Email Alerts on WhatsApp

Project Description

In our much busy schedule we usually don't have time to keep up with our emails, so we'll be working on a project to set up alerts of recent emails to WhatsApp.

Author

Pratyaksh Saini

Collaborator(s)

Ayush Kumar Shaw

Project Language(s)

JavaScript

Difficulty

Intermediate

Duration

15 hours

Prerequisite(s)

JavaScript, JavaScript Promises

Skills to be learned

Async/Await, Twilio Functions, Twilio WhatsApp Sandbox, IMAP library

Overview

Objective

You will be creating a workflow in Twilio which queries the requested email data, according to a given search criteria, and sends their details to WhatsApp.

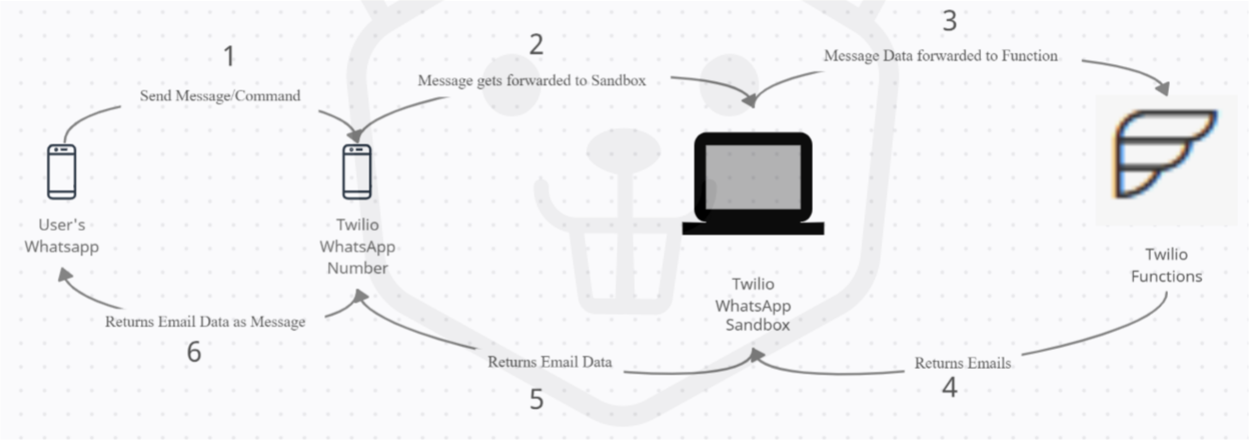
Project Context

Emails have become one of the most used digital communication mediums. But, sad truth, that since they are so aggressively used, it's quite difficult to keep up with them. Moreover, people keep on subscribing to new newsletters every now and then, which also adds to this cause.

So, in order to make our life a little bit easier, we can build a tool which will fetch us the various details from our mail box on query. Twilio is an efficient platform which provides us with the features needed to accomplish this. It's a message, email, call and notification automation tool/platform. We'll be utilizing some of it's features through this project.

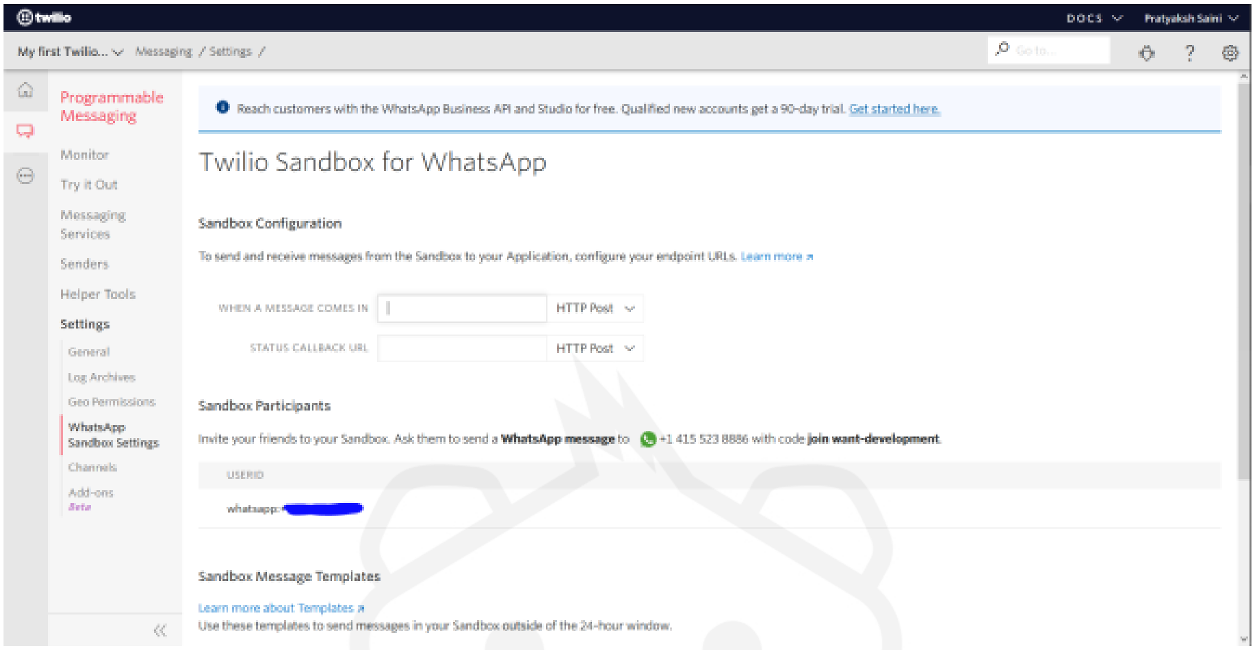
Project Workflow

Working principle of the project is as follows:



High-Level Approach

- Setup WhatsApp Sandbox in Twilio



Reach customers with the WhatsApp Business API and Studio for free. Qualified new accounts get a 90-day trial. Get started here.

Twilio Sandbox for WhatsApp

Sandbox Configuration

To send and receive messages from the Sandbox to your Application, configure your endpoint URLs. Learn more

WHEN A MESSAGE COMES IN: HTTP Post

STATUS CALLBACK URL: HTTP Post

Sandbox Participants

Invite your friends to your Sandbox. Ask them to send a **WhatsApp message** to +1 415 523 8886 with code **join want-development**.

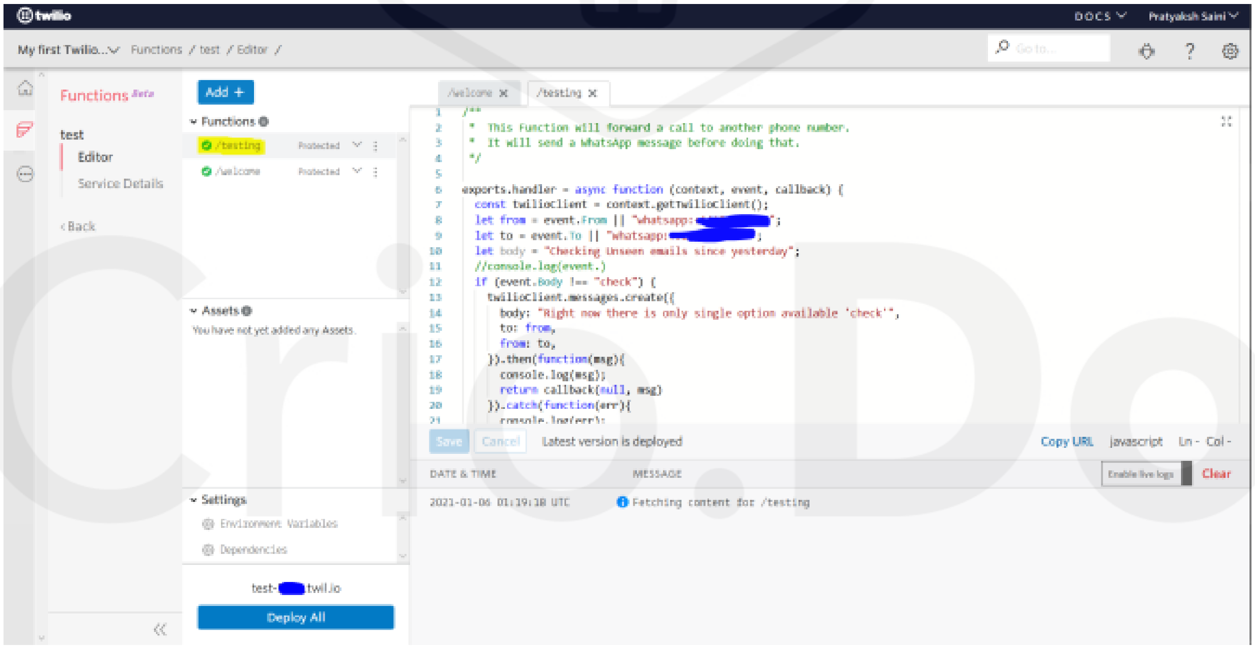
USERID
whatsapp:[redacted]

Sandbox Message Templates

Learn more about Templates

Use these templates to send messages in your Sandbox outside of the 24-hour window.

- Inside Twilio Functions Create New Service -> New Function



You have not yet added any Assets.

test-[redacted]twil.io

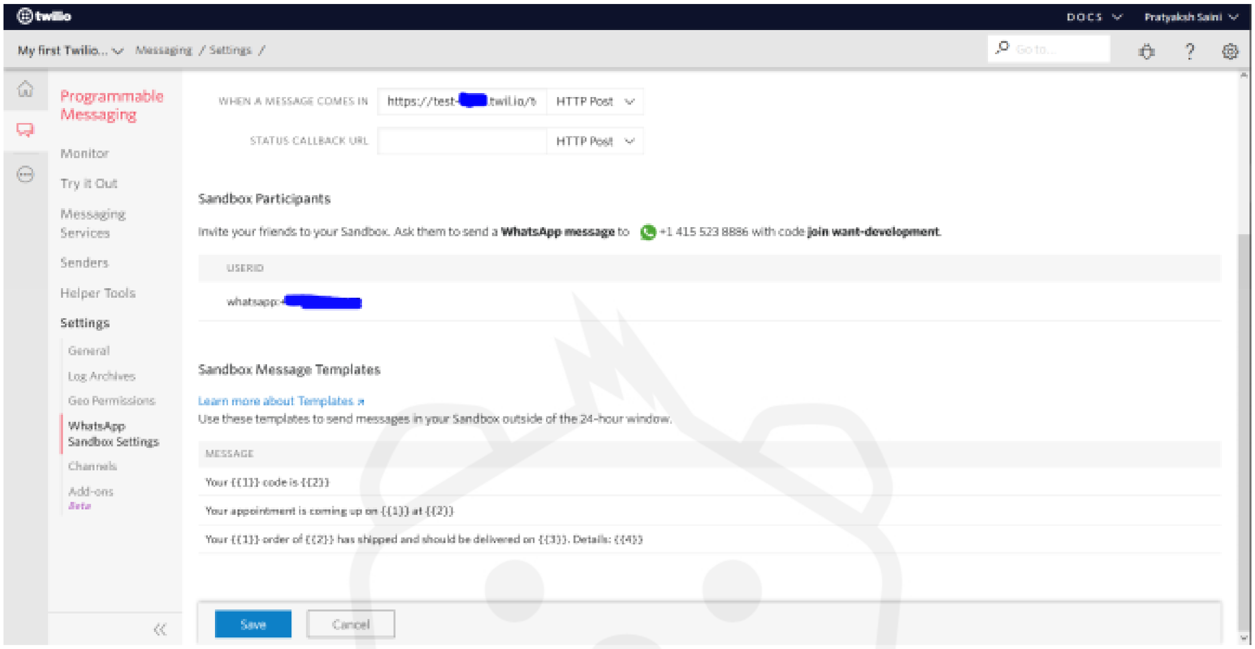
```
/**
 *  This function will forward a call to another phone number.
 *  It will send a WhatsApp message before doing that.
 */

exports.handler = async function (context, event, callback) {
  const twilioClient = context.getTwilioClient();
  let from = event.From || "whatsapp:[redacted]";
  let to = event.To || "whatsapp:[redacted];
  let body = "Checking Unseen emails since yesterday";
  //console.log(event.)
  if (event.Body !== "check") {
    twilioClient.messages.create({
      body: "Right now there is only single option available 'check'",
      to: from,
      from: to,
    }).then(function(msg){
      console.log(msg);
      return callback(null, msg)
    }).catch(function(err){
      console.log(err);
```

Latest version is deployed

```
DATE & TIME                 MESSAGE
2021-01-06 01:19:18 UTC     Fetching content for /testing
```

- Complete the Code in Functions and Save and Deploy them. This will create a REST API endpoint which will be used by WhatsApp Sandbox to operate.

- Configure WhatsApp Sandbox with functions URL



WHEN A MESSAGE COMES IN: https://test-[redacted].twil.io/t — HTTP Post

STATUS CALLBACK URL: — HTTP Post

Sandbox Participants

Invite your friends to your Sandbox. Ask them to send a **WhatsApp message** to +1 415 523 8886 with code **join want-development**.

USERID
whatsapp:+[redacted]

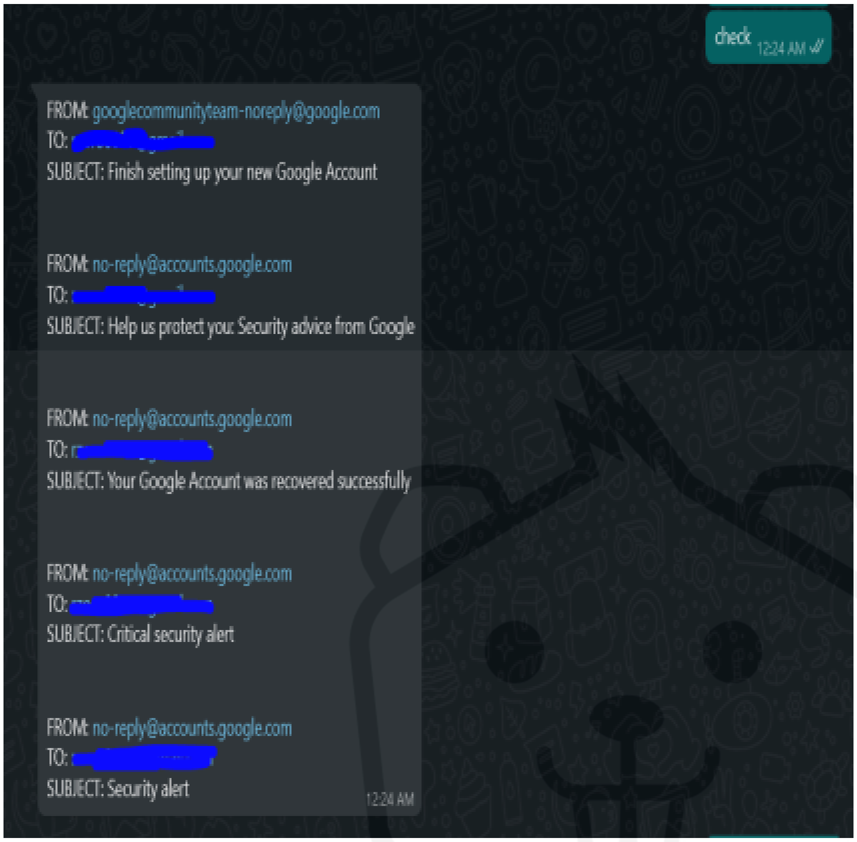
Sandbox Message Templates

Learn more about Templates

Use these templates to send messages in your Sandbox outside of the 24-hour window.

MESSAGE
Your {{1}} code is {{2}}
Your appointment is coming up on {{1}} at {{2}}
Your {{1}} order of {{2}} has shipped and should be delivered on {{3}}. Details: {{4}}

- Test it on WhatsApp.



check 12:24 AM

FROM: googlecommunityteam-noreply@google.com
TO: [redacted]
SUBJECT: Finish setting up your new Google Account

FROM: no-reply@accounts.google.com
TO: [redacted]
SUBJECT: Help us protect you: Security advice from Google

FROM: no-reply@accounts.google.com
TO: [redacted]
SUBJECT: Your Google Account was recovered successfully

FROM: no-reply@accounts.google.com
TO: [redacted]
SUBJECT: Critical security alert

FROM: no-reply@accounts.google.com
TO: [redacted]
SUBJECT: Security alert

12:24 AM

Primary goals

- Create a serverless REST API using Twilio Functions.
- Configure Twilio WhatsApp Sandbox.
- Keep testing and debugging using Twilio Functions Debugger.

Reference Material

- The Twilio Documentation
- Twilio on Stack Overflow
- The Twilio Blog

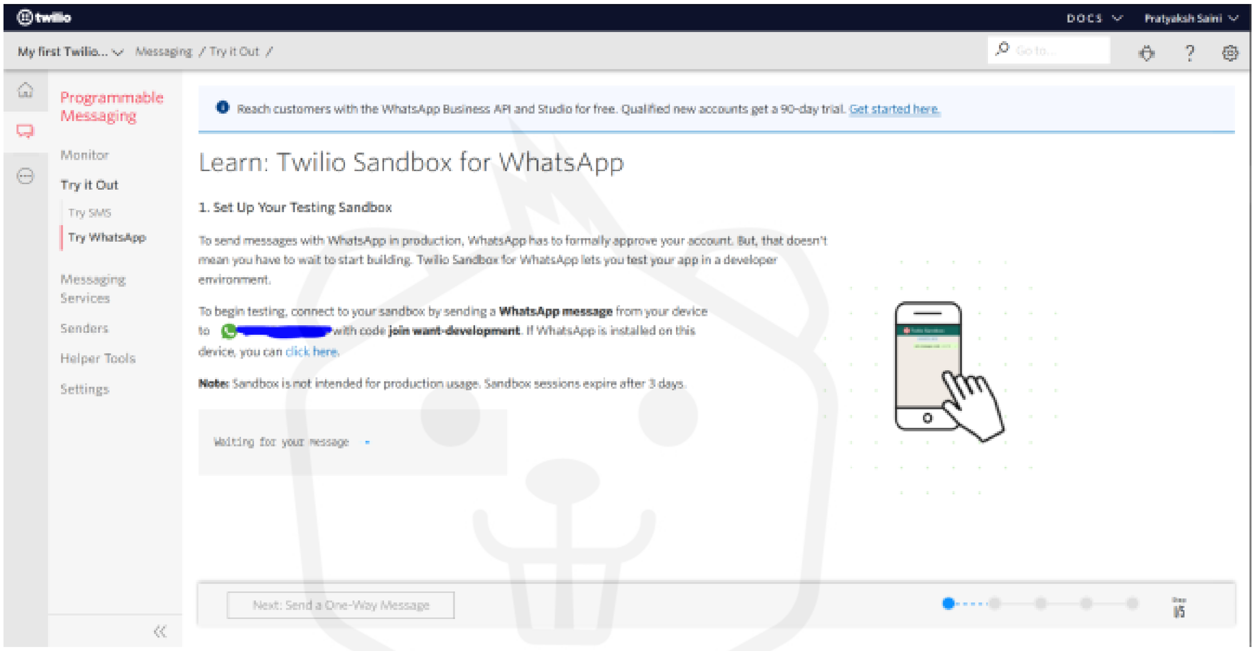
Task 1

Configuring Twilio WhatsApp Sandbox

The Twilio Sandbox for WhatsApp is a pre-configured environment available through the Twilio Console in which you can prototype sending outbound messages, replying to incoming messages, and configuring things like message delivery callbacks.

Requirements

- Go to `Try WhatsApp` in `Programmable Messaging` Section. You should see something like this



Reach customers with the WhatsApp Business API and Studio for free. Qualified new accounts get a 90-day trial. Get started here.

Learn: Twilio Sandbox for WhatsApp

1. Set Up Your Testing Sandbox

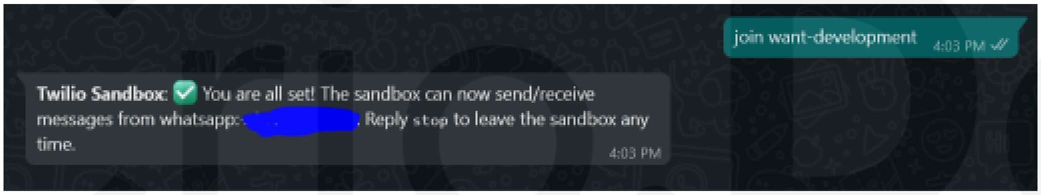
To send messages with WhatsApp in production, WhatsApp has to formally approve your account. But, that doesn't mean you have to wait to start building. Twilio Sandbox for WhatsApp lets you test your app in a developer environment.

To begin testing, connect to your sandbox by sending a **WhatsApp message** from your device to [redacted] with code **join want-development**. If WhatsApp is installed on this device, you can click here.

Note: Sandbox is not intended for production usage. Sandbox sessions expire after 3 days.

```
Waiting for your message
```

- After sending "join want-development" to the number, you should receive a confirmation like this



join want-development 4:03 PM

Twilio Sandbox: ✅ You are all set! The sandbox can now send/receive messages from whatsapp:[redacted]. Reply `stop` to leave the sandbox any time. 4:03 PM

- You will see 2 sections. The first is "Send a One-Way WhatsApp Message" and second is "Two-Way Messaging". We highly recommend trying them out and take a look at the code used in those examples. (To skip use Next button).
- When you are in Twilio Sandbox Configuration make sure that your WhatsApp number is in Sandbox Participants.
- You will need to configure WhatsApp sandbox every 3 days, If you want to avoid this:

- Buy a Phone Number on Twilio (costs about $1/month), and request WhatsApp Business Account on it.
- Create a automation bot and deploy it somewhere, which will configure this for you.

References

- Twilio WhatsApp - Documentation
- Twilio WhatsApp with Node.js
- Twilio Support - Documentation

Bring it On!

- Try to add multiple numbers in the Sandbox by just sending join text.

Expected Outcome

It is expected that you'll be able to set up the WhatsApp Sandbox. We highly recommend going through the code for one-way and two-way messaging.

Task 2

Using IMAP library to fetch emails.

- There are 2 libraries which can be use for operations related to Emails.
  - IMAP is used for reading the emails.
  - SMTP is used for sending emails to others.
- IMAP is the only one you need to know for this project, but after completion you can also add functionality to send emails using SMTP.
- We utilized IMAP-simple which is wrapped on imap-node library. We used it because of ease of functionality. But you can use any of them.
- If you get an error in starting with code `DEPTH_ZERO_SELF_SIGNED_CERT`, there's a work around for that: add `tlsOptions: { rejectUnauthorized: false }`property in the `config` object along with user, password...etc.

Requirements

- If you are using Gmail, you will need to either get `App Passwords` (recommended) or disable 2 way auth and turn on less secure access.
  - We recommend using App-password because that way you won't have to remove 2 way auth. Click here to see steps to create `App Passwords`.
  - Guide for Less Secure Access.
- Whether you are using `App Passwords` or `Less Secure Access`, you must not put password in the main code file (In case your intent is to put code on GitHub or anywhere else publicly).
- You will need to code an async function that will use IMAP library methods to fetch the email data and print it or maybe return it.

- We recommend first making a working async function on your local machine, then copying it to the Twilio Functions, because debugging there is a bit difficult.

References

- IMAP-simple docs
- Main IMAP library
- Cool dev.to blog on IMAP

Bring it On!

- Try exploring different ways of using search criteria (which is first parameter of connection.search method).

Expected Outcome

On using your local machine for coding the functions you must see the emails corresponding to the search criteria as output of the function.

Task 3

Intro to Twilio Functions

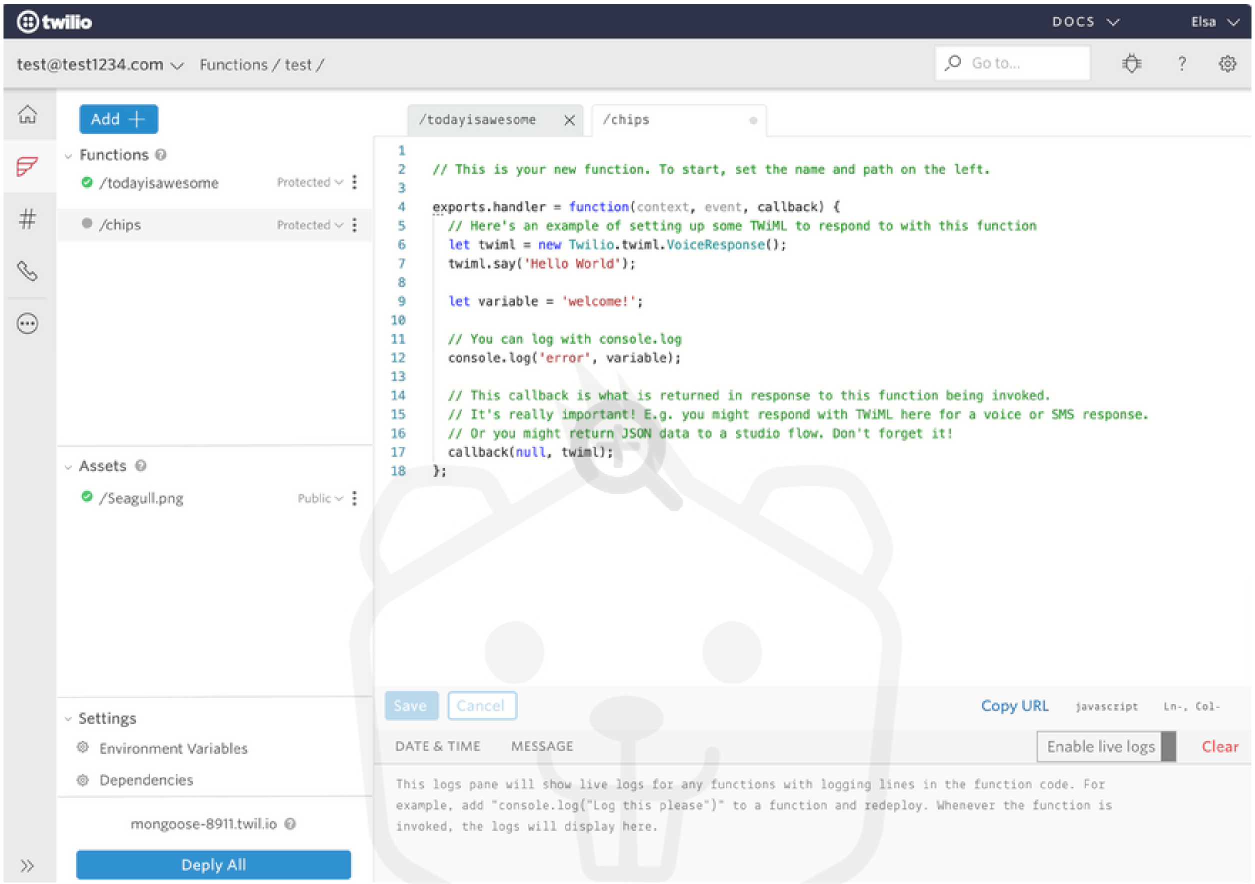
Twilio Functions is a serverless environment where developers can directly write code and access the code from an API endpoint for free.

Requirements

- First create a Twilio service in Twilio Functions available under Runtime section in Main Menu.

- After creating a Twilio service, you can add functions in it, which will be separate endpoints for the API.

In the image you can see 2 endpoints (`/todayisawesome`, `/chips`) and 1 asset (`Seagull.png`).



```

// This is your new function. To start, set the name and path on the left.

exports.handler = function(context, event, callback) {
  // Here's an example of setting up some TWiML to respond to with this function
  let twiml = new Twilio.twiml.VoiceResponse();
  twiml.say('Hello World');

  let variable = 'welcome!';

  // You can log with console.log
  console.log('error', variable);

  // This callback is what is returned in response to this function being invoked.
  // It's really important! E.g. you might respond with TWiML here for a voice or SMS response.
  // Or you might return JSON data to a studio flow. Don't forget it!
  callback(null, twiml);
};
```

```
This logs pane will show live logs for any functions with logging lines in the function code. For
example, add "console.log("Log this please")" to a function and redeploy. Whenever the function is
invoked, the logs will display here.
```

- You will need to add all the required libraries in `Dependencies` from the `Settings` section.

- We strongly recommend adding sensitive information like passwords into Environment Variables (You need to find out yourself how to access ENV variables in the code).

- As you can see the above image that the URL of service is **mongoose-8911.twil.io**, and suppose the function we created is `/chips`, so we need to put **mongoose-8911.twil.io/chips** in `When a message comes` in input box of WhatsApp Sandbox Configuration (Don't forget to SAVE).

- Write a code in the function which will send "Hello How are you?" back if the user messages "Hello", other wise send "Not Found" back.

- If you are facing any problems, jump to the last task to know how debugging works, then come back.

References

- Twilio Functions for WhatsApp

Expected Outcome

After completion of this task it is expected to have the functionality that when you send the message "Hello" on WhatsApp to the Twilio number, it sends back "Hello How are you", other wise it sends back "Not Found".

Task 4

Integrating the IMAP function into Twilio Functions

Integration will enable Twilio Function to fetch email data and send it to configured WhatsApp Number.

Requirements

- In this task we will put the async IMAP function that we created and debugged, on our local machine, into the Twilio Functions.
- Edit some code from Task-3, which sends "Hello How are you". Now it should send the string data returned from IMAP function.
- For starters, only send back `From`, `To` and `Subject` data, as there is a 1600 character limit on the message text.

Bring it On!

- Try to add more option eg if user sends text "check unseen" then it returns all the unseen messages.
- If message is greater than 1600 characters, divide the message in parts (in case of attempting to fetch email body).
- Try to implement a workflow such the Twilio itself sends a message whenever an email comes, instead of querying for the same.

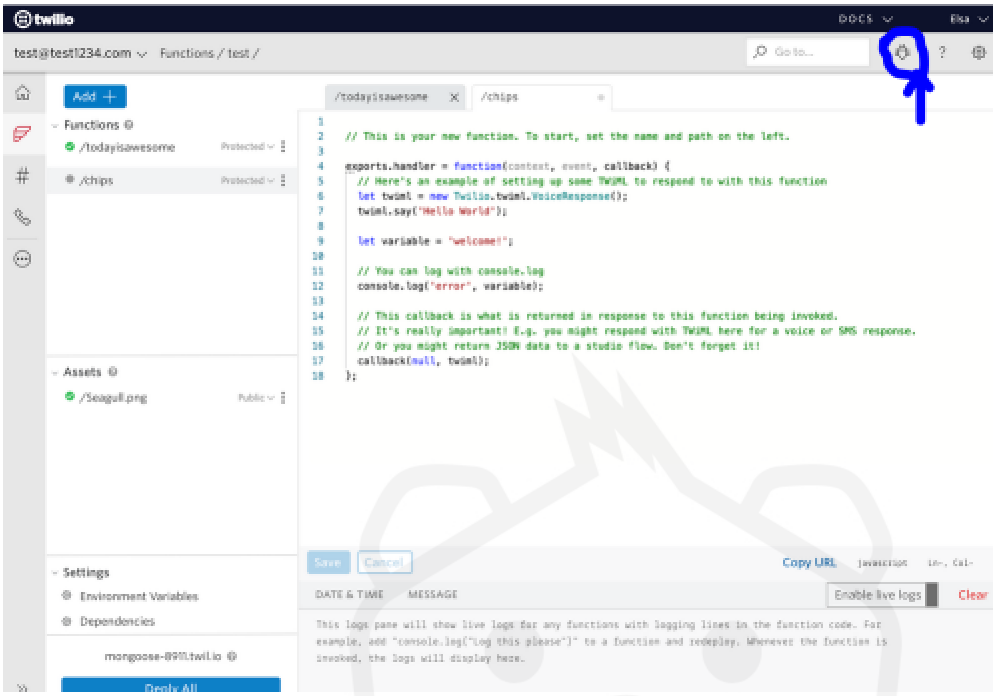
Expected Outcome

After successful completion of this task you should be able to get the Email details on your WhatsApp.

Task 5

How Debugging Works

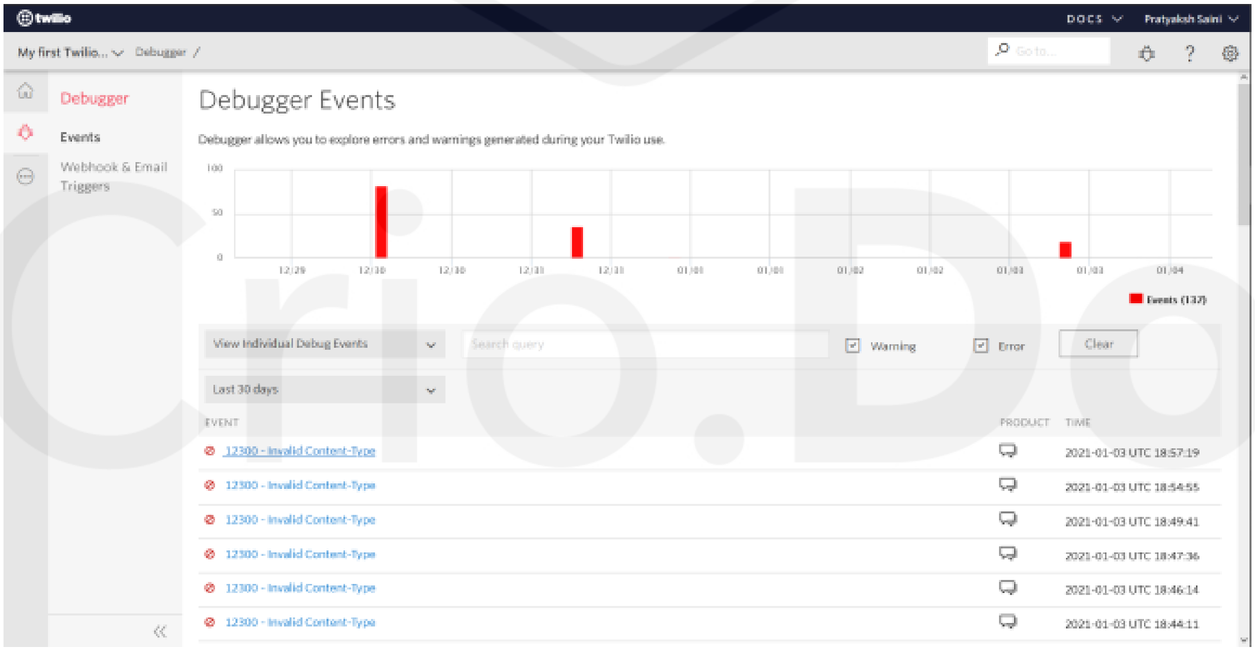
You can see the Debugger icon ("Bug") in the top right corner of Twilio function window



```
// This is your new function. To start, set the name and path on the left.

exports.handler = function(context, event, callback) {
  // Here's an example of setting up some TWiML to respond to with this function
  let twiml = new Twilio.twiml.VoiceResponse();
  twiml.say('Hello World');

  let variable = 'welcome!';

  // You can log with console.log
  console.log('error', variable);

  // This callback is what is returned in response to this function being invoked.
  // It's really important! E.g. you might respond with TWiML here for a voice or SMS response.
  // Or you might return JSON data to a studio flow. Don't forget it!
  callback(null, twiml);
};
```

This logs pane will show live logs for any functions with logging lines in the function code. For example, add "console.log("log this please")" to a function and redeploy. Whenever the function is invoked, the logs will display here.

It looks like this



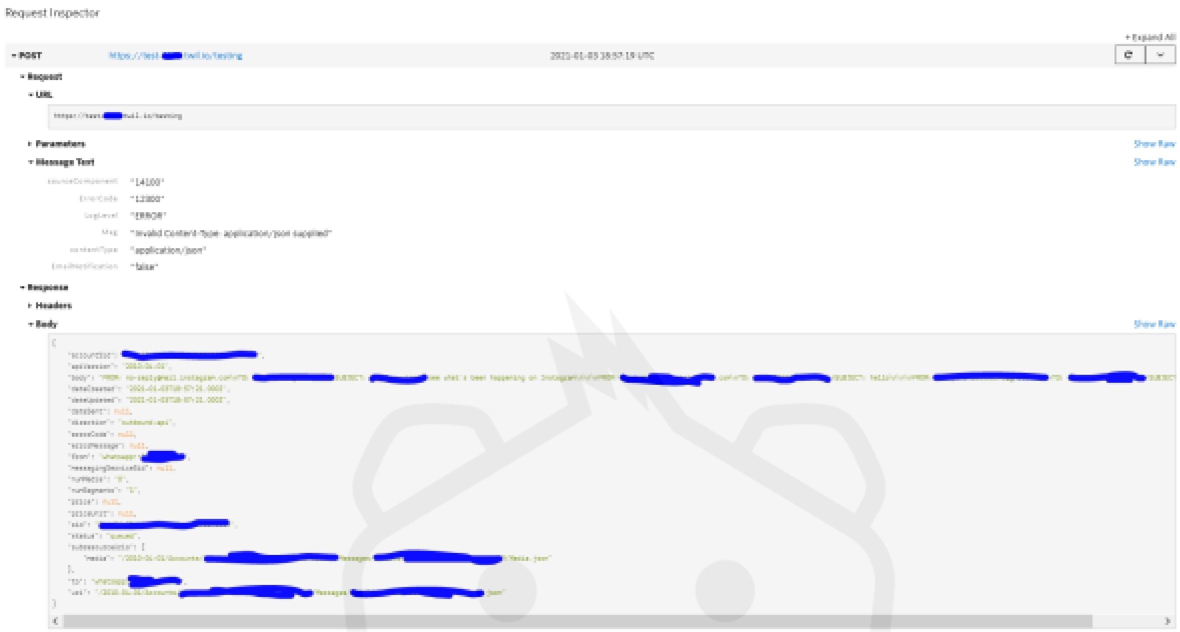
Debugger Events

Debugger allows you to explore errors and warnings generated during your Twilio use.

☑ Warning ☑ Error

EVENT	PRODUCT	TIME
12300 - Invalid Content-Type		2021-01-03 UTC 18:57:19
12300 - Invalid Content-Type		2021-01-03 UTC 18:54:55
12300 - Invalid Content-Type		2021-01-03 UTC 18:49:41
12300 - Invalid Content-Type		2021-01-03 UTC 18:47:36
12300 - Invalid Content-Type		2021-01-03 UTC 18:46:14
12300 - Invalid Content-Type		2021-01-03 UTC 18:44:11

You can see Response and Request body of every event.



Request Inspector

+ Expand All

▾ POST https://test-[redacted].twil.io/testing 2021-01-03 18:57:19 UTC

▾ Request

▾ URL

```
https://test-[redacted]twil.io/testing
```

▸ Parameters Show Raw

▾ Message Text Show Raw

sourceComponent	"14100"
ErrorCode	"12300"
LogLevel	"ERROR"
Msg	"Invalid Content-Type: application/json supplied"
contentType	"application/json"
EmailNotification	"false"

▾ Response

▸ Headers

▾ Body Show Raw

```
{
  "accountSid": [redacted],
  "apiVersion": "2010-04-01",
  "body": "FROM: no-reply@mail.instagram.com\r\nTO: [redacted]SUBJECT: [redacted]See what's been happening on Instagram\r\n\r\nFROM: [redacted].com\r\nTO: [redacted]SUBJECT: hello\r\n\r\nFROM: [redacted]TO: [redacted]SUBJECT
  "dateCreated": "2021-01-03T18:57:21.000Z",
  "dateUpdated": "2021-01-03T18:57:21.000Z",
  "dateSent": null,
  "direction": "outbound-api",
  "errorCode": null,
  "errorMessage": null,
  "from": "whatsapp:+[redacted],
  "messagingServiceSid": null,
  "numMedia": "0",
  "numSegments": "1",
  "price": null,
  "priceUnit": null,
  "sid": [redacted],
  "status": "queued",
  "subresourceUris": {
     "media": "/2010-04-01/Accounts/[redacted]/Messages/[redacted]/Media.json"
  },
  "to": "whatsapp:[redacted],
  "uri": "/2010-04-01/Accounts/[redacted]/Messages/[redacted].json"
}
```

Requirements

- Try to use the debugger for efficient error resolving if needed.

Tip

- The main part is the Request Inspector inside an Event, which contains the body of response.
- All error occurred during an API call can be seen here.
- All the console.logs can be seen inside the Response's body.

Bring it On!

- If your project is big, you can set up a function which will send all logs as a file (maybe try to send the logs to your email using SMTP library).

Expected Outcome

You are expected to efficiently debug Twilio Functions if required.