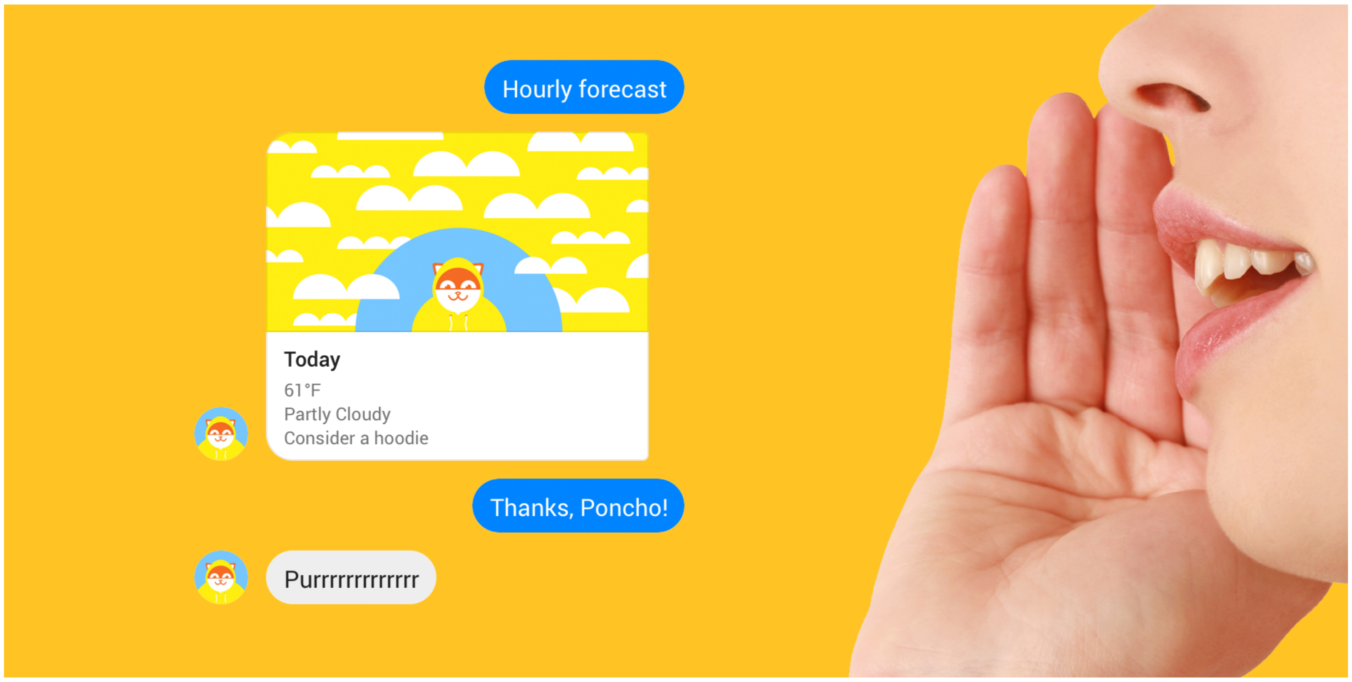


The Secret To Making Your Own Facebook Messenger Bot In Less Than 15 Minutes

Read this if you want to make a bot.

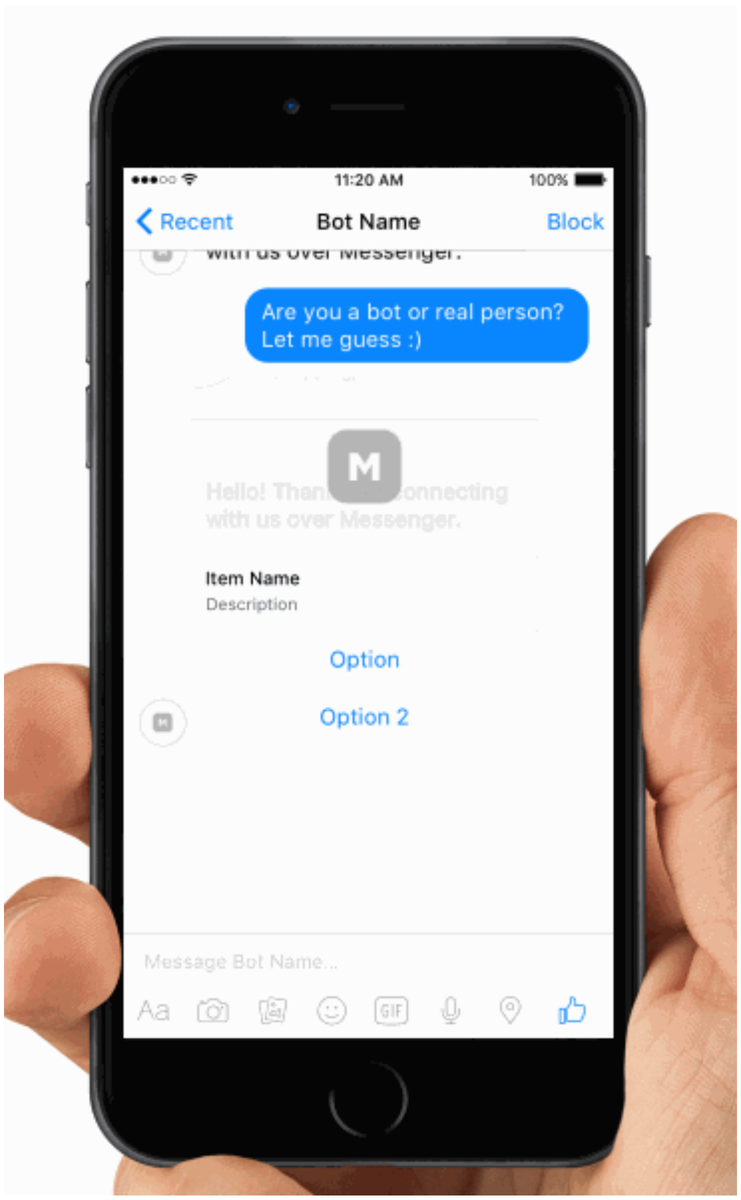
So, you want to make a bot. **You've come to the right place.**

This is a step by step guide to making your own Facebook Messenger bot in less than 15 minutes.

There are three basic steps, and I'm going to break them all down for you in excruciating detail. **If you literally just do everything I tell you then you will be able to make a bot.**

- Setting up your environment.
- Setting up a Facebook Application.
- Setting up your Facebook Messenger Bot.

Let’s get to it.


11:20 AM
100%
Recent
Bot Name
Block
Are you a bot or real person? Let me guess :)
Hello! Thanks for connecting with us over Messenger.
Item Name
Description
Option
Option 2
Message Bot Name...

Step 1. Set up your environment.

Messenger bots uses a web server to process messages it receives or to figure out what messages to send. You also need to have the bot be authenticated to speak with the web server and the bot approved by Facebook to speak with the public.

> *You can also skip the whole thing by cloning this git repository here, running npm install, and run a server somewhere.*

Build the server

Install the Heroku toolbelt from here https://toolbelt.heroku.com to launch, stop and monitor instances. Sign up for free at https://www.heroku.com if you don't have an account yet.

Install Node from here https://nodejs.org, this will be the server environment. Then open up Terminal or Command Line Prompt and make sure you've got the very most recent version of npm by installing it again:

```
sudo npm install npm --global
```

3. Create a new folder somewhere and let's create a new Node project. Hit Enter to accept the defaults.

```
npm init
```

4. Install the additional Node dependencies. Express is for the server, request is for sending out messages and body-parser is to process messages.

```
npm install express request body-parser --save
```

5. Create an index.js file in the folder and copy this into it. We will start by authenticating the bot.

```
var express = require('express')
var bodyParser = require('body-parser')
var request = require('request')
var app = express()

app.set('port', (process.env.PORT || 5000))

// Process application/x-www-form-urlencoded
app.use(bodyParser.urlencoded({extended: false}))

// Process application/json
app.use(bodyParser.json())

// Index route
app.get('/', function (req, res) {
    res.send('Hello world, I am a chat bot')
})

// for Facebook verification
app.get('/webhook/', function (req, res) {
    if (req.query['hub.verify_token'] ===
'my_voice_is_my_password_verify_me') {
        res.send(req.query['hub.challenge'])
    }
    res.send('Error, wrong token')
})

// Spin up the server
app.listen(app.get('port'), function() {
    console.log('running on port', app.get('port'))
})
```

6. Make a file called Procfile and copy this. This is so Heroku can know what file to run.

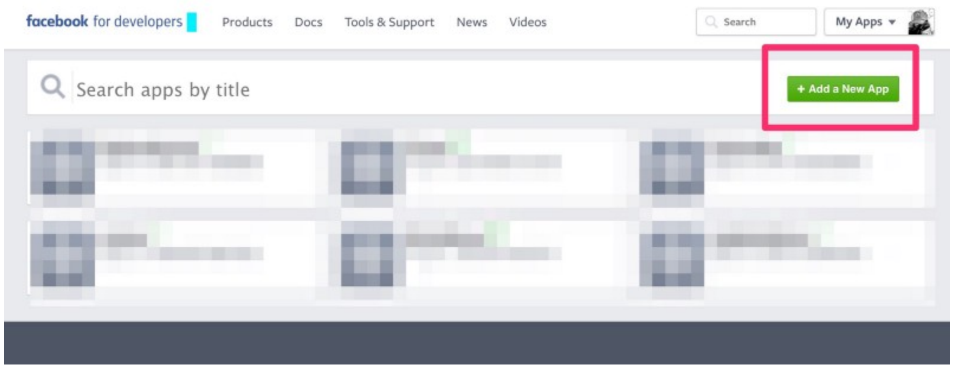
```
web: node index.js
```

7. Commit all the code with Git then create a new Heroku instance and push the code to the cloud.

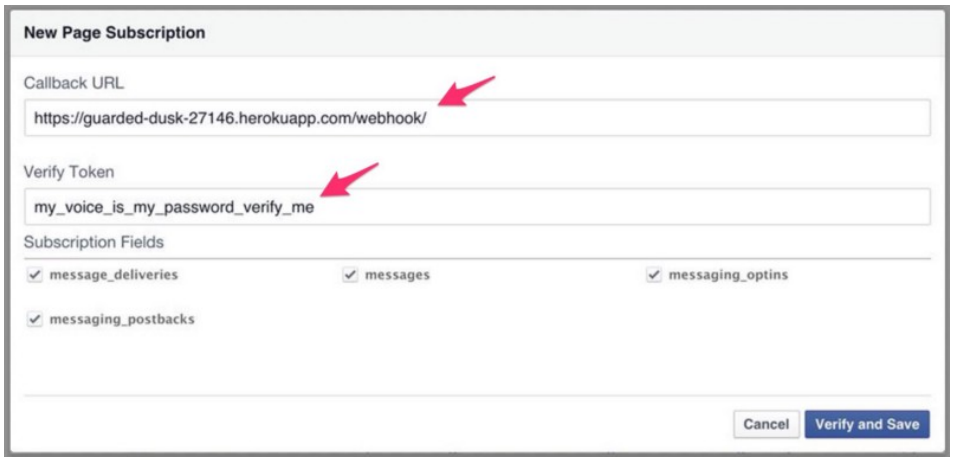
```
git init
git add .
git commit --message 'hello world'
heroku create
git push heroku master
```

Step 2. Set up your Facebook App

> Create or configure a Facebook App or Page here
> https://developers.facebook.com/apps/



2. In the app go to Messenger tab then click Setup Webhook. Here you will put in the URL of your Heroku server and a token. Make sure to check all the subscription fields.



New Page Subscription

Callback URL: https://guarded-dusk-27146.herokuapp.com/webhook/

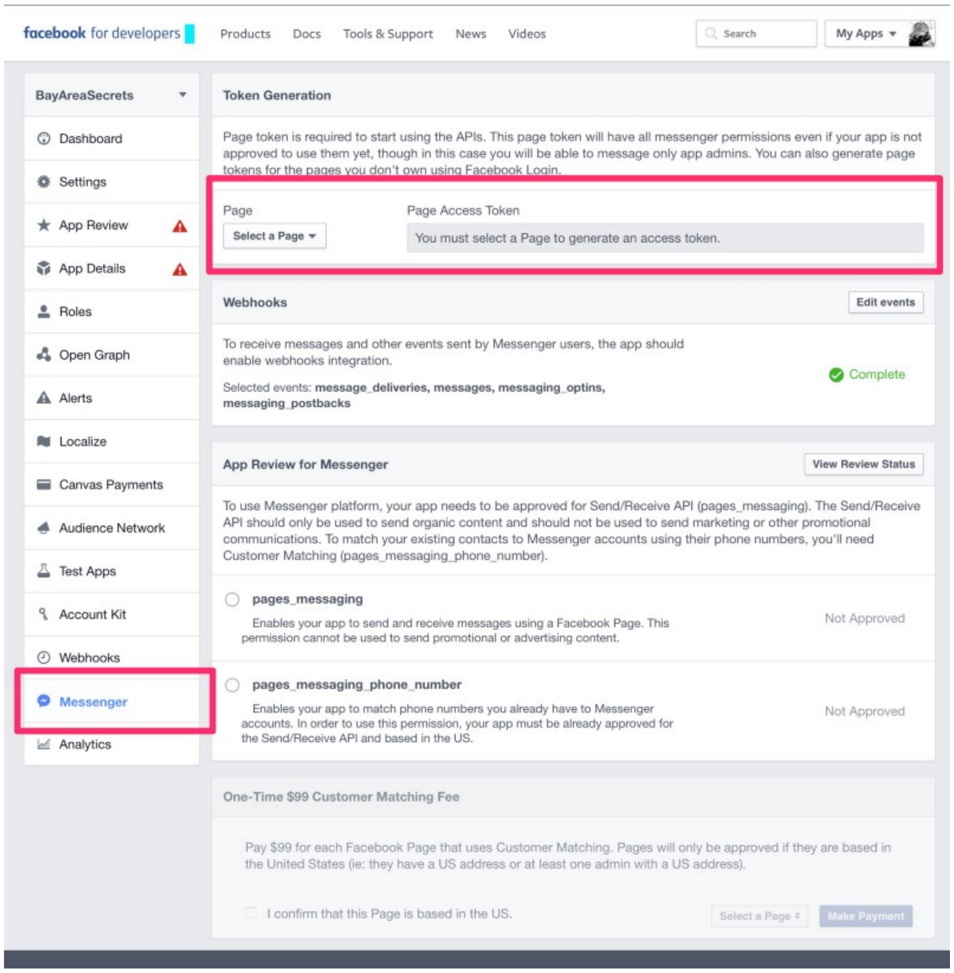
Verify Token: my_voice_is_my_password_verify_me

Subscription Fields

✓ message_deliveries ✓ messages ✓ messaging_optins

✓ messaging_postbacks

3. Get a Page Access Token and save this somewhere.



Token Generation

Page token is required to start using the APIs. This page token will have all messenger permissions even if your app is not approved to use them yet, though in this case you will be able to message only app admins. You can also generate page tokens for the pages you don't own using Facebook Login.

Page: Select a Page
Page Access Token: You must select a Page to generate an access token.

Webhooks — Edit events

To receive messages and other events sent by Messenger users, the app should enable webhooks integration.

Selected events: **message_deliveries, messages, messaging_optins, messaging_postbacks** — Complete

App Review for Messenger — View Review Status

To use Messenger platform, your app needs to be approved for Send/Receive API (pages_messaging). The Send/Receive API should only be used to send organic content and should not be used to send marketing or other promotional communications. To match your existing contacts to Messenger accounts using their phone numbers, you'll need Customer Matching (pages_messaging_phone_number).

pages_messaging — Not Approved
Enables your app to send and receive messages using a Facebook Page. This permission cannot be used to send promotional or advertising content.

pages_messaging_phone_number — Not Approved
Enables your app to match phone numbers you already have to Messenger accounts. In order to use this permission, your app must be already approved for the Send/Receive API and based in the US.

One-Time $99 Customer Matching Fee

Pay $99 for each Facebook Page that uses Customer Matching. Pages will only be approved if they are based in the United States (ie: they have a US address or at least one admin with a US address).

I confirm that this Page is based in the US. — Select a Page — Make Payment

4. Go back to Terminal and type in this command to trigger the Facebbook app to send messages. Remember to use the token you requested earlier.

```
curl -X POST "https://graph.facebook.com/v2.6/me
/subscribed_apps?access_token=<PAGE_ACCESS_TOKEN>"
```

Step 3. Setup your Facebook Messenger Bot

Now that Facebook and Heroku can talk to each other we can code out the rest of the bot.

Add an API endpoint to index.js to process messages. Remember to also include the token we got earlier.

```
app.post('/webhook/', function (req, res) {
    messaging_events = req.body.entry[0].messaging
    for (i = 0; i < messaging_events.length; i++) {
        event = req.body.entry[0].messaging[i]
        sender = event.sender.id
        if (event.message && event.message.text) {
            text = event.message.text
            sendTextMessage(sender, "Text received, echo: " +
text.substring(0, 200))
        }
    }
    res.sendStatus(200)
})

var token = "<PAGE_ACCESS_TOKEN>"
```

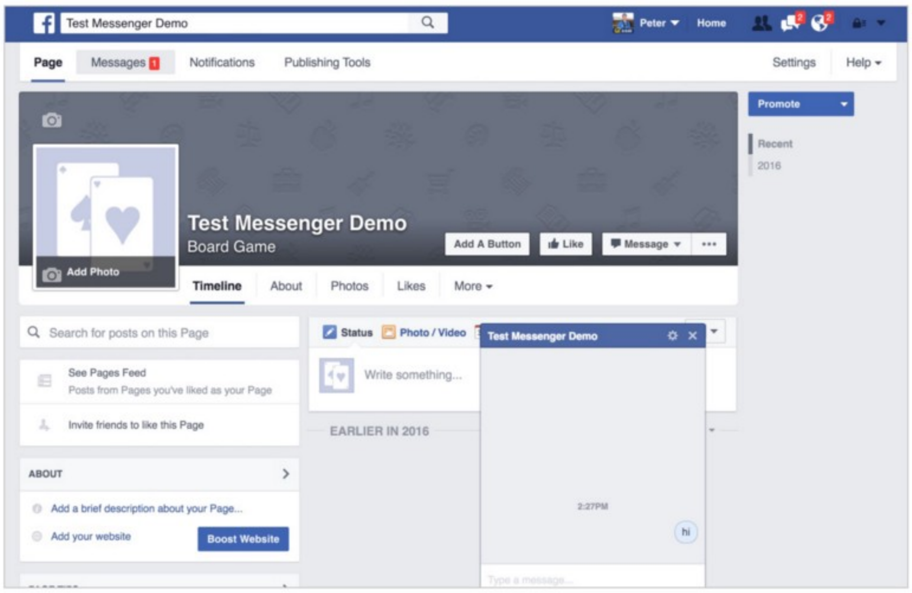
2. Add a function to echo back messages

```
function sendTextMessage(sender, text) {
    messageData = {
        text:text
    }
    request({
        url: 'https://graph.facebook.com/v2.6/me/messages',
        qs: {access_token:token},
        method: 'POST',
        json: {
            recipient: {id:sender},
            message: messageData,
        }
    }, function(error, response, body) {
        if (error) {
            console.log('Error sending messages: ', error)
        } else if (response.body.error) {
            console.log('Error: ', response.body.error)
        }
    })
}
```

3. Commit the code again and push to Heroku

```
git add .
git commit --message 'updated the bot to speak'
git push heroku master
```

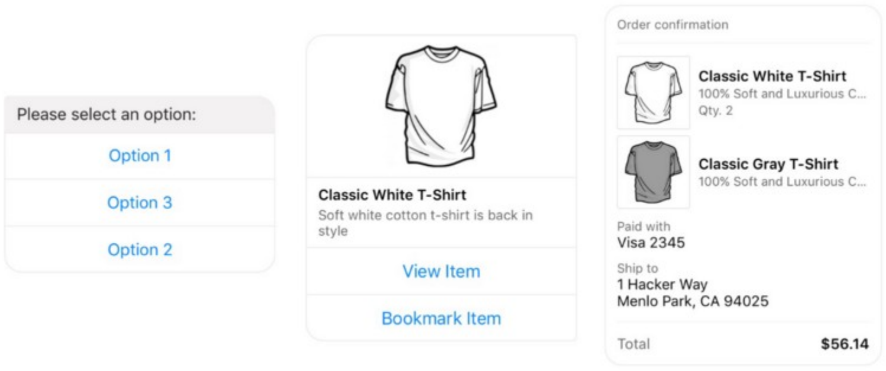
4. Go to the Facebook Page and click on Message to start chatting!



⚙ Customize what the bot says

Send a Structured Message

Facebook Messenger can send messages structured as cards or buttons.



Copy the code below to index.js to send an test message back as two cards.

```
function sendGenericMessage(sender) {
    messageData = {
        "attachment": {
            "type": "template",
            "payload": {
                "template_type": "generic",
                "elements": [{
                    "title": "First card",
                    "subtitle": "Element #1 of an hscroll",
                    "image_url": "http://messengerdemo.parseapp.com
/img/rift.png",
                    "buttons": [{
                        "type": "web_url",
                        "url": "https://www.messenger.com",
                        "title": "web url"
                    }, {
                        "type": "postback",
                        "title": "Postback",
                        "payload": "Payload for first element in a
generic bubble",
                    }],
                }, {
                    "title": "Second card",
                    "subtitle": "Element #2 of an hscroll",
                    "image_url": "http://messengerdemo.parseapp.com
/img/gearvr.png",
                    "buttons": [{
                        "type": "postback",
                        "title": "Postback",
                        "payload": "Payload for second element in a
generic bubble",
                    }],
                }]
            }
        }
    }
    request({
        url: 'https://graph.facebook.com/v2.6/me/messages',
        qs: {access_token:token},
        method: 'POST',
        json: {
            recipient: {id:sender},
            message: messageData,
        }
    }, function(error, response, body) {
        if (error) {
            console.log('Error sending messages: ', error)
        } else if (response.body.error) {
            console.log('Error: ', response.body.error)
        }
    })
}
```

2. Update the webhook API to look for special messages to trigger the cards

```
app.post('/webhook/', function (req, res) {
    messaging_events = req.body.entry[0].messaging
    for (i = 0; i < messaging_events.length; i++) {
        event = req.body.entry[0].messaging[i]
        sender = event.sender.id
        if (event.message && event.message.text) {
            text = event.message.text
            if (text === 'Generic') {
                sendGenericMessage(sender)
                continue
            }
            sendTextMessage(sender, "Text received, echo: " +
text.substring(0, 200))
        }
    }
    res.sendStatus(200)
})
```

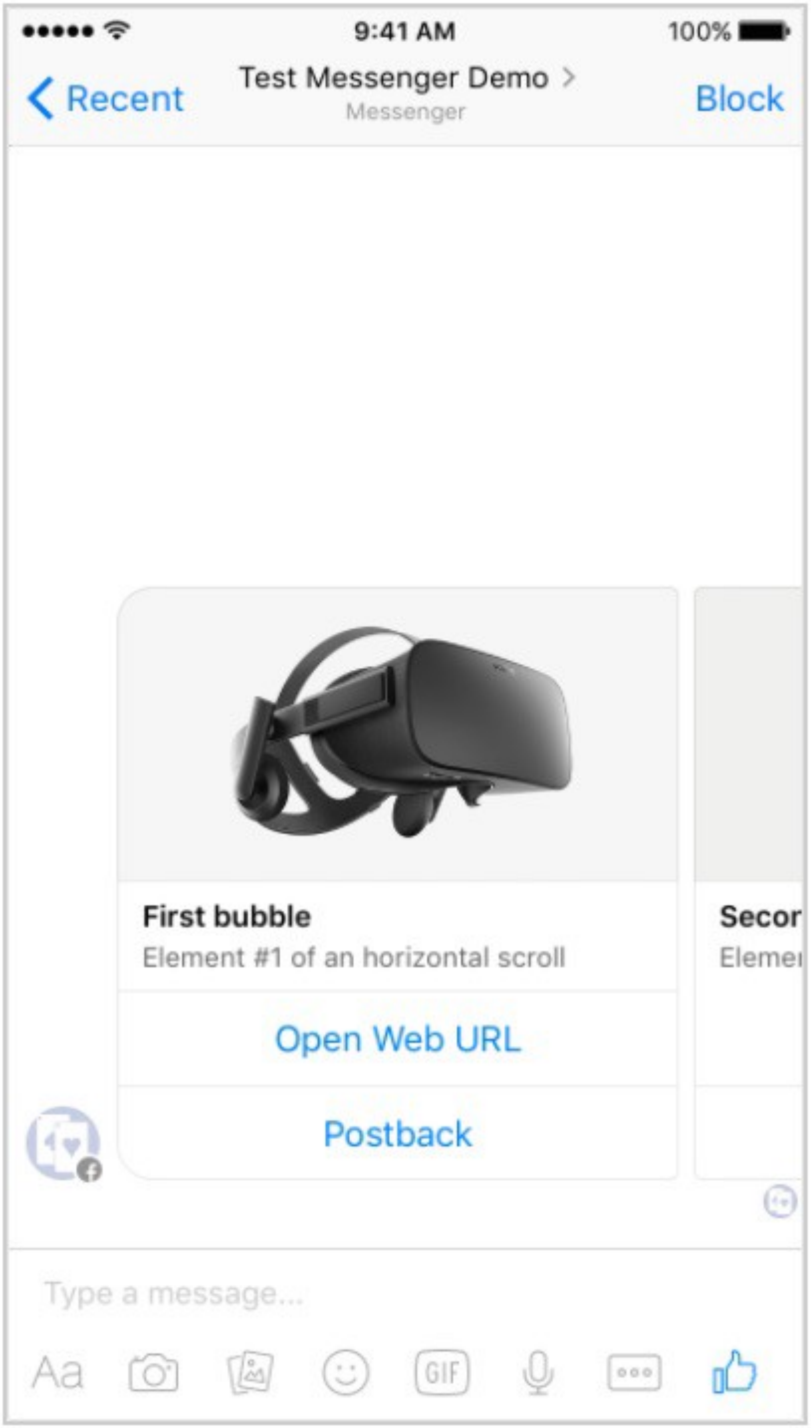
Act on what the user messages

What happens when the user clicks on a message button or card though? Let's update the webhook API one more time to send back a postback function.

```
app.post('/webhook/', function (req, res) {
    messaging_events = req.body.entry[0].messaging
    for (i = 0; i < messaging_events.length; i++) {
        event = req.body.entry[0].messaging[i]
        sender = event.sender.id
        if (event.message && event.message.text) {
            text = event.message.text
            if (text === 'Generic') {
                sendGenericMessage(sender)
                continue
            }
            sendTextMessage(sender, "Text received, echo: " +
text.substring(0, 200))
        }
        if (event.postback) {
            text = JSON.stringify(event.postback)
            sendTextMessage(sender, "Postback received:
"+text.substring(0, 200), token)
            continue
        }
    }
    res.sendStatus(200)
})
```

Git add, commit, and push to Heroku again.

Now when you chat with the bot and type ‘Generic’ you can see this.



How to share your bot

Add a chat button to your webpage

Go here to learn how to add a chat button your page.

Create a shortlink

You can use https://m.me/<PAGE_USER_NAME> to have someone start a chat.

What's next?

So, you made a bot. Now what? Don't worry! The fun is only just beginning.

- Join the official Chatbot group on Facebook.
- Learn how to get your bot approved for public use here.
- Connect an AI brain to your bot here
- Read about all things chat bots with the Chatbots Magazine here
- Design Messenger bots in Sketch with the Bots UI Kit!

Don't forget to follow me and the Chatbot magazine!

. . .

Want to chat about bots? Click here to hit me up on Twitter right this moment, I'd love to talk to you.

. . .

⚡The short link for this tutorial is http://j.mp/fb-chatbot-tutorial

😎 *Need help? Jerry reads every response on Medium so just reply and ask!*

👀 *By the way I run a bot factory over at http://www.capbots.com*