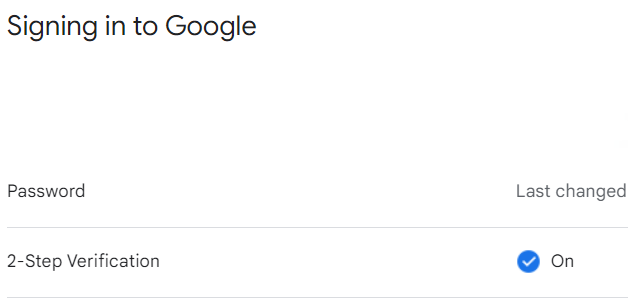
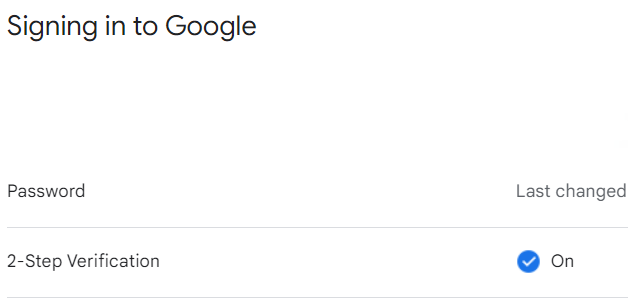
Sending emails from Alexa

**Send email from Alexa**

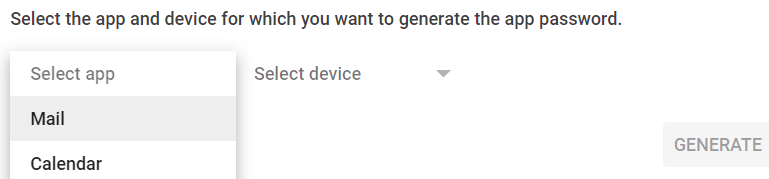
1 ensure google lets you.

1. Log in into your email account: <https://myaccount.google.com>’
2. Go to security and check two-stage verification is ON

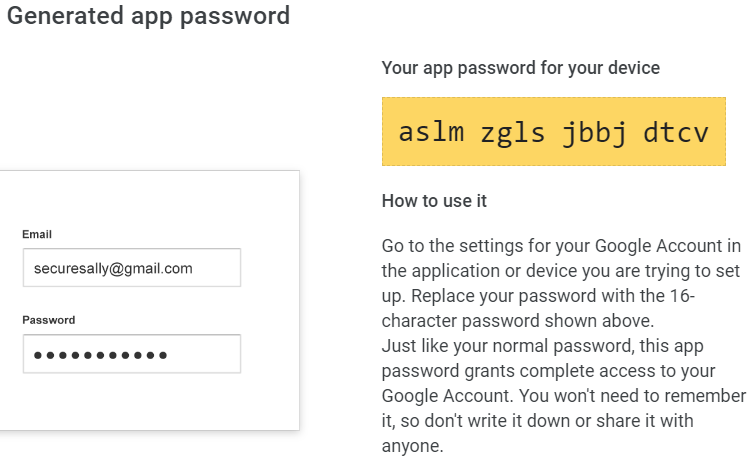




1. click on "App password” and select Mail, type Alexa for the device and Generate



A password is generated



This is the password that you have to use in your python code.

You can use smtplib and ssl, or email-to

1. Python code to send your email – using smtplib and ssl

import smtplib, ssl

# these are needed for attachment

from email.mime.text import MIMEText

from email.mime.multipart import MIMEMultipart

port = 465 # For SSL

smtp\_server = "smtp.gmail.com"

sender\_email = " sender\_email @gmail.com" # Your address as validated above

receiver\_email = " receiver\_email @gmail.com" # Enter the recipient address

password = "aslmzglsjbbcdef" # from Gmail

message = """\

Hi there

This message is sent from Python – check your spam folder."""

context = ssl.create\_default\_context()

with smtplib.SMTP\_SSL(smtp\_server, port, context=context) as server:

server.login(sender\_email, password)

server.sendmail(sender\_email, receiver\_email, message)

1. Now let’s do it from Alexa, using smtplib and ssl

Start an Alexa Hosted skill, python, start from scratch.

First, we’ll just send a header with some text, then we’ll add an attachment

I’ve saved the sender, recipient and password in a credentials.py file that looks like this: (n.b. the password is that given to you by Gmail)

key = {

'sender\_email' : “sender\_email@gmail.com",

'receiver\_email' : receiver\_email@gmail.com',

'password' : "aslmzglsjbbabcd”

}

Add this at the top of your code (we’ll use some of it later)

import smtplib, ssl

from email.mime.text import MIMEText

from email.mime.multipart import MIMEMultipart

import credentials

import os

import boto3

Change the LaunchRequest to

class LaunchRequestHandler(AbstractRequestHandler):

#Handler for Skill Launch.

def can\_handle(self, handler\_input):

# type: (HandlerInput) -> bool

return ask\_utils.is\_request\_type("LaunchRequest")(handler\_input)

def handle(self, handler\_input):

# type: (HandlerInput) -> Response

speak\_output = "I've sent your email"

port = 465 # For SSL

smtp\_server = "smtp.gmail.com"

# don't forget to import credentials

sender\_email = credentials.key['sender\_email'] # Enter your address

receiver\_email = credentials.key['receiver\_email'] # Enter receiver address

password = credentials.key['password'] # "given gmail password"

# Send simple text.

#message = "This message is sent using smtplib from Python."

# send multipart message

message = MIMEMultipart("alternative")

message["Subject"] = "multipart message from Alexa"

message["From"] = sender\_email

message["To"] = receiver\_email

# add plain text and HTML versions

text = """\

You can find perkin at tinyurl.com/y8dk523j

"""

html = """\

<html>

<body>

<p>Hi,<br>

<a href="tinyurl.com/y8dk523j">Here’s perkin</a>

</p>

</body>

</html>

"""

# Convert to plain and html MIMEText objects

plainMIME = MIMEText(text, "plain")

HTMLMIME = MIMEText(html, "html")

# Add the HTML and plain-text to the MIME message

# The email client renders the last part first

message.attach(plainMIME)

message.attach(HTMLMIME)

# and send message as string()

context = ssl.create\_default\_context()

with smtplib.SMTP\_SSL("smtp.gmail.com", 465, context=context) as server:

server.login(sender\_email, password)

server.sendmail(sender\_email, receiver\_email, message.as\_string())

return (

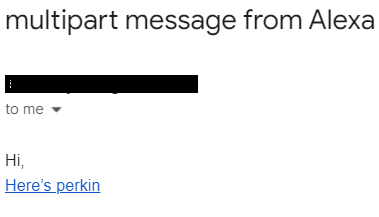
handler\_input.response\_builder

.speak(speak\_output)

.ask(speak\_output)

.response

)



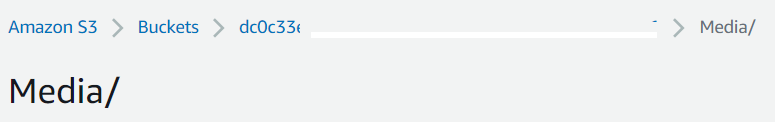
1. Adding Attachments

To send binary files, they need to be encoded, normally using base64

Upload a file to S3.

We need the key: Media/document.pdf

We also need the bucket name. You can see this by clicking the S3 tab. It’s at the top, something like dc0c33ed-3ce8-4a33-b7c6-987654321abcde-us-east-1.



But you can get this from your code, by using

import os

bucket\_name = os.environ.get('S3\_PERSISTENCE\_BUCKET')

(see utils.py)

and now use the boto download\_file method. This accepts the bucket name, object name (Media/..) filename to save to. LaunchRequest is:

class LaunchRequestHandler(AbstractRequestHandler):

#Handler for Skill Launch.

def can\_handle(self, handler\_input):

# type: (HandlerInput) -> bool

return ask\_utils.is\_request\_type("LaunchRequest")(handler\_input)

def handle(self, handler\_input):

# type: (HandlerInput) -> Response

speak\_output = "I've sent your email"

port = 465 # For SSL

smtp\_server = "smtp.gmail.com"

# don't forget to import credentials

sender\_email = credentials.key['sender\_email'] # Enter your address

receiver\_email = credentials.key['receiver\_email'] # Enter receiver address

password = credentials.key['password'] # "given gmail password"

# send attachment

import email

from email import encoders

from email.mime.base import MIMEBase

from email.mime.multipart import MIMEMultipart

from email.mime.text import MIMEText

subject = "Email with attachment from Alexa"

body = "This is the body - the email has an attachment"

sender\_email = credentials.key['sender\_email']

receiver\_email = credentials.key['receiver\_email']

password = credentials.key['password']

# Create a multipart message and set headers

message = MIMEMultipart()

message["From"] = sender\_email

message["To"] = receiver\_email

message["Subject"] = subject

message["Bcc"] = receiver\_email # To send a blind copy

# Add body to email

message.attach(MIMEText(body, "plain"))

from utils import create\_presigned\_url

bucket\_name = os.environ.get('S3\_PERSISTENCE\_BUCKET')

object\_name = "Media/document.pdf" # saved in this code’s s3 bucket

s3\_client = boto3.client('s3')

# Download the file from S3 and save in tmp

filename = '/tmp/document.pdf'

s3\_client.download\_file(bucket\_name, object\_name , filename)

with open(filename, "rb") as attachment:

part = MIMEBase("application", "octet-stream") # unknown binary file

part.set\_payload(attachment.read())

# Encode file

encoders.encode\_base64(part)

# Add key/value pair header to attachment part

part.add\_header(

"Content-Disposition",

f"attachment; filename= {filename}",

)

# Add attachment to message

message.attach(part)

# and send message as string()

context = ssl.create\_default\_context()

with smtplib.SMTP\_SSL("smtp.gmail.com", 465, context=context) as server:

server.login(sender\_email, password)

server.sendmail(sender\_email, receiver\_email, message.as\_string())

return (

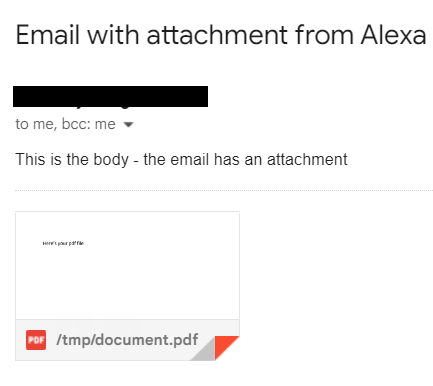
handler\_input.response\_builder

.speak(speak\_output)

.ask(speak\_output)

.response

)



1. **Using email-to**

Reference: https://pypi.org/project/email-to/

You can use email-to, to send a simple message. Note this uses port 587

You need to add email-to to requirements.txt so that it becomes:

boto3==1.9.216

ask-sdk-core==1.11.0

**email-to==0.1.0**

Add the following at the top of your code

import email-to

import credentials # if using credentials.py to hold your details

Change the launch request to the following:

class LaunchRequestHandler(AbstractRequestHandler):

#Handler for Skill Launch.

def can\_handle(self, handler\_input):

# type: (HandlerInput) -> bool

return ask\_utils.is\_request\_type("LaunchRequest")(handler\_input)

def handle(self, handler\_input):

# type: (HandlerInput) -> Response

speak\_output = "I've sent your email"

smtp\_server = "smtp.gmail.com"

# don't forget to import credentials

sender\_email = credentials.key['sender\_email'] # Enter your address

receiver\_email = credentials.key['receiver\_email'] # Enter receiver address

password = credentials.key['password'] # "given gmail password"

# Send simple text.

server = email-to.EmailServer(smtp\_server, 587, sender\_email, password)

server.quick\_email(receiver\_email, 'Subject: From Alexa',

['# A Heading', 'This message is sent using email-to from Alexa',

'<html>',

'<body>',

'<p>Hi,<br>',

'Here\'s some HTML<br>',

'<a href="tinyurl.com/y8dk523j">Here’s perkin</a>',

'</p>',

'</body>',

'</html>'

],

style='h1 {color: red}')

return (

handler\_input.response\_builder

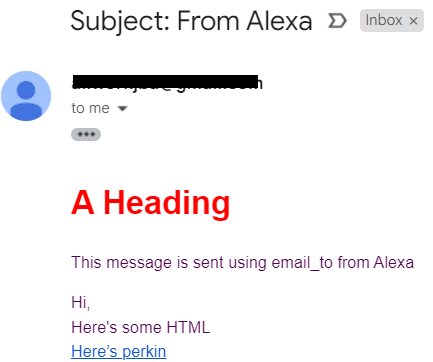
.speak(speak\_output)

.ask(speak\_output)

.response

)

Save and deploy and you should receive this:



That’s it. This doc and code at github

References:

<https://stackoverflow.com/questions/72478573/how-to-send-an-email-using-python-after-googles-policy-update-on-not-allowing-j>

<https://boto3.amazonaws.com/v1/documentation/api/latest/guide/s3-example-download-file.html>

<https://stackoverflow.com/questions/29378763/how-to-save-s3-object-to-a-file-using-boto3>

<https://realpython.com/python-send-email/>