an audio file in the specified language (the same as the language of the text).
7. The new MP3 file is saved in a dedicated S3 bucket.
8. Information about the post is updated in the DynamoDB table. Then, the reference (URL) to the S3 bucket is saved with the previously stored data.

When the application retrieves information about posts:

1. The RESTful web service is deployed using Amazon API Gateway. Amazon API Gateway exposes the method for retrieving information about posts. These methods contain the text of the post and the link to the S3 bucket where the MP3 file is stored. In our scenario, this web service is invoked by a static webpage hosted on Amazon S3.
2. Amazon API Gateway invokes the “Get Post” Lambda function, which deploys the logic for retrieving the post data.
3. The “Get Post” Lambda function retrieves information about the post (including the reference to Amazon S3) from the DynamoDB table.



4. Interface Requirements

4.1 User Interfaces

Application can be accessed from a static webpage hosted on Amazon Simple Storage Service The package for the website contains three files:

- html
- css
- js

4.2 Software Interfaces

Amazon Simple Storage Service (S3)

- S3 bucket will be used to store all audio files created by the application. S3 bucket to be configured from AWS console and it must have unique name as it's a global service
- . A small web page on Amazon S3 hosted as static web pages. This web page uses JavaScript to connect to API gateway and provide text-to-speech conversion functionalities.
- Amazon Polly delivers the consistently fast response times required to support real-time, interactive dialog. There are no additional text-to-speech charges for using the speech. Polly is easy to use. It accepts a text user want to convert into speech to the Amazon Polly API. Amazon Polly immediately returns the audio stream to the application so that user can play it directly from the application or store it in a standard audio file format such as an MP3.

Dynamo DB Uses

Application stores information about posts, including the text and URL for the MP3 file, on DynamoDB. From the DynamoDB console we create a single table, which we call "posts." Our primary key (id) is a string, which the "New Post" Lambda function creates when new records (posts) are inserted into a database.

The columns provide the following information:

- id – The ID of the post
- status – UPDATED or PROCESSING, depending on whether an MP3 file has already been created
- text – The post's text, for which an audio file is being created
- url – A link to an S3 bucket where an audio file is being stored

- voice – The Amazon Polly voice that was used to create audio file.

Amazon Lambda Function:

Two Lambda functions is being used to split the logic of converting a post (text) into an audio file.

- First, it allows application to use asynchronous calls so that the user who sends a new post to the application receives the ID of the new DynamoDB item; The first Lambda function that we create is the entry point for our application. It receives information about new posts that should be converted into audio files.

- With small posts, the process of converting to audio files can take milliseconds, but with bigger posts (100,000 words or more), converting the text can take a bit longer. In other use cases, when we want to do real-time streaming, size isn't a problem, because Amazon Polly starts to stream speech back as soon as the first bytes are available.

- The second Lambda function, which allows a single execution to run as long as 5 minutes. This should be more than enough time to convert posts request.

"New Post" Lambda function

Lambda function does the following:

1. Retrieves two input parameters:
    - Voice – one of dozens of voices that are supported by Amazon Polly
    - Text – the text of the post that we want to convert into an audio file
2. Creates a new record in the DynamoDB table with information about the new post
3. Publishes information about the new post to SNS (the ID of the DynamoDB item/post ID is published there as a message)
4. Returns the ID of the DynamoDB item to the use

"Convert to Audio" Lambda function

Above Lambda function converts text that is stored in a DynamoDB table into an audio file. This Lambda function does the following:

5. Retrieves the ID of the DynamoDB item (post ID) which should be converted into an audio file from the input message (SNS event)

6. Retrieves the item from DynamoDB
7. Converts the text into an audio stream
8. Places the audio (MP3) file into an S3 bucket
9. Updates the DynamoDB table with a reference to the S3 bucket and the new status

Lambda function as a RESTful web service

we need to expose application logic as a RESTful web service using Amazon API Gateway service. it can be invoked easily using a standard HTTP protocol.

Amazon Polly

The synthesize speech method of amazon polly API receives the text that should be converted and the voice that should be used. In return, it provides the audio stream. The catch is that there is a size limit of 1,500 characters on the text that can be provided as input. Because our posts can be big, we need to divide them into blocks of about 1,000 characters, depending where the final word in the block ends. After converting the blocks into an audio stream, we join them together again.