Reaching a large podcast audience can present some significant infrastructure scaling challenges. In this session, startup company Whooshkaa walks you through the podcasting landscape. During this session, you will learn about the new audiences you can reach through podcasts. We will explore technical solutions such as Amazon Lightsail, S3 and CloudFront which can facilitate experimentation and help you reach a global audience at low cost. We will dive into Whooshkaa's podcasting platform and explore advanced architectures, leveraging AWS services, allowing you to curate and customize content for each listener. We will also explore tools and solutions for measuring engagement and connecting with your audience through podcasting.

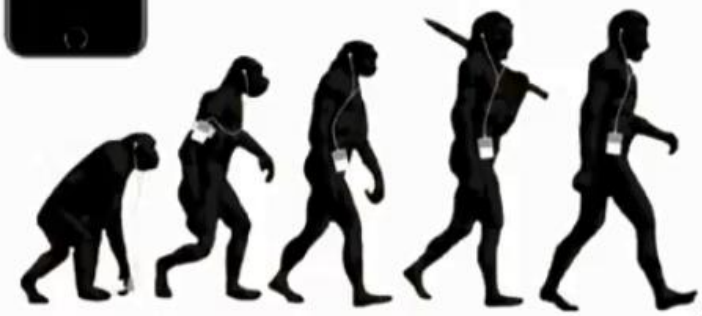
Podcast evolution

2004

Open iTunes
Browse store
Select podcast
Download audio
Sync iPod
Press Play

2017

Open Podcast app
Select and play



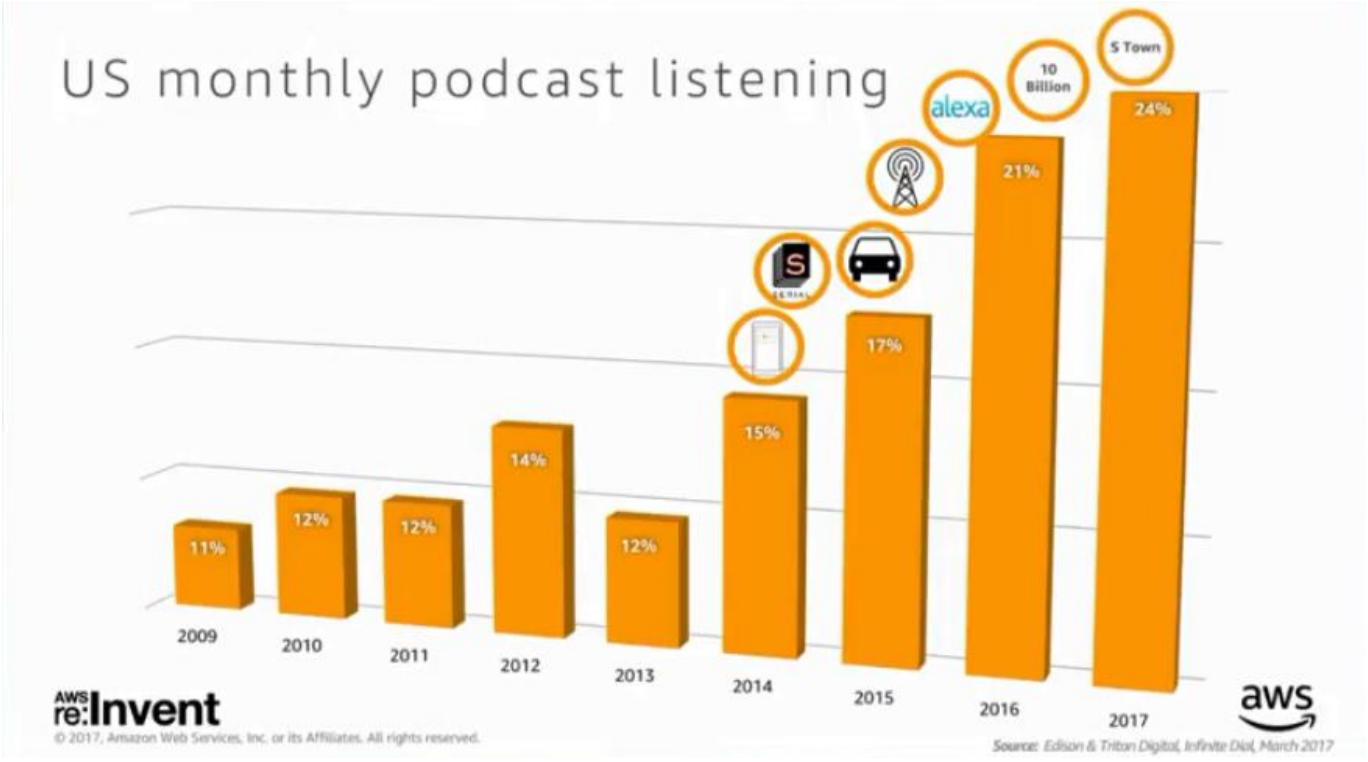
aws re:Invent

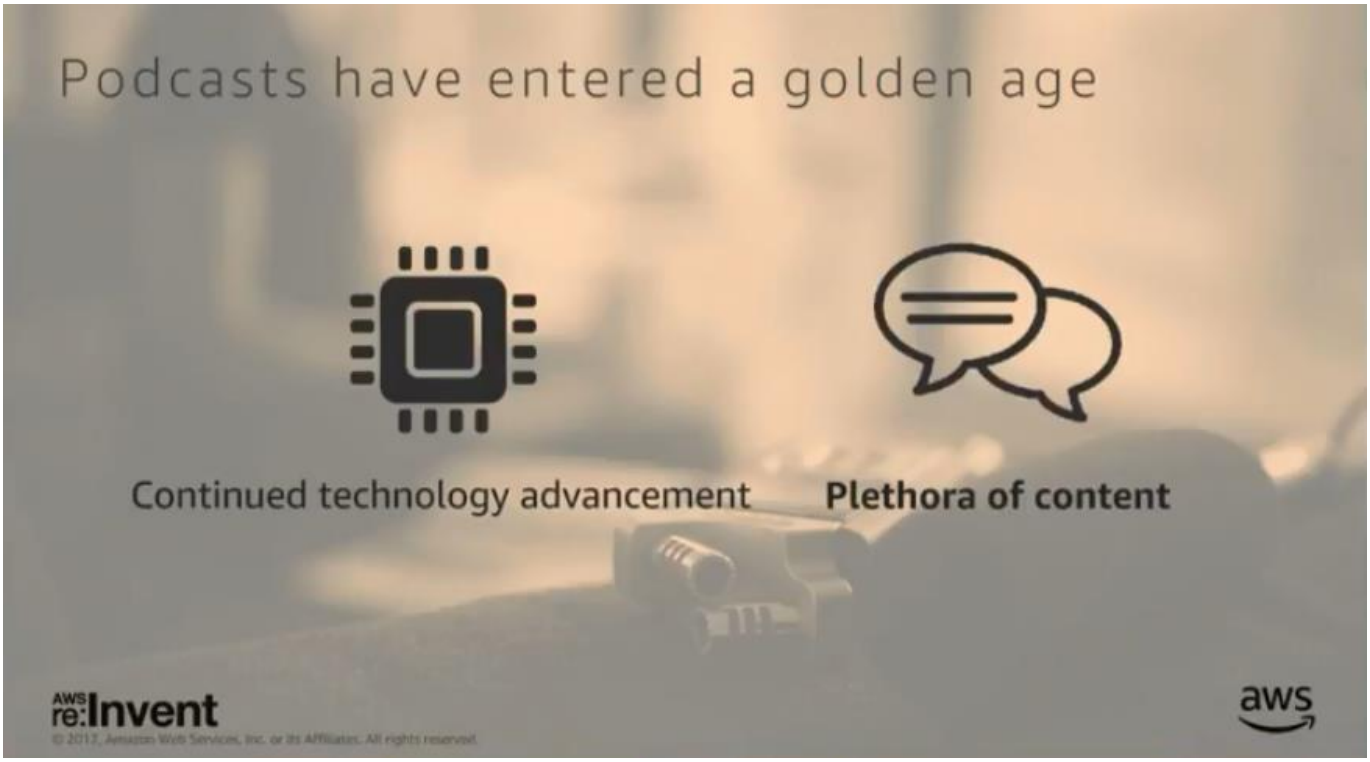
aws

"I want it now" generation

aws re:Invent

aws

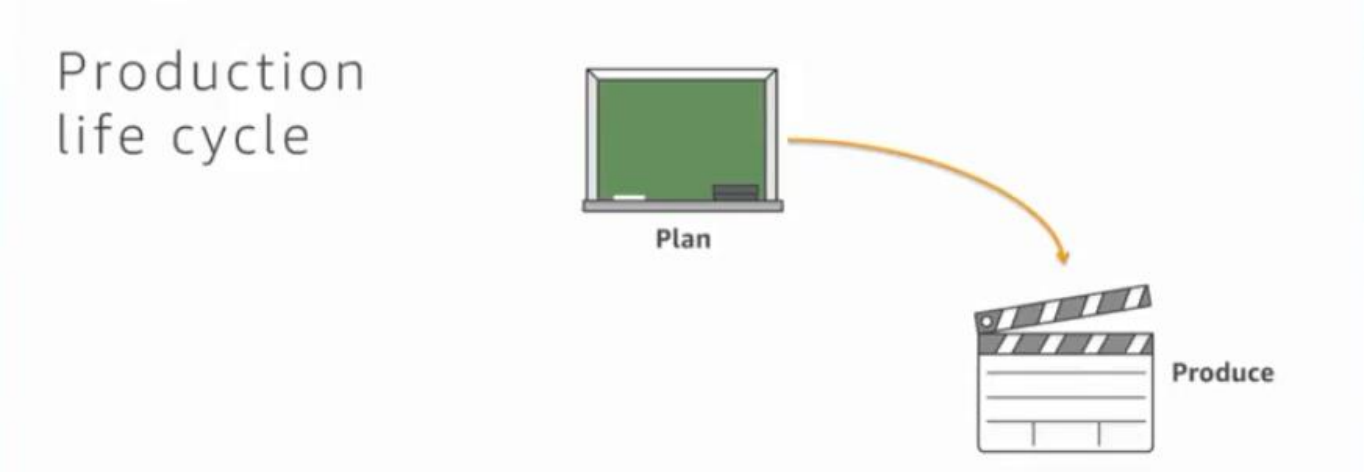

US monthly podcast listening
alexa
10 Billion
S Town
11%
12%
12%
14%
12%
15%
17%
21%
24%
2009
2010
2011
2012
2013
2014
2015
2016
2017
aws re:Invent
© 2017, Amazon Web Services, Inc. or its Affiliates. All rights reserved.
Source: Edison & Triton Digital, Infinite Dial, March 2017
aws


Podcasts have entered a golden age
Continued technology advancement
Plethora of content
aws re:Invent
aws

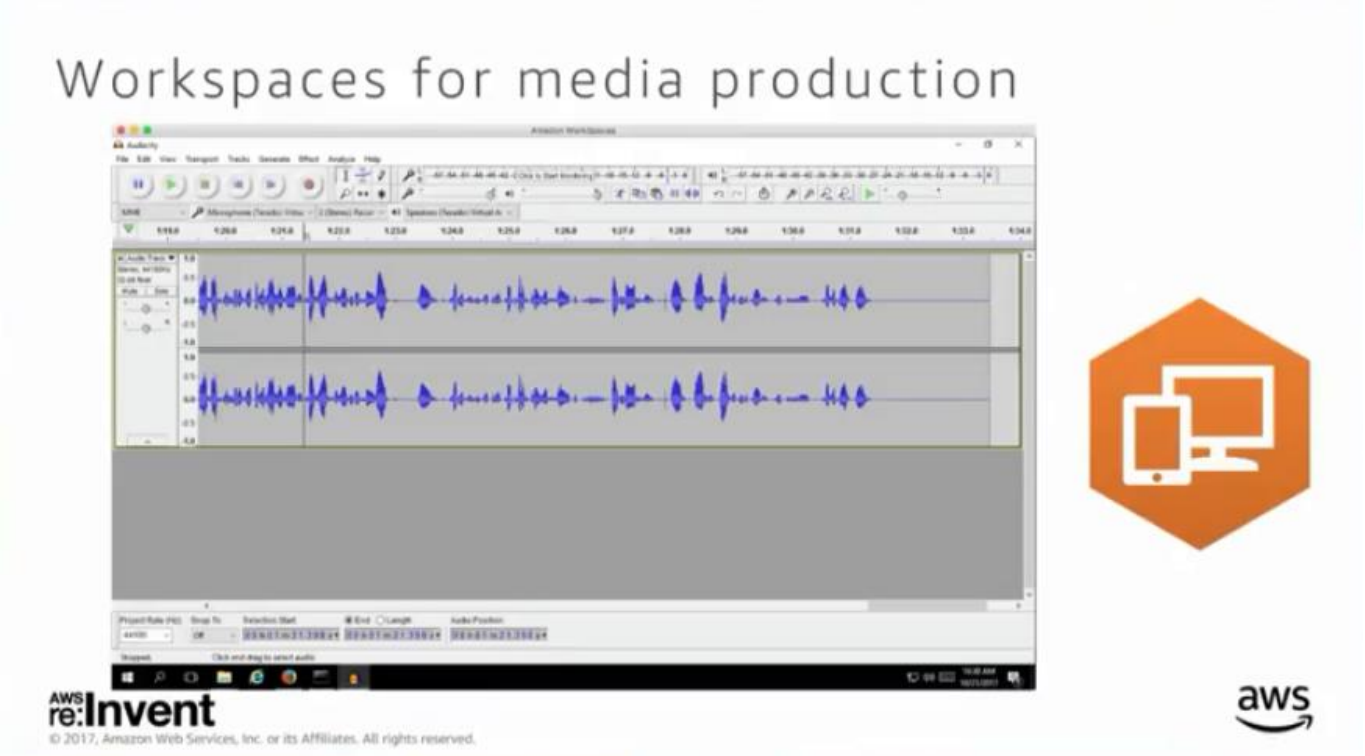

Podcasting on AWS

You need to plan what you are going to say in front of the microphone when creating a podcast

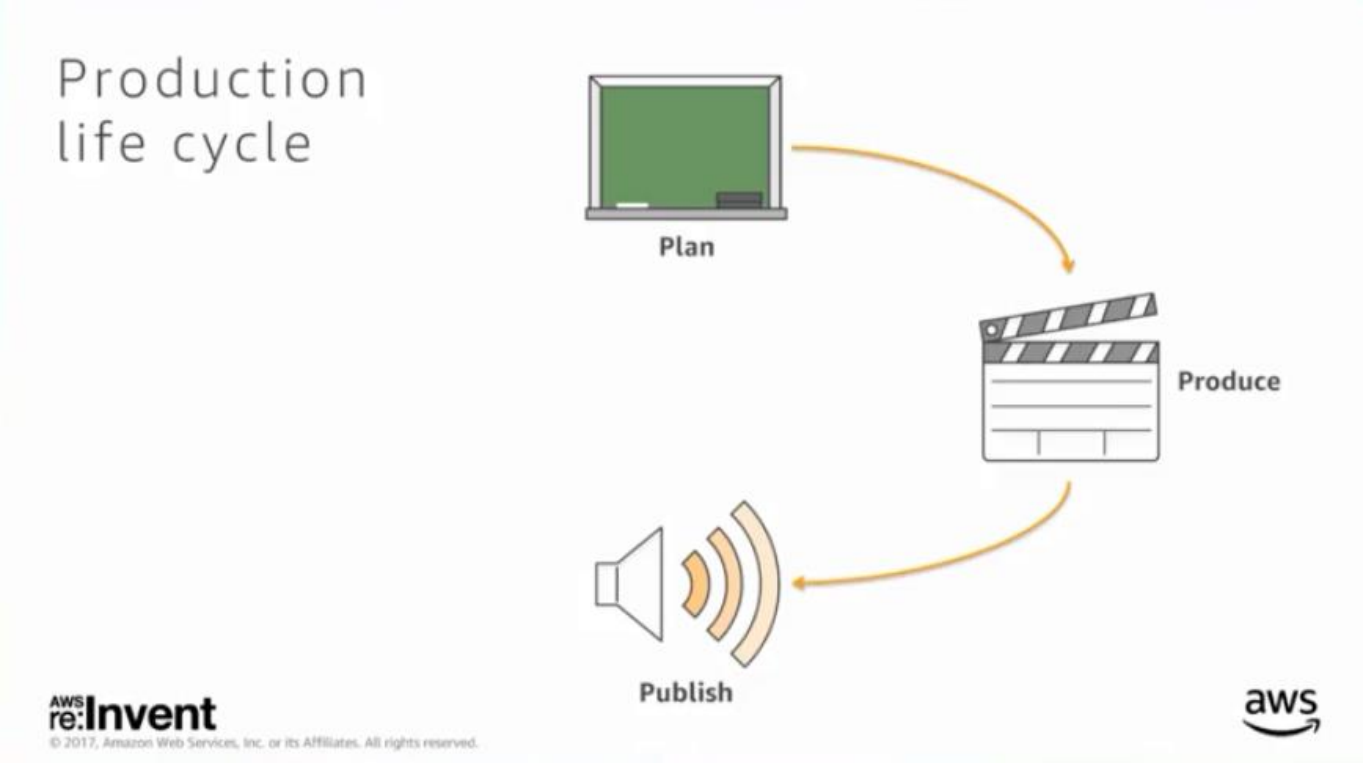
Do your ***homework***, then you need to decide how to present the format, you need a good cover art for publishing your app on the store. You need a theme for your podcast and also pod consistently like every Friday, this is how you build an audience for your podcast. ***Amazon WebDocs*** helps you do your documentation and do version control, ***Trello*** can also be used for organizing what to do.



How do we actually produce the podcast?



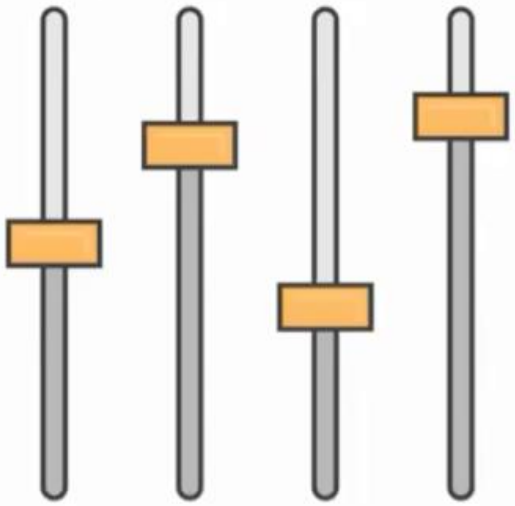
Amazon Workspaces is a virtual desktop in the cloud that also supports audio in and out, you can record your podcast directly in the cloud and also do editing within it on your laptop.



We have our podcast and now want it to go out to the world. We have our mp3 or mp4 format of our podcast.

What makes up a podcast?

- Audio file – MP3/MP4
- RSS feed
- Artwork
- Show notes



RSS feed is a simple text file is how we provide metadata about our podcast.

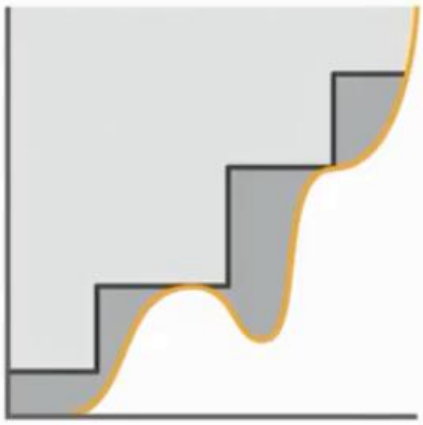
Getting started: Lightsail

- Choose from five plans that include bundled compute, storage, and networking
- Wordpress, Drupal, and Joomla! – all support podcasting extensions
- Low, predictable pricing
- Manage and operate from the Lightsail console

Amazon Lightsail is a great platform for starting out with podcasting.

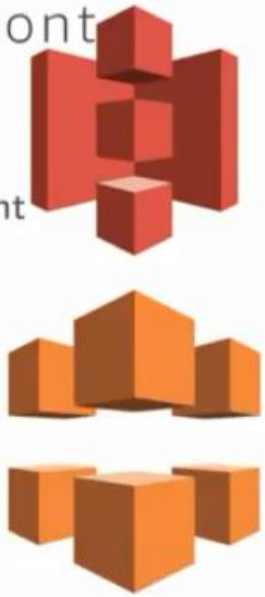
Hosting challenges

- Load on RSS feed
- Traffic spikes
- Measuring and reporting



Amazon S3 and Amazon CloudFront

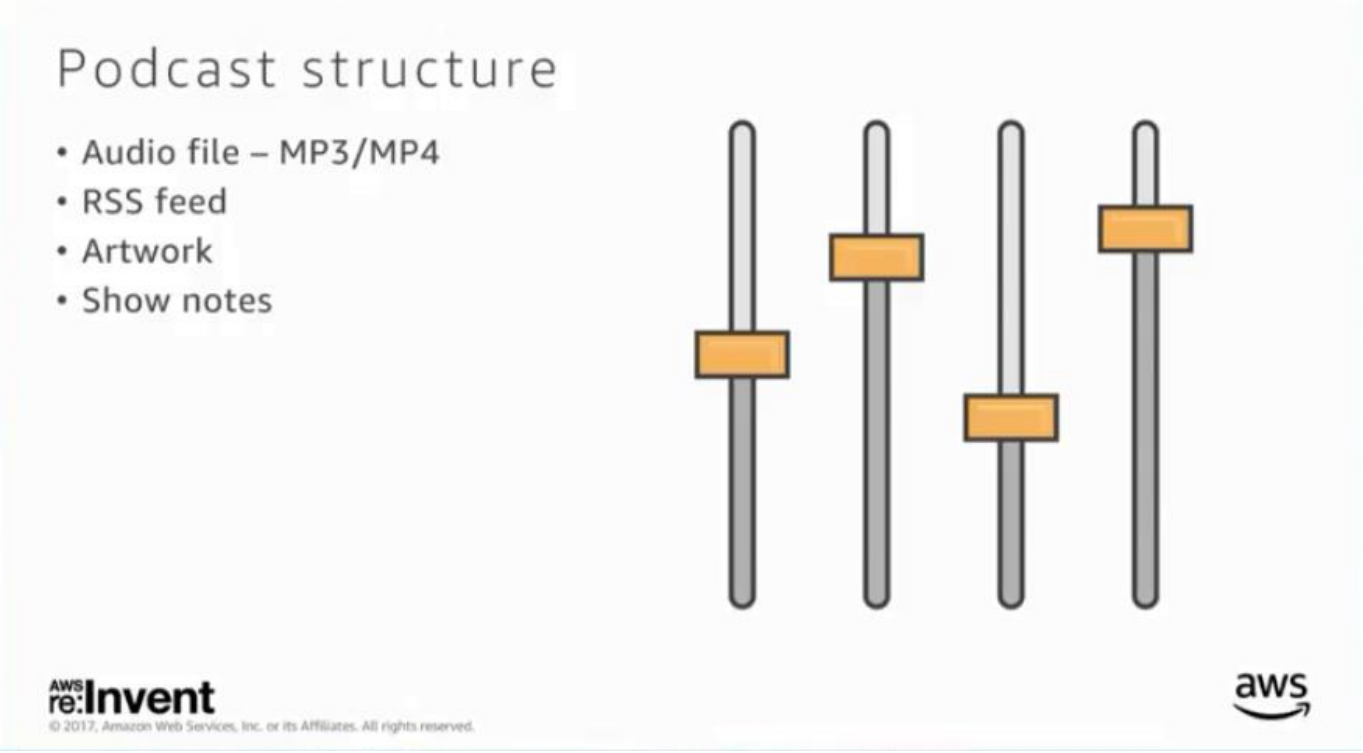
- Low cost, scalable asset storage
- Global distribution via 101 Points of Presence
- ACM integration: easy SSL/TLS certificate management
- Byte-range requests
- Signed URLs
- Detailed logging and usage reports



We have 2 files that we really want to distribute via CloudFront, the mp3 file that is quite large in the tens of MB range and does not change once created, and the RSS file that is a very small text file that might change often. So, you want to have different caching mechanism for both file using the Cache-Control HTTP header setting, set the value for the mp3 file to a large value and a lower value for the RSS feed like 300s. you can also control the setting in the CloudFront settings directly but this is not recommended.

You might want to track your audience using things like querystrings to understand your user, but you need to whitelist specific querystrings that you needso that CloudFront allows it

You might want to purchase reserve capacity on CloudFront for reduced pricing



The RSS feed: A deep dive

- RSS 2.0 standardized in 2003!
- XML based format
- iTunes extensions

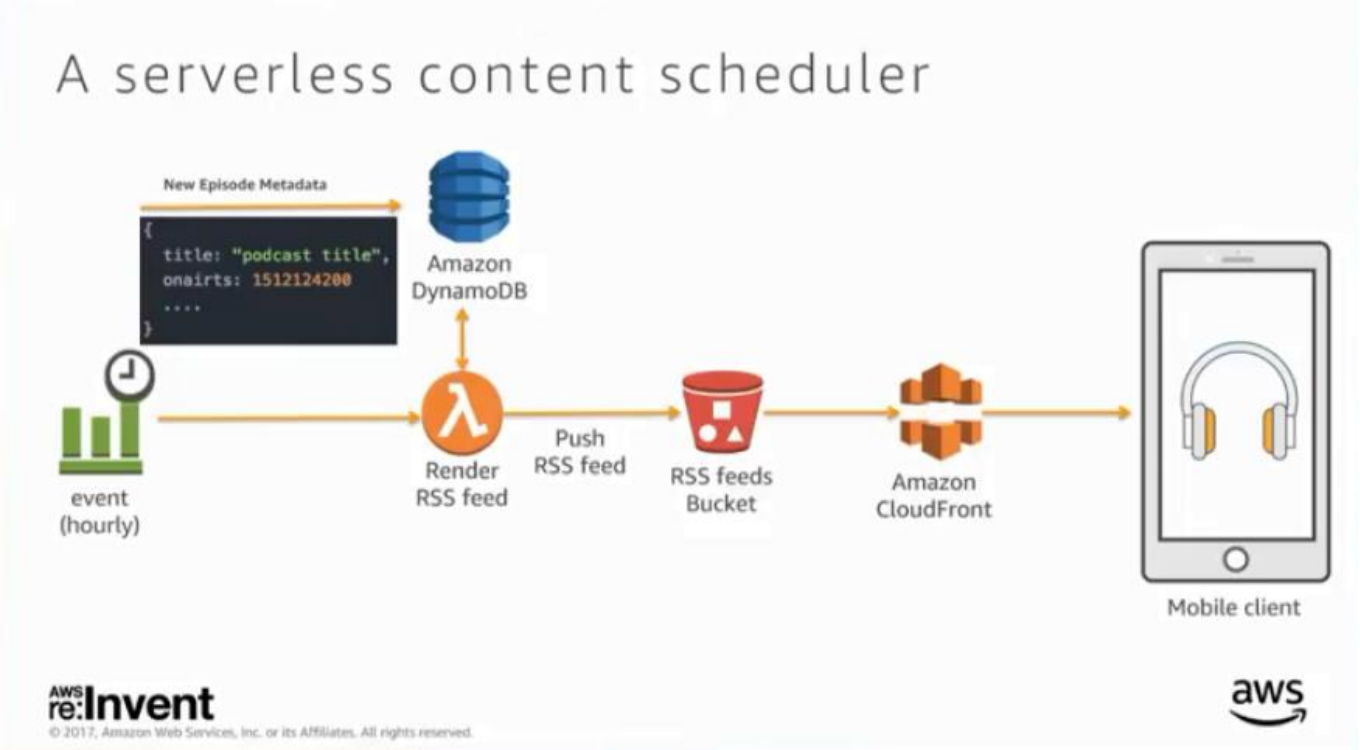
RSS feed: Channel

```
<?xml version="1.0" encoding="utf-8"?>
<rss xmlns:itunes="http://www.itunes.com/dtds/podcast-1.0.dtd" xmlns:atom="http://www.w3.org/2005/Atom" version="2.0">
    <channel>
        <title>Open the Pod Bay Doors</title>
        <atom:link href="https://rss.whooshkaa.com/rss/podcast/id/1755" rel="self" type="application/rss+xml"/>
        <link>https://player.whooshkaa.com/shows/open-the-pod-bay-doors</link>
        <language>en-us</language>
        <lastBuildDate>Wed, 18 Oct 2017 11:47:24 +0000</lastBuildDate>
        <itunes:subtitle>A weekly deep dive into Australia's tech and startup ecosystem. Brought to you by Innovation Bay</itunes:subtitle>
        <itunes:author>Innovation Bay</itunes:author>
        <itunes:summary>There is a lot of activity in the Australian startup ecosystem.</itunes:summary>
        <description><![CDATA[There is a lot of activity in the Australian startup ecosystem.]]></description>
        <itunes:explicit>No</itunes:explicit>
        <itunes:type>episodic</itunes:type>
        <itunes:owner>
            <itunes:name>Innovation Bay</itunes:name>
            <itunes:email>                    </itunes:email>
        </itunes:owner>
        <itunes:image href="http://files.whooshkaa.com/podcasts/podcast_1755/podcast_media/1084ac-hal2.PNG"/>
        <itunes:category text="Business"/>
        <image>
            <url>http://files.whooshkaa.com/podcasts/podcast_1755/podcast_media/1084ac-hal2.PNG</url>
            <title>Open the Pod Bay Doors</title>
            <link>https://player.whooshkaa.com/shows/open-the-pod-bay-doors</link>
            <description>There is a lot of activity in the Australian startup ecosystem.</description>
        </image>
    </channel>
</rss>
```

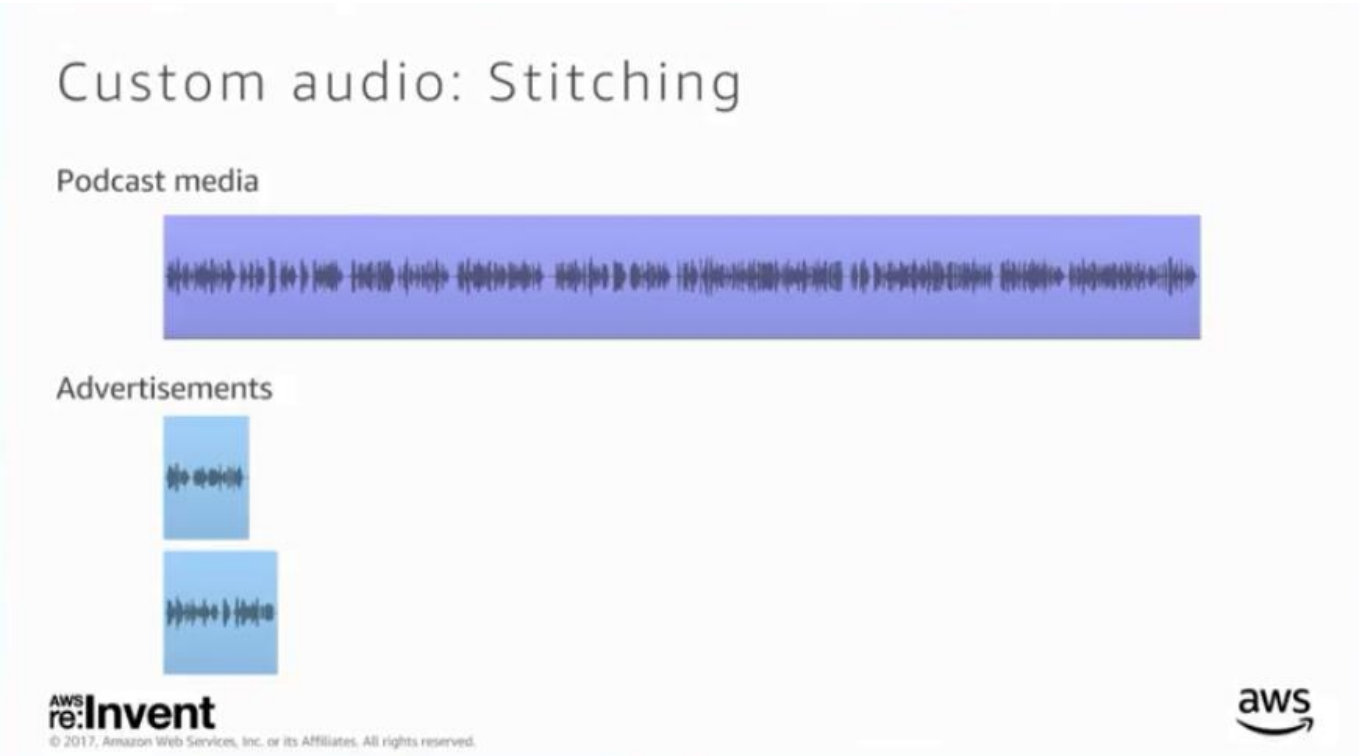
RSS feed: Episode

```
<item>
    <title>E21 - Dan Cohen, Flare HR</title>
    <link>https://player.whooshkaa.com/episode?id=148645</link>
    <pubDate>Thu, 19 Oct 2017 06:00:03 +1100</pubDate>
    <itunes:author>Innovation Bay</itunes:author>
    <itunes:summary>Dan Cohen is an Australian serial entrepreneur.</itunes:summary>
    <description><![CDATA[<p>Dan Cohen is an Australian serial entrepreneur.</p>]]></description>
    <itunes:subtitle>Dan Cohen is an Australian serial entrepreneur.</itunes:subtitle>
    <itunes:image href="http://files.whooshkaa.com/podcasts/podcast_1755/podcast_media/717ece-dan-c-big-3000x.jpg"/>
    <enclosure url="http://media.whooshkaa.com/podcasts/1755/episodes/48c275-ep-open-the-pod-bay-doors-dan-cohen.mp3?id=148645" length="122595968" type="audio/mpeg"/>
    <itunes:duration>51:05</itunes:duration>
    <guid isPermaLink="false">fBeccbdb-c773-46ad-93f7-7bf6b51d7ead</guid>
    <itunes:keywords>dan cohen, flare hr, reinventure, ian gardiner, innovation bay, venture capital, technology</itunes:keywords>
</item>
```

You might have to update the Episode section frequently within the RSS feed

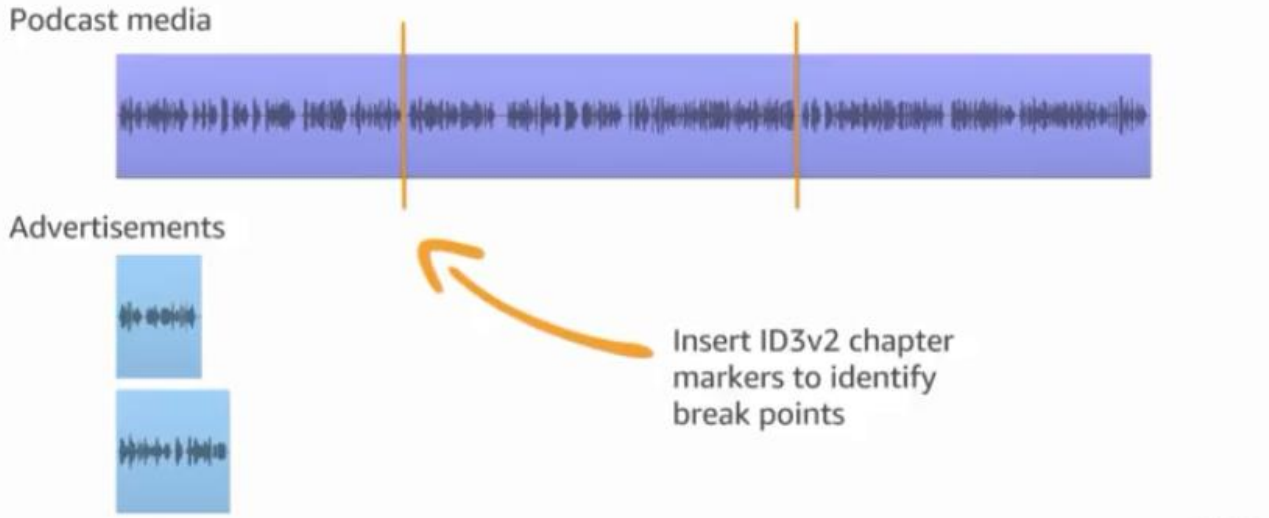


You can automate the process of publishing the podcast at a specific date and time using a CloudWatch event, lambda function, DynamoDB table with the metadata about your podcast to push them into your RSS feeds and deploy



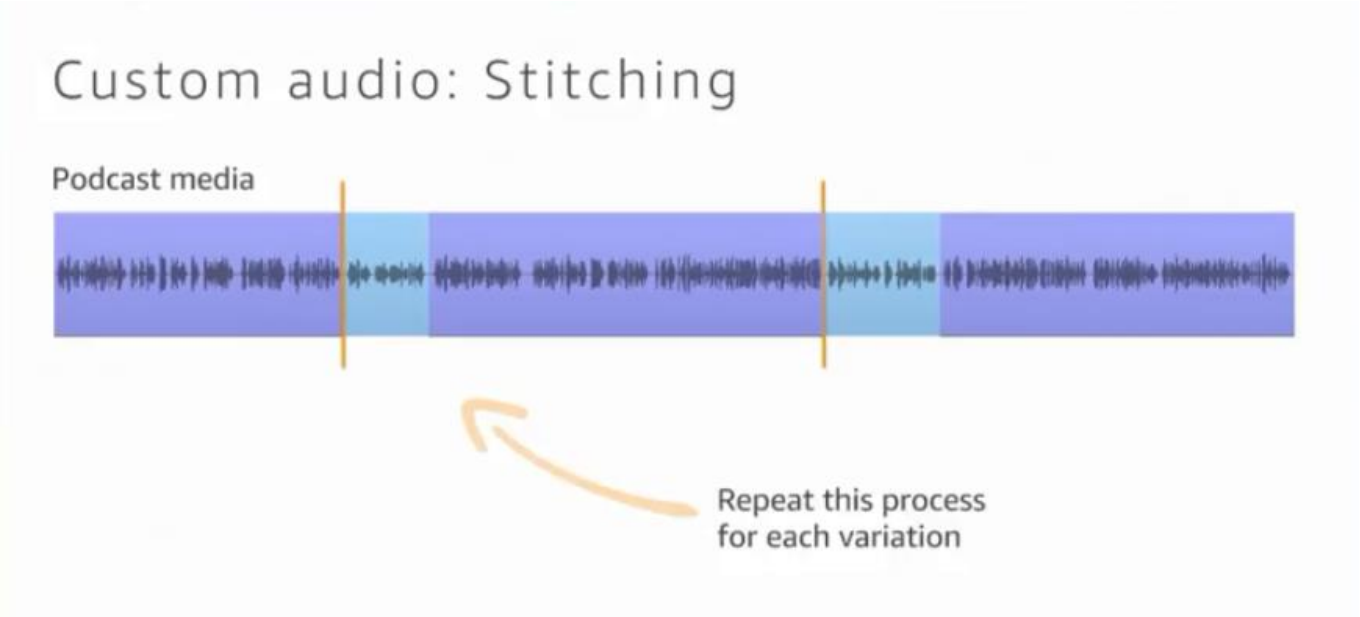
You might want to put different ads in for different audiences and markets, you can also remove ads after some time by re-publishing your podcasts without the ads

Custom audio: Stitching

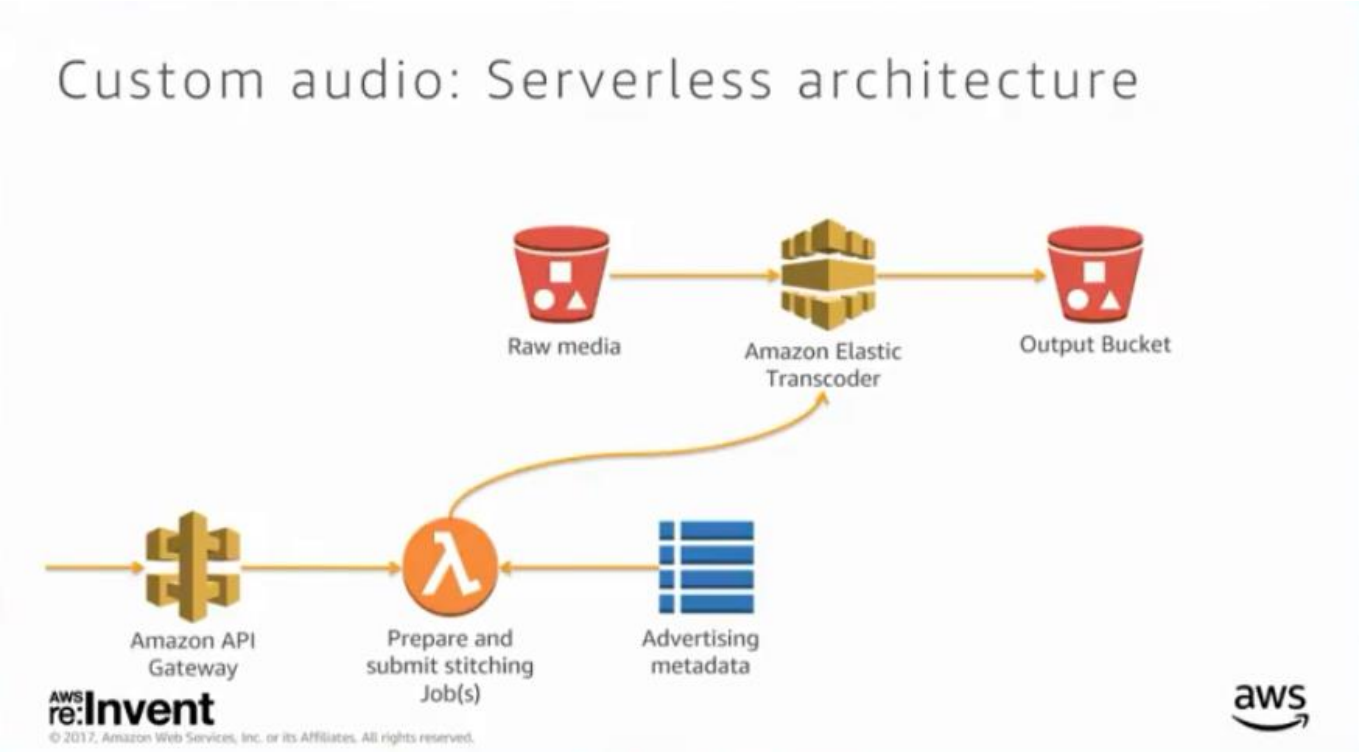


Custom audio: Stitching

aws re:Invent
© 2017, Amazon Web Services, Inc. or its Affiliates. All rights reserved.
aws



You need to repeat this automated process for each of your audience markets using different ads. You can have the ads all having a standardized length.



You can use API Gateway with lambda for this serverless architecture. The lambda function needs to understand the source media using like the file name of the input master mp3 clip and what ads to insert, this information can be stored in a DynamoDB table. The Advertising metadata includes things like time, markets, audience for the markets, ads to insert for each market, number of ad insertions, etc. then you can prepare a Job and use the Elastic Transcoder to do stitching tasks by sending it as a transcode job. The Elastic Transcoder will pull in your different mp3 files and media for the podcast and ads, then it will stitch them together and re-encode the content to the format you want and write it out to the Output bucket, you then update your RSS feed with the details of the final mp3 file.

Finding chapter markers

```
fs.readFile(pathToMp3, (error, buffer) => {
    buffer = new DataView(toArrayBuffer(buffer));

    const tags = mp3Parser.readTags(buffer);
    tags[0].frames.forEach(function(tag, index, initial_array) {
        if (tag.header.id == 'CHAP'
            && tag.content.endTime > tag.content.startTime) {
            startTime_s = (tag.content.startTime / 1000).toFixed(3);
            endTime_s = (tag.content.endTime / 1000).toFixed(3);
            ...
        }
    })
});
```

https://biril.github.io/mp3-parser/index.html

The code above uses an OSS library called mp3Parser. The chapter markers are where we want to insert our ads

Amazon Elastic Transcoder clip stitching

```
createJob_params = {
   "Inputs":[{
      "Key":"sourcemedia.mp3",
      "TimeSpan":{
         "StartTime": startTime_s[0],
         "Duration": (endTime_s[0] - startTime_s[0]).toFixed(3)
      }
   },{
      "Key":"advertisement1.mp3",
   },{
      "Key":"sourcemedia.mp3",
      "TimeSpan":{
         "StartTime": startTime_s[1],
         "Duration": (endTime_s[1] - startTime_s[1]).toFixed(3)
      }
   }]
   ...
}
```

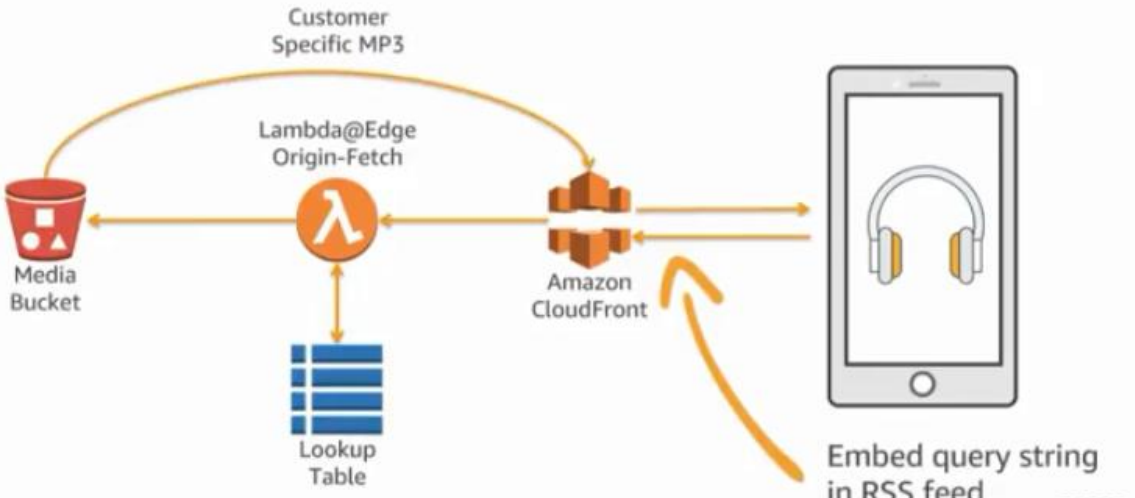
Once we know the points where those chapter markers are and we want to put our ads there, we can then go ahead and generate an elastic Transcoder job like above, it is a JSON document where we specify inputs, and the time offsets.

Amazon Elastic Transcoder clip stitching

```
createJob_params = {
   "Inputs":[{
      "Key":"sourcemedia.mp3",
      "TimeSpan":{
         "StartTime": startTime_s[0],
         "Duration": (endTime_s[0] - startTime_s[0]).toFixed(3)
      }
   },{
      "Key":"advertisement1.mp3",
   },{
      "Key":"sourcemedia.mp3",
      "TimeSpan":{
         "StartTime": startTime_s[1],
         "Duration": (endTime_s[1] - startTime_s[1]).toFixed(3)
      }
   }]
   ...
}
```

Re-use the same media with different StartTime values

Custom audio: Delivery architecture



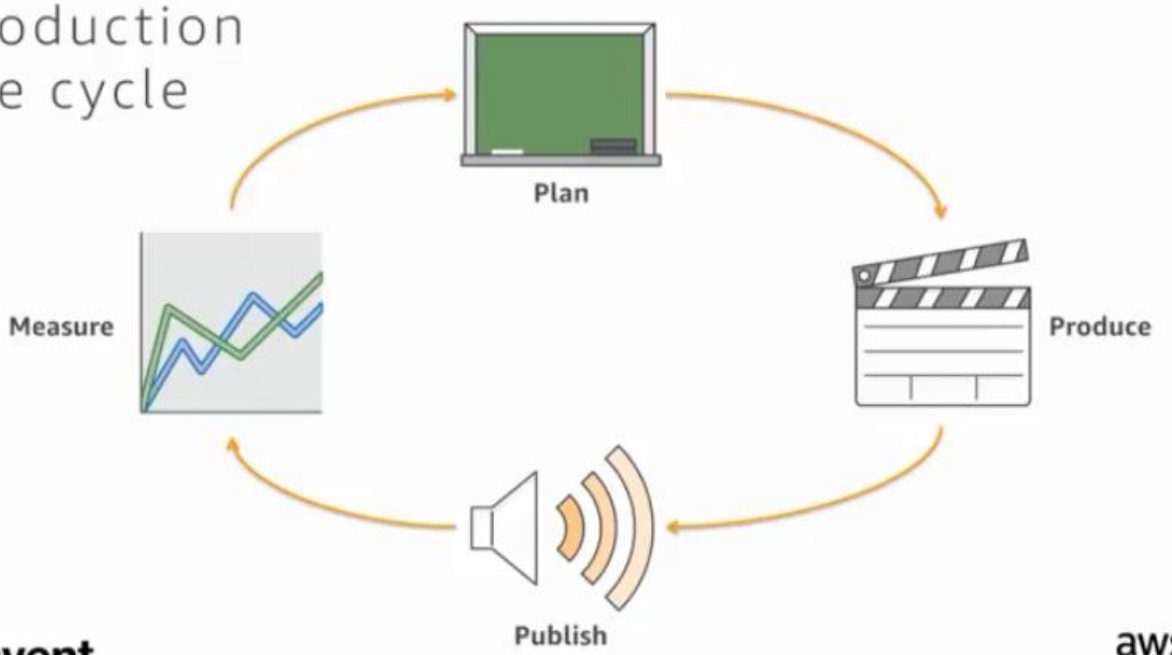
Now we have our custom audio with the ads inserted into the final mp3 file. We can now serve different mp3 versions to the different markets we serve using a querystring in CloudFront Lambda@Edge to transparently rewrite the URL to fetch different mp3 files based on the RSS feed.

Lambda@Edge: Rewriting requests

- Origin-request function
- Watch out for inter-region calls
- Consider replicating DynamoDB and other resources to reduce latency
- Strip query string before forwarding to Amazon S3

```
exports.handler = (event, context, callback) => {
    var ddb = new AWS.DynamoDB({apiVersion: '2012-10-08'});
    const request = event.Records[0].cf.request;
    const params = querystring.parse(request.querystring);

    var query_params = {
        TableName: 'campaigntable',
        Key: {
          'URI': {S: request.uri},
          'Campaign': {S: params.campaignid}
        }
    }

    ddb.getItem(query_params, function(err, data) {
        if(err) console.log(err, err.stack);
        else {
            request.uri = data.Item.CustomURI.S;
            delete params.campaignid; // Strip querystring
            console.log("Request uri set to "+request.uri);
            request.querystring = querystring.stringify(params);
            callback(null, request);
        }
    })
};
```


Production life cycle

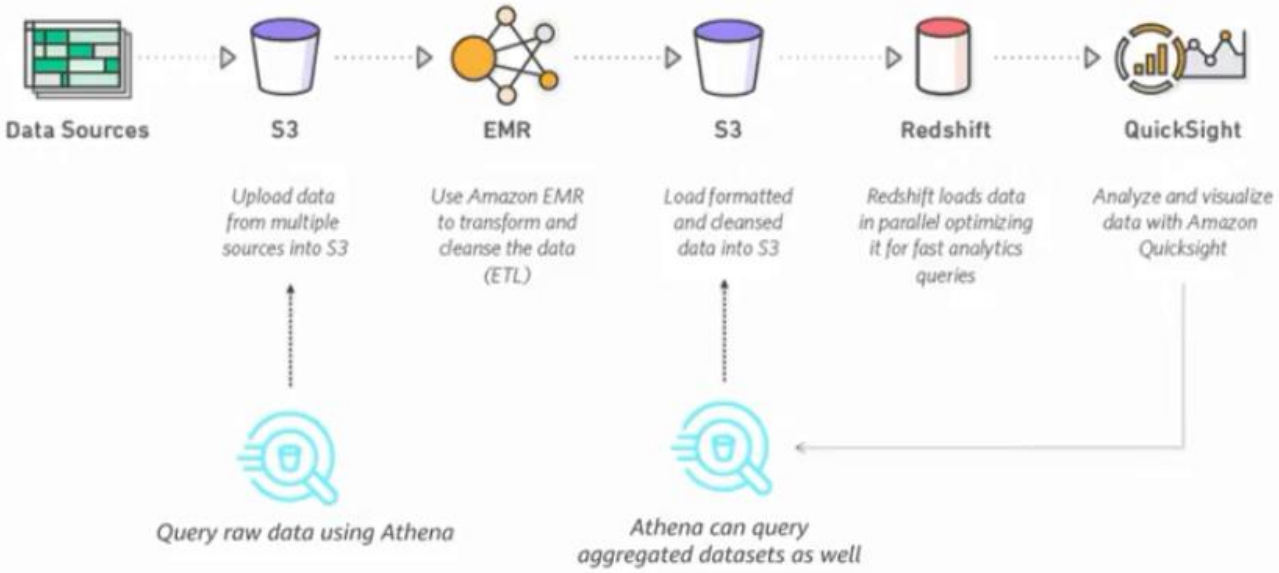


Key metrics

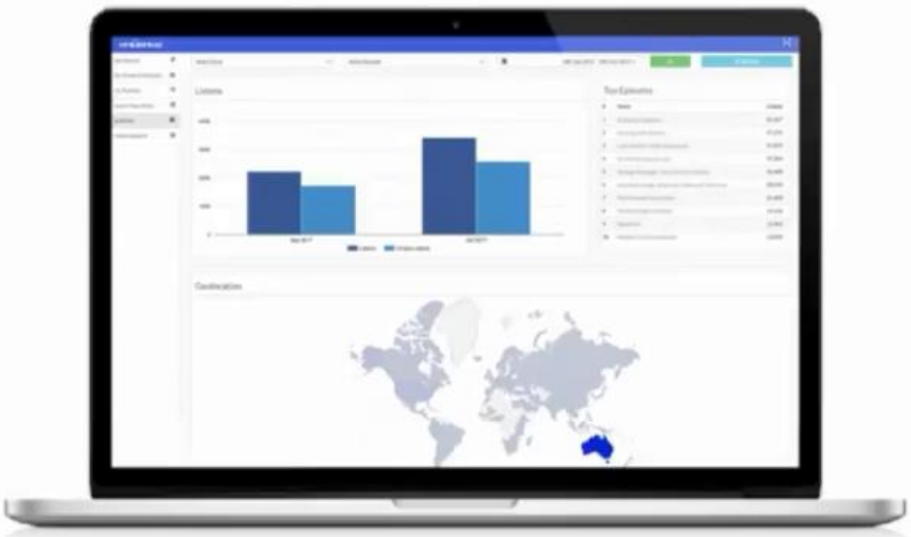
CloudFront reports
- Audience reach
- Popular objects
- Top referrers
- Cache performance

Custom reports
- Audience reach per podcast
- Unique listeners over time
- Top referrers over time

Podcast analytics pipeline



Whooshkaa analytics



Whooshkaa – End-to-end audio

Whooshkaa – End-to-end audio

Platform

Amplification

Dynamic ads

Capture/editing

Live

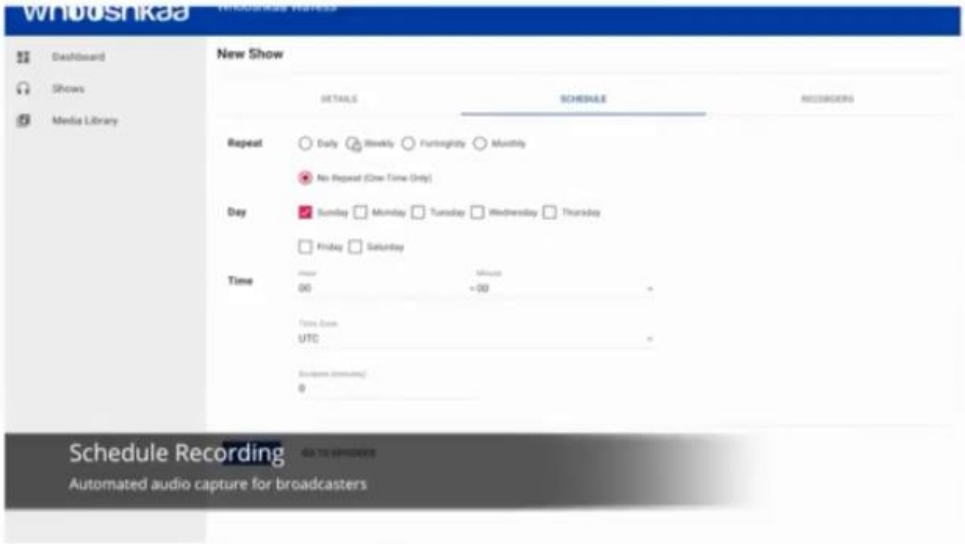
Text to voice personalised audio

Whooshkaa Wavess

Whooshkaa Wavess

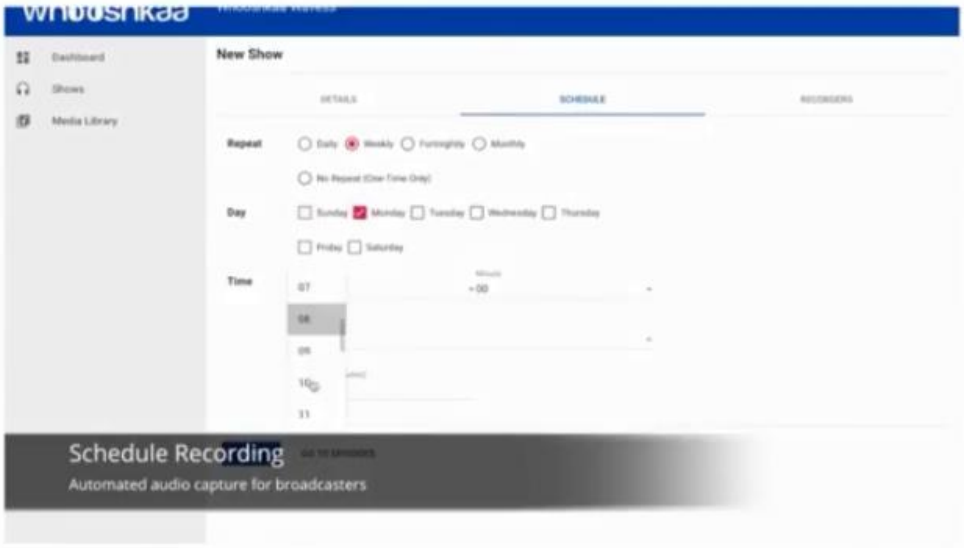
Whooshkaa Wavess

Whooshkaa Wavess

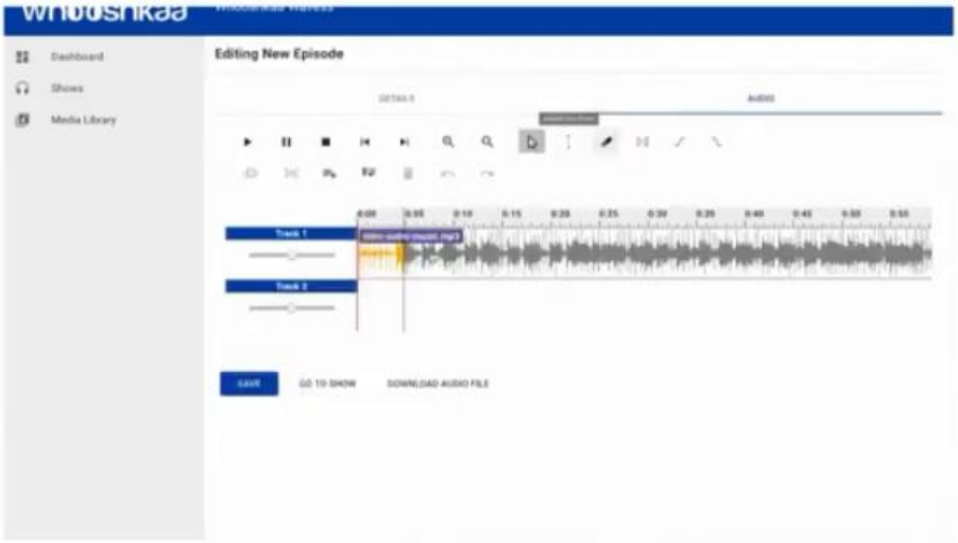


AWS re:Invent
© 2017, Amazon Web Services, Inc. or its Affiliates. All rights reserved.
aws

Whooshkaa Wavess

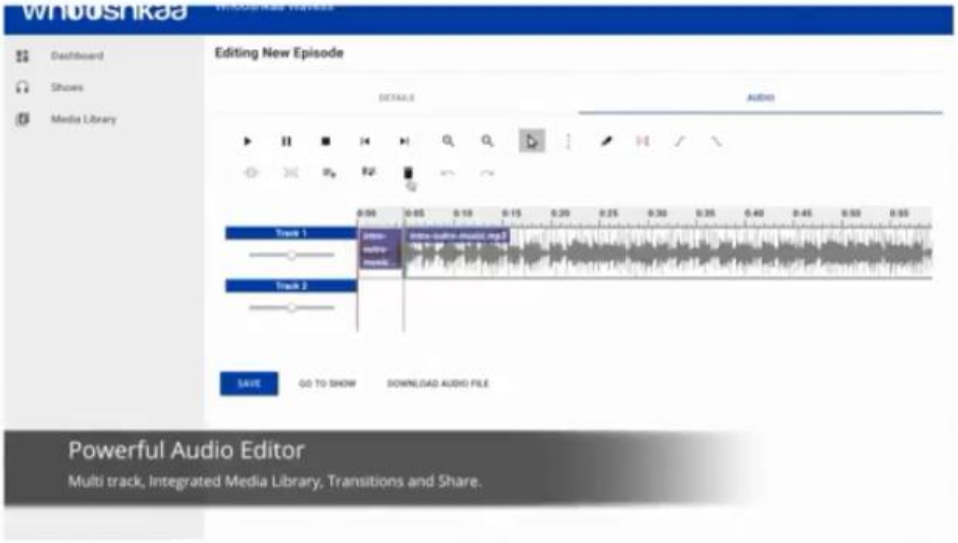


Whooshkaa Wavess

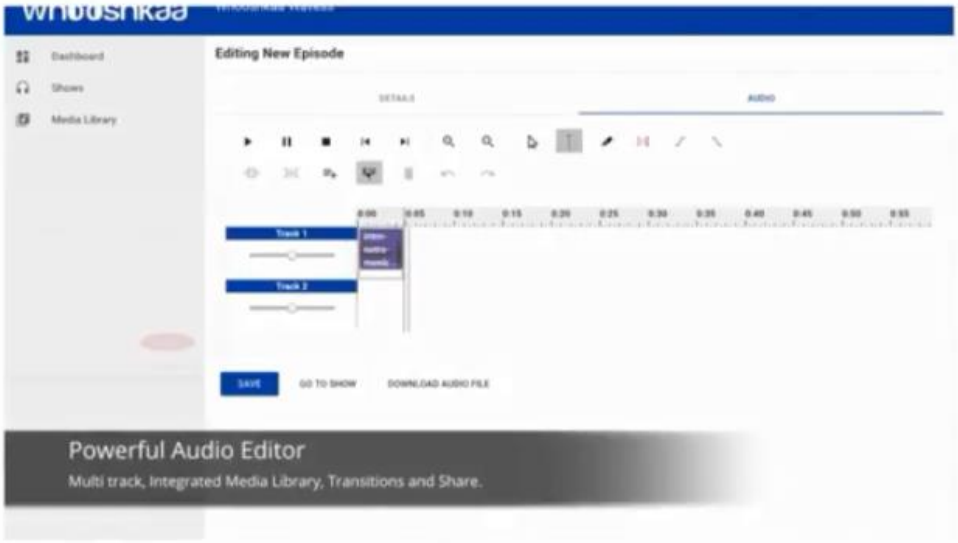


AWS re:Invent
© 2017, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Whooshkaa Wavess



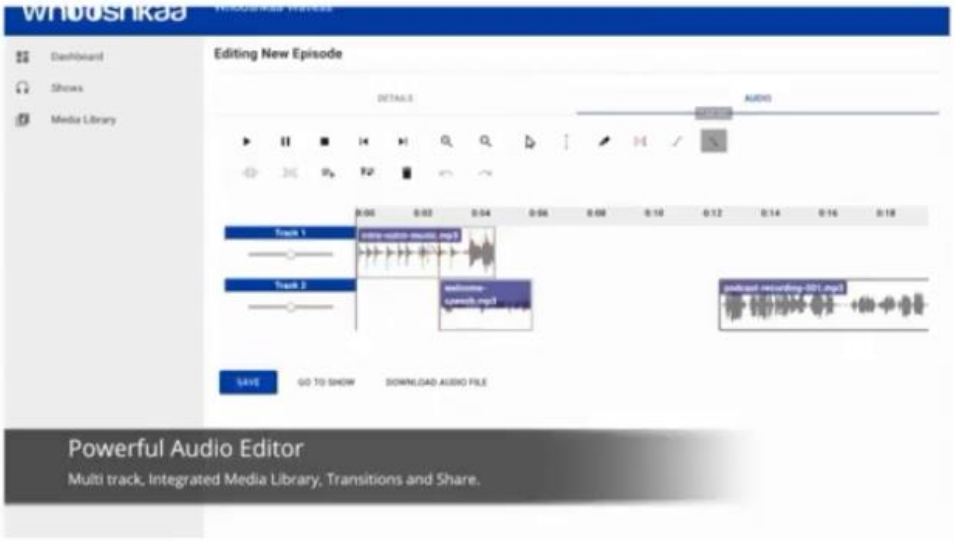
Whooshkaa Wavess



© 2017, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

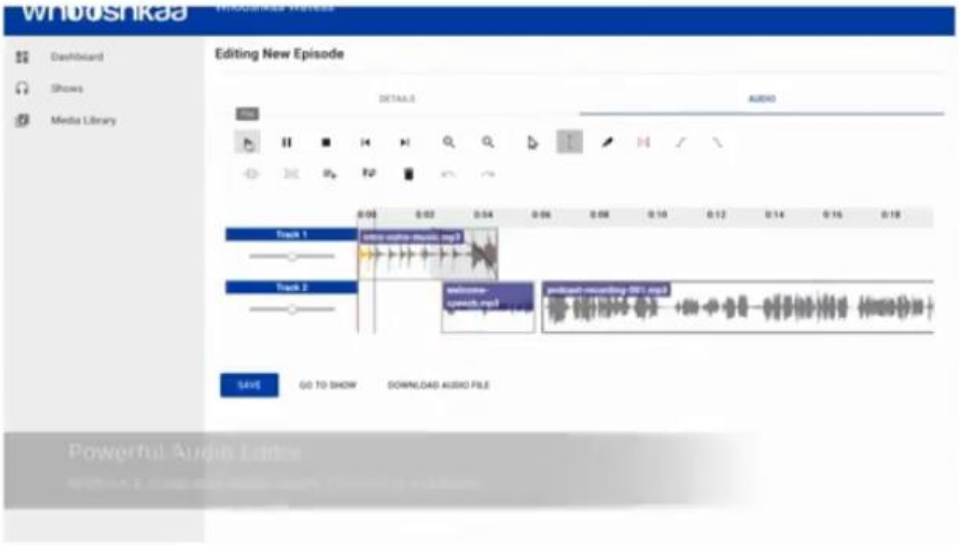
Whooshkaa Wavess

Whooshkaa Wavess



AWS re:Invent
© 2017, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Whooshkaa Wavess



Whooshkaa Wavess

AWS re:Invent
© 2017, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Whooshkaa Wavess

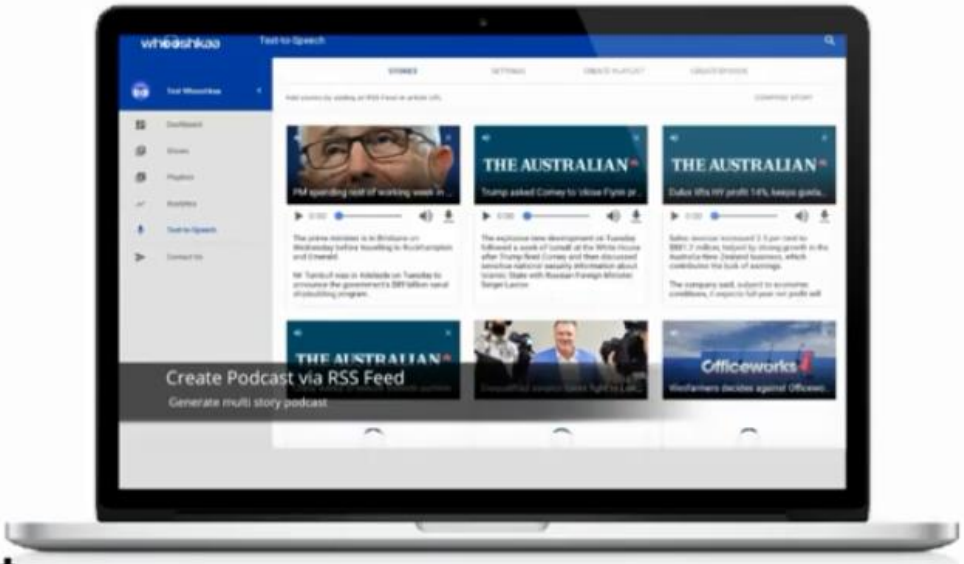
LOUDER
THAN LIFE

Whooshkaa Wavess

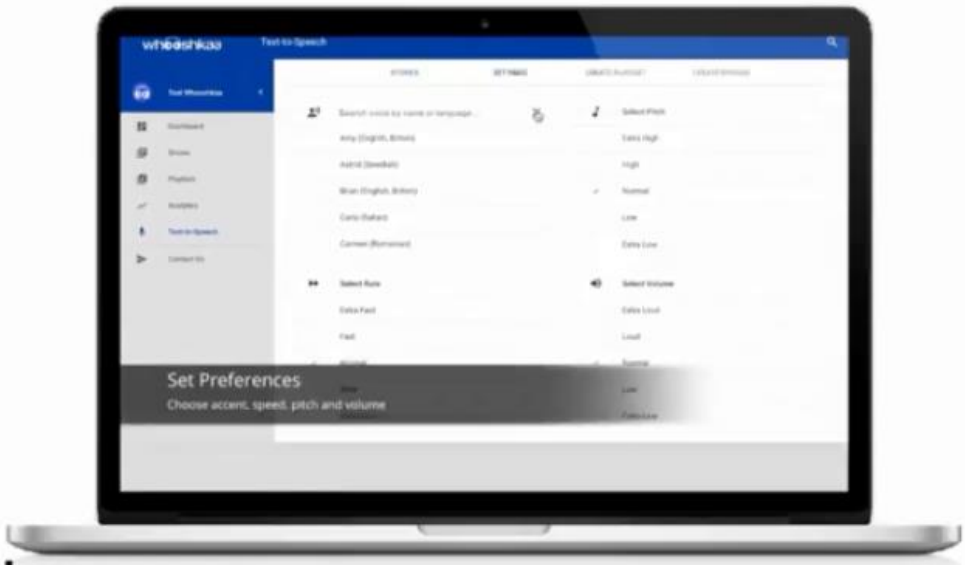
AWS re:Invent
© 2017, Amazon Web Services, Inc. or its Affiliates. All rights reserved.
aws

Live streaming

Revolutionizing text with voice



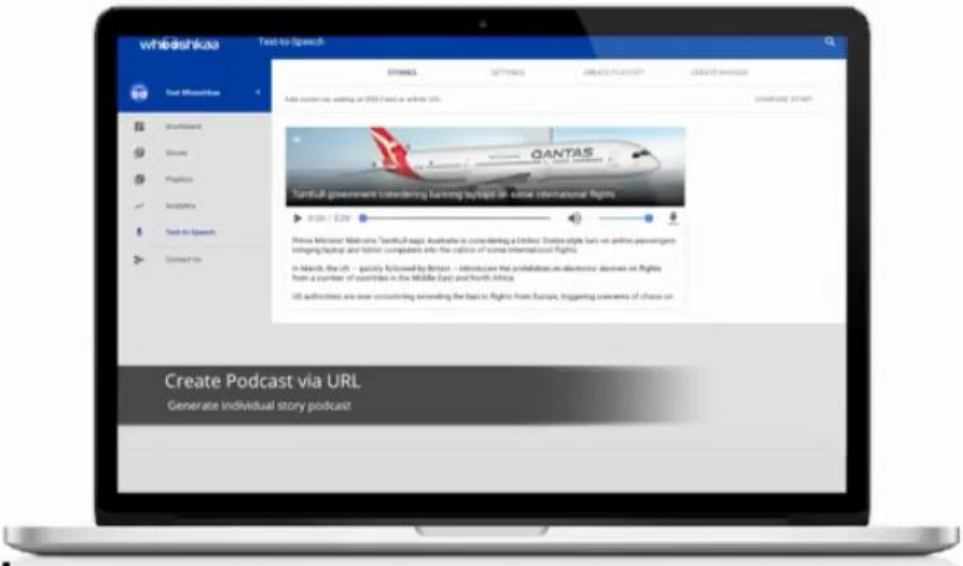
Revolutionizing text with voice



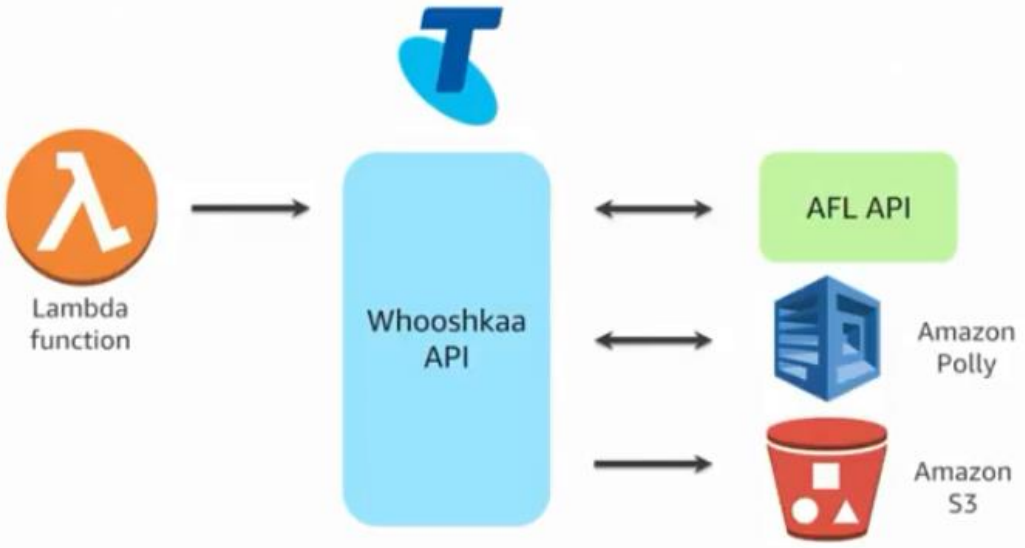
Revolutionizing text with voice

AWS re:Invent
© 2017, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Revolutionizing text with voice



Polly powering sports scores

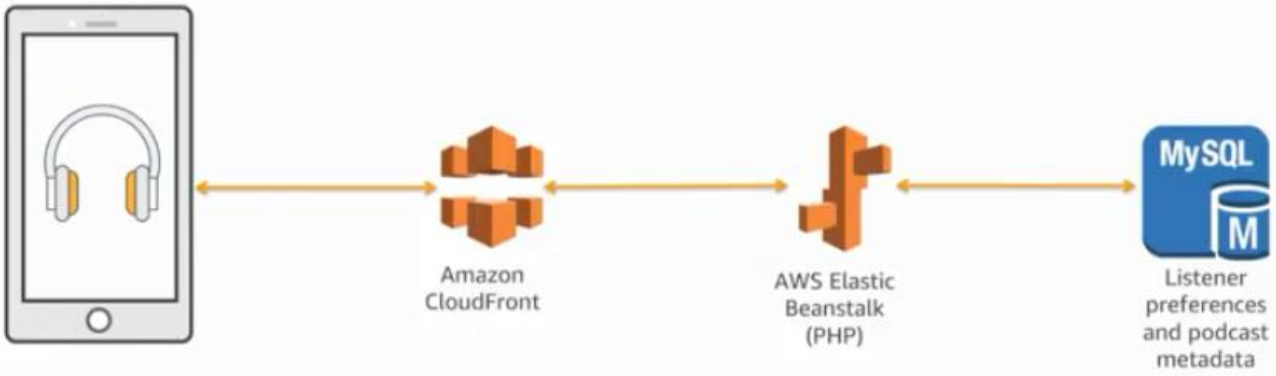


Polly powering sports scores

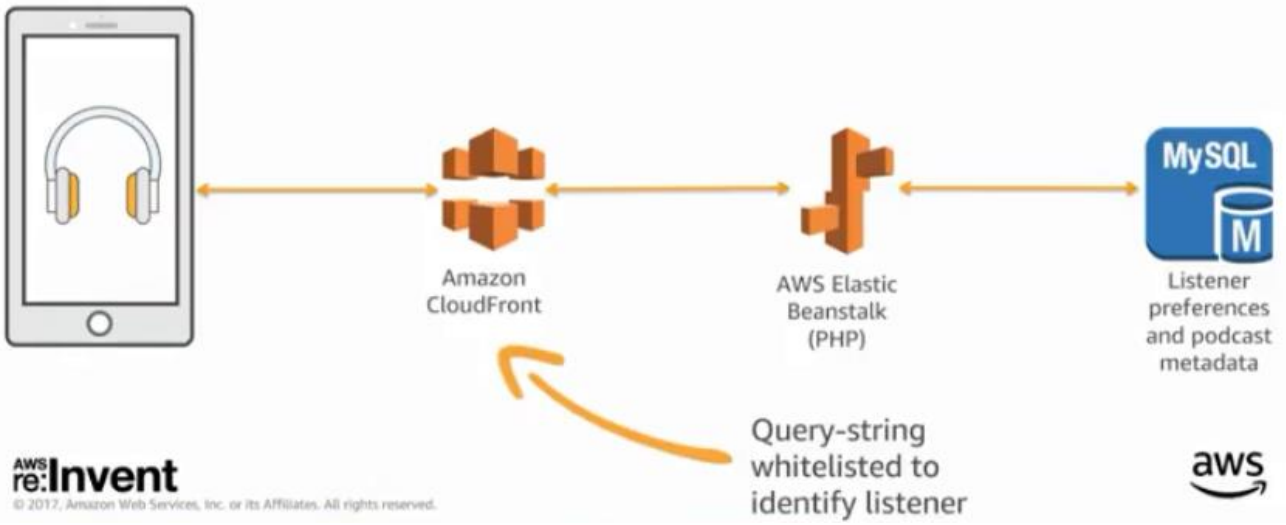
```
private function makeAudio($story)
{
    $polly = new Polly;
    $polly->setPhonemes(['live' => 'laɪve']);
    $polly->setProsody('AFL', ['rate' => 'fast']);
    $polly->setSayAs(['MCG' => 'spell-out']);

    $text = $story->getStory();
    // Trim the text to a maximum of 1500 characters.
    if (strlen($text) > 1499) {
        $text = $this->text->setText($text)->trimToWordBoundary(1499);
    }

    try {
        $audioStream = $polly->fetchAudioStream($text);
    }
    catch (\Exception $e) {
        $this->response->error($e->getMessage(), $e->getStatusCode());
    }

    return response()->make($audioStream)->header('Content-Type', 'audio/mpeg');
}
```


Amazon Polly allows you to programmatically generated voice from text.

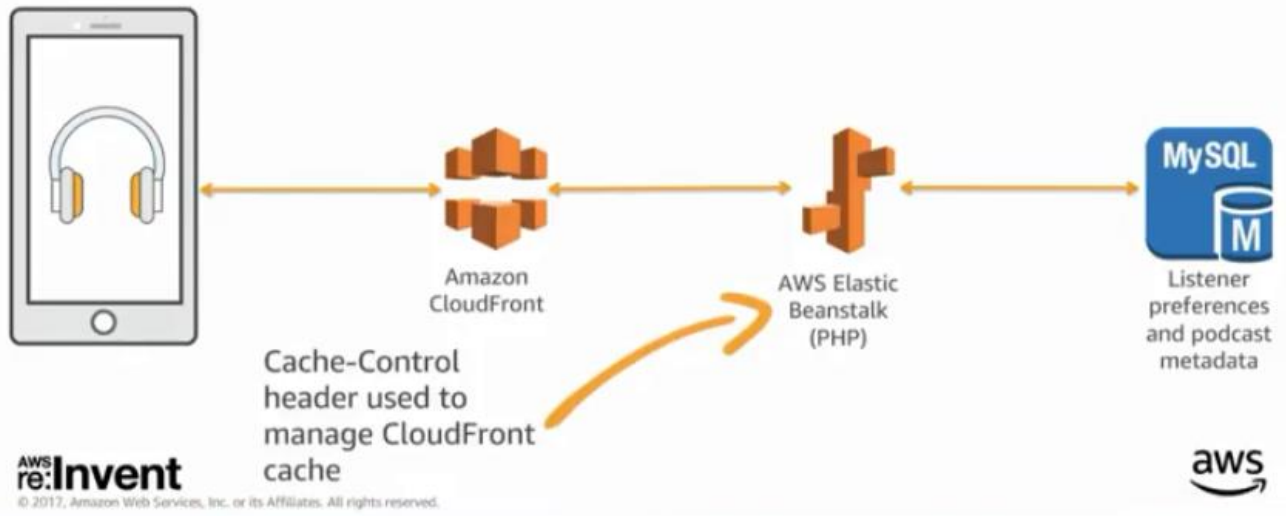
Customized audio architecture



Customized audio architecture



Customized audio architecture

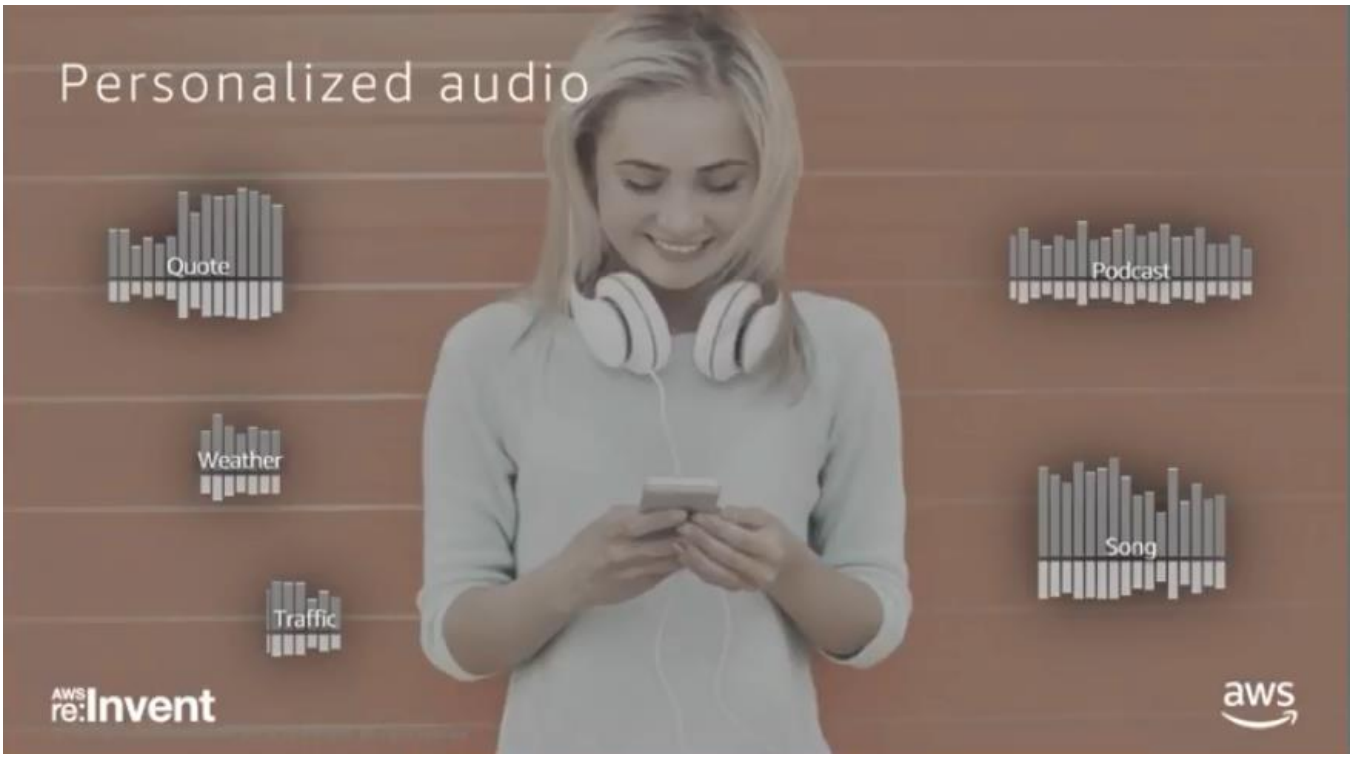


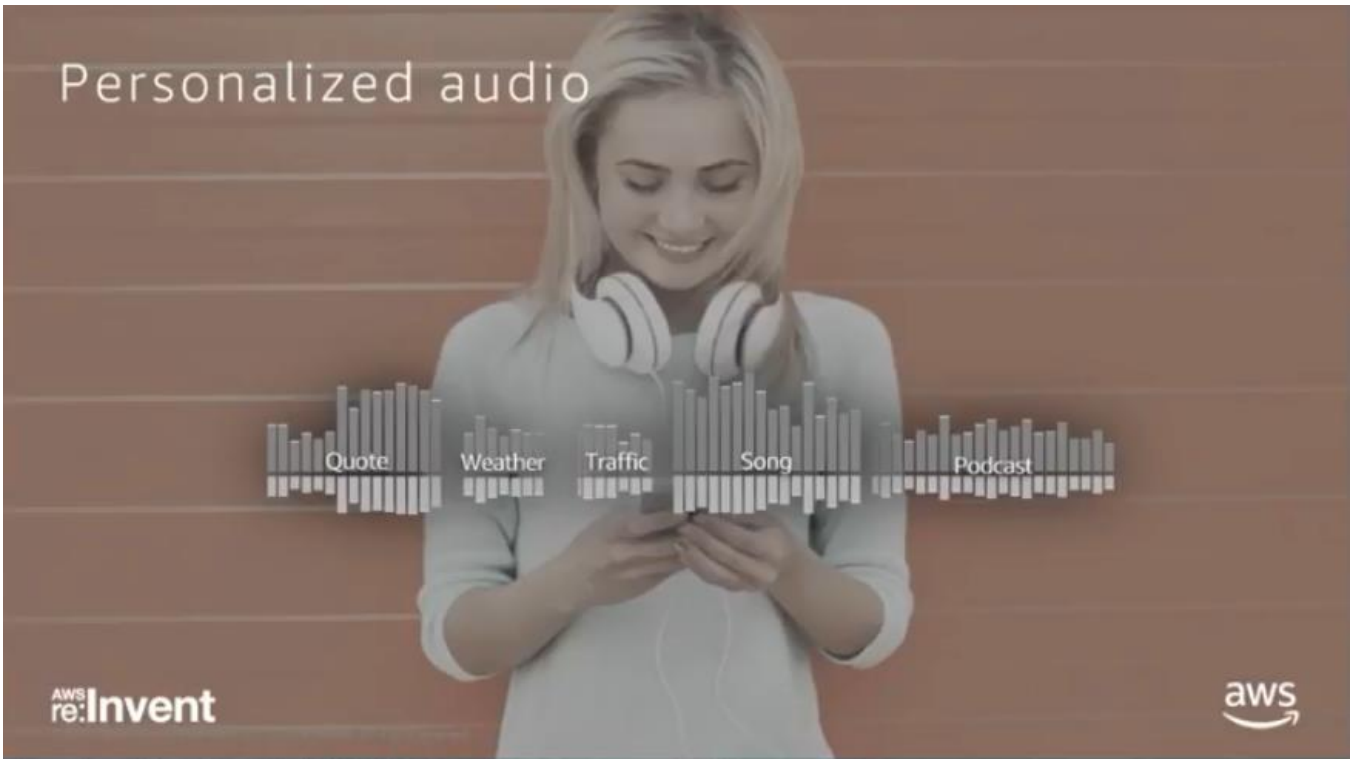
Rendering custom RSS feeds: Feed

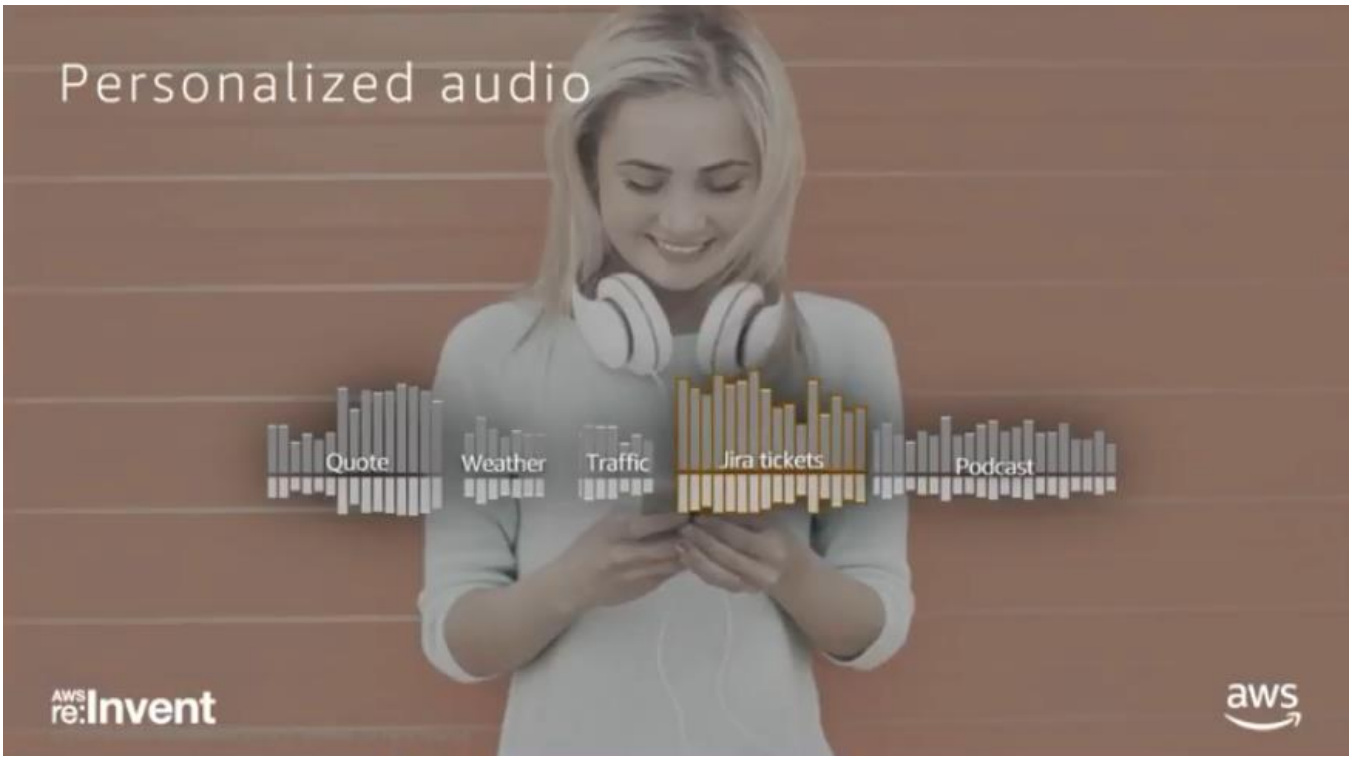
```php
private function makeFeed(Team $team, $story)
{
    $feed = new Feed($this->publisher->getRemoteURL('feed.xml'));
    $feed->setTitle($team->getName() . "'s Official Live Feed");
    $feed->setDescription('An official live feed from the AFL.');
    $feed->setLink('http://www.afl.com.au');
    $feed->setOwner('The Australian Football League', 'podcast@afl.com.au');
    $feed->setImage($team->getImage());
    $feed->appendElements([
        'itunes:subtitle' => "Follow {$team->getName()}'s Live Matches",
        'itunes:explicit' => 'no',
        'language' => 'en-us',
        'lastBuildDate' => Carbon::now('UTC')->toRssString(),
        'ttl' => 2,
        'copyright' => 'The Australian Football League',
    ]);
```


Rendering custom RSS feeds: Items

```php
    $item = $feed->addItem([
        'title' => $story->getTitle(),
        'link' => $story->getArticleURL(),
        'pubDate' => Carbon::now('UTC')->toRssString(),
        'itunes:duration' => $metaData['playtime_string'],
    ]);
    $item->appendDescription($story->getStory());
    $item->appendEnclosure($this->publisher->getRemoteUrl($fileName, true),
       $metaData['filesize'], $metaData['mime_type']);
    $item->append('itunes:image', null, ['href' => $team->getImage()]);
    $item->append('guid', $this->publisher->getGuid(), ['isPermaLink' => 'false']);

    return response()->make($feed->output())->header('Content-Type', 'text/xml');
}
```


Personalized audio
Quote
Podcast
Weather
Song
Traffic
AWS re:Invent
aws


Personalized audio
Quote
Weather
Traffic
Song
Podcast
AWS re:Invent
aws


Personalized audio
Quote
Weather
Traffic
Jira tickets
Podcast
AWS re:Invent
aws


AWS re:Invent
Thank you!
AWS re:Invent
aws
© 2017, Amazon Web Services, Inc. or its Affiliates. All rights reserved.