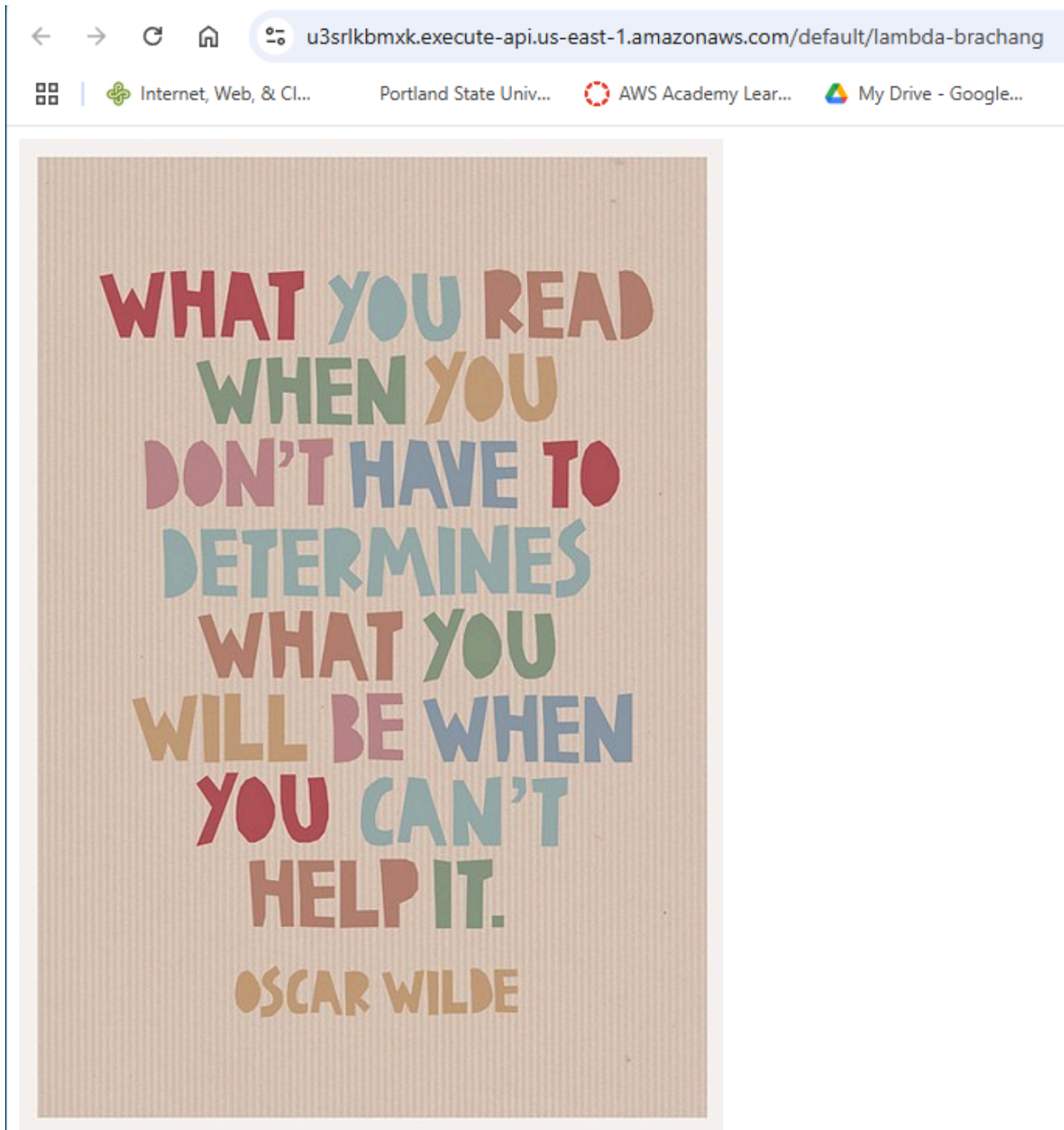


<b>8.1a: API Gateway</b> .....	<b>2</b>
4. Test code.....	2
8. Test code.....	4
<b>8.2a: Lambda, API Gateway Guestbook</b> .....	<b>4</b>
10. Deploy API to production and view entries.....	4
12. API endpoint for signing (2).....	4
<b>8.2g: Cloud Functions, API Gateway Guestbook</b> .....	<b>6</b>
10. Create the API Gateway.....	6
11. Test the API via Python Requests (GET).....	6
12. Test the API via Python Requests (POST).....	7
15. Version #1: Local file system.....	7
16. Version #2: Google Cloud Storage bucket.....	8
<b>8.3g OAuth2 Guestbook</b> .....	<b>9</b>
14. Visit the application.....	9
15. Secrets manager deployment.....	12
16. Removing access.....	13
<b>8.4g: Firebase</b> .....	<b>13</b>
4. Authentication setup.....	13
8. Bundling with Webpack.....	14
16. Test application with text messaging.....	15
17. Manual message insertion.....	16
18. Add image messaging.....	16
19. Test application with image messaging.....	17
20. Deploy application.....	18

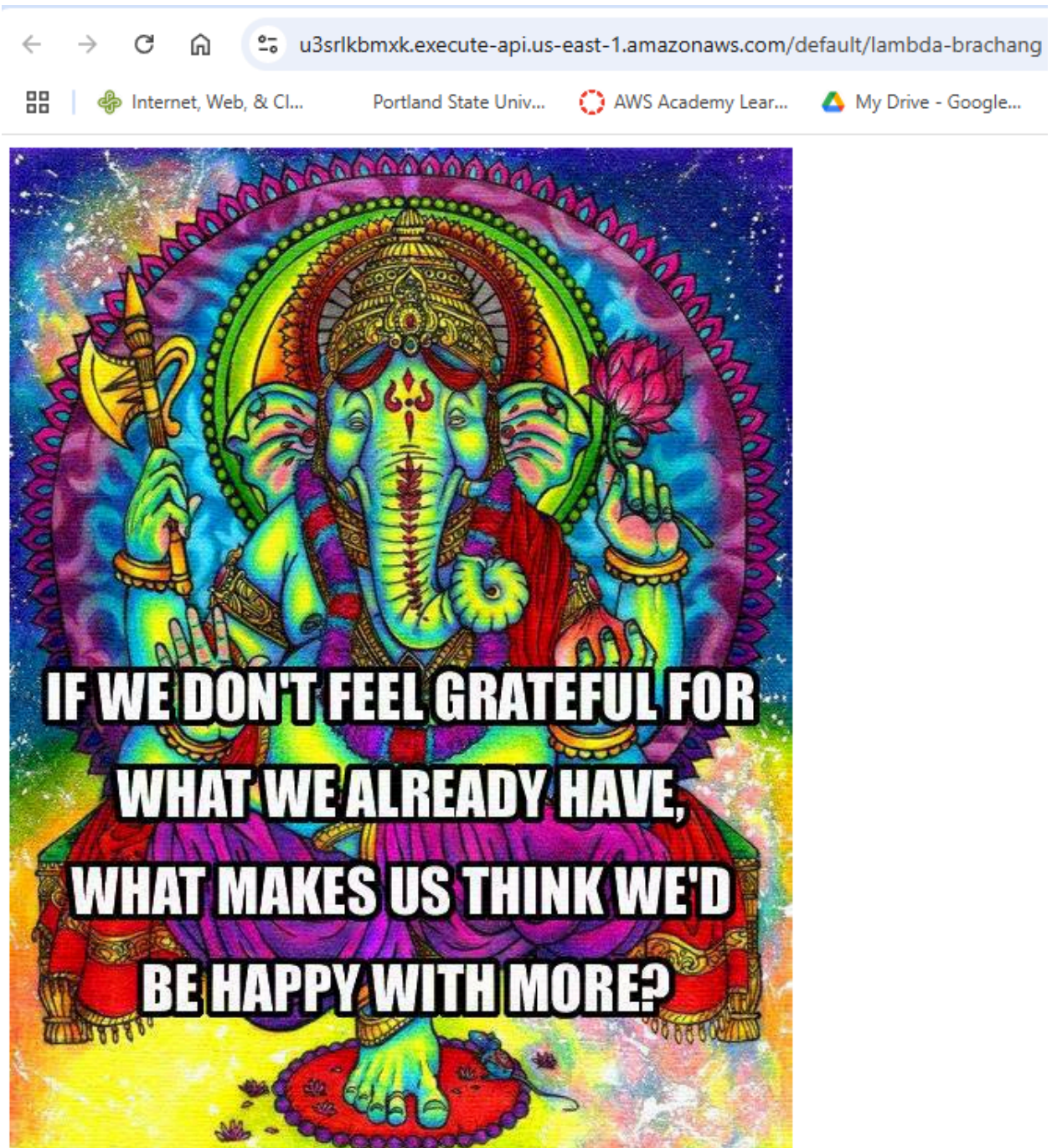
## 8.1a: API Gateway

### 4. Test code

Take a screenshot of the resulting page including the URL bar.



Click "Reload" in the browser and take another screenshot showing the image has changed:



## 8. Test code

Use curl on your Linux VM to access the API endpoint and show the results. Take a screenshot for your lab notebook.

```
OpenSSH SSH client
brachang@ada:~$ brachang@ada:~$ curl https://j529p22p03.execute-api.us-east-1.amazonaws.com/default/gettime-brachang
{"currentTime": "2025-02-27 00:51:12.053577"}brachang@ada:~$ S
```

## 8.2a: Lambda, API Gateway Guestbook

## 10. Deploy API to production and view entries

```
/Documents/cs430p/cs430-src/06_aws_restapi_lambda/frontend/src/index.html
```

### Guestbook

Name:

Email:

Message:

Hello DynamoDB!

### Entries

## 12. API endpoint for signing (2)

Take a screenshot showing that the submission worked.

Create resource

/

/entries

GET

OPTIONS

/entry

POST

Request body

```
1 {
2   "name" : "Bradley Chang",
3   "email" : "brachang@pdx.edu",
4   "message" : "Hello API Gateway"
5 }
```

Test

1

/entry - POST method test results

Request

/entry

Latency ms

737

Status

200

Response body

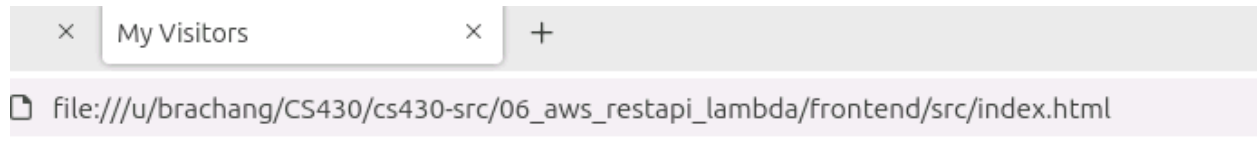
[{"message": "Hello DynamoDB", "date": "2025-02-10 23:31:02.983899", "email": "brachang@pdx.edu", "name": "Bradley Chang"}, {"message": "Hello Docker DynamoDB", "date": "2025-02-10 23:43:24.797965", "email": "brachang@pdx.edu", "name": "Bradley Chang"}, {"message": "Hello Cloud9!", "date": "2025-02-11 00:21:01.362325", "email": "brachang@pdx.edu", "name": "Bradley Chang"}, {"message": "Hello EC2!", "date": "2025-02-11 00:48:27.094863", "email": "brachang@pdx.edu", "name": "Bradley Chang"}, {"message": "Hello Elastic Beanstalk!", "date": "2025-02-15 21:48:21.580718", "email": "brachang@pdx.edu", "name": "Bradley Chang"}, {"message": "Hello ECS!", "date": "2025-02-17 08:29:41.421293", "email": "brachang@pdx.edu", "name": "Bradley Chang"}, {"message": "Hello API Gateway", "date": "2025-02-27 02:24:33.979564", "email": "brachang@pdx.edu", "name": "Bradley Chang"}]

Response headers

{
 "Access-Control-Allow-Origin": "\*\*",
 "X-Amzn-Trace-Id": "Root=1-67bfccel-948cd320f530407d980c8f74;Parent=43fdcbf875981048;Sampled=0;Lineage=1:1c33203c:0"
}

Logs

Execution log for request 7c002515-8d5c-44e1-9d80-5da43b6f8bde
Thu Feb 27 02:24:33 UTC 2025 : Starting execution for request: 7c002515-8d5c-44e1-9d80-5da43b6f8bde
Thu Feb 27 02:24:33 UTC 2025 : HTTP Method: POST, Resource Path: /entry
Thu Feb 27 02:24:33 UTC 2025 : Method success with status: 200



## Guestbook

Name:

Email:

Message:

## Entries

### 8.2g: Cloud Functions, API Gateway Guestbook

### 10. Create the API Gateway

Include the hostname for the API gateway in your lab notebook.

```
brachang@cloudshell:~/cs430-src/06_gcp_restapi_cloudfunctions (cloud-chang-brachang)$ gcloud api-gateway gateways describe gbapigw --location=us-central1
apiConfig: projects/660545260092/locations/global/apis/gbapi/configs/gbapiconfig
createTime: '2025-03-03T20:09:41.051422486Z'
defaultHostname: gbapigw-8fg7lcn0.uc.gateway.dev
displayName: gbapigw
name: projects/cloud-chang-brachang/locations/us-central1/gateways/gbapigw
state: ACTIVE
updateTime: '2025-03-03T20:11:33.208375515Z'
brachang@cloudshell:~/cs430-src/06_gcp_restapi_cloudfunctions (cloud-chang-brachang)$
```

### 11. Test the API via Python Requests (GET)

Take a screenshot of its output



```
>>> print(resp.json())
[{'name': 'Bradley Chang', 'email': 'brachang@pdx.edu', 'date': '2025-02-25 00:35:09.303205+00:00', 'message': 'Hello Kubernetes!'}, {'name': 'Bradley Chang', 'email': 'brachang@pdx.edu', 'date': '2025-02-16 20:25:26.934303+00:00', 'message': 'Hello App Engine!'}, {'name': 'Bradley Chang', 'email': 'brachang@pdx.edu', 'date': '2025-02-11 05:14:46.423535+00:00', 'message': 'Hello Datastore!'}, {'name': 'Bradley Chang', 'email': 'brachang@pdx.edu', 'date': '2025-02-25 01:02:29.159730+00:00', 'message': 'Hello Cloud Build!'}, {'name': 'Bradley Chang', 'email': 'brachang@pdx.edu', 'date': '2025-02-17 20:03:05.763443+00:00', 'message': 'Hello Cloud Run!'}, {'name': 'Bradley Chang', 'email': 'brachang@pdx.edu', 'date': '2025-02-11 05:41:12.016337+00:00', 'message': 'Hello Cloud Shell!'}, {'name': 'Bradley Chang', 'email': 'brachang@pdx.edu', 'date': '2025-02-11 05:24:52.206731+00:00', 'message': 'Hello Docker Datastore!'}, {'name': 'Bradley Chang', 'email': 'brachang@pdx.edu', 'date': '2025-02-11 06:01:22.416517+00:00', 'message': 'Hello Compute Engine!'}]
>>> first_entry = guest_data[0]
>>> for key, value in first_entry.items():
...     print(f"{key}: {value}")
...     File "<stdin>", line 2
...         print(f"{key}: {value}")
...         ^
IndentationError: expected an indented block after 'for' statement on line 1
>>> for key, value in first_entry.items():
...     print(f"{key}: {value}")
...
... name: Bradley Chang
... email: brachang@pdx.edu
... date: 2025-02-25 00:35:09.303205+00:00
... message: Hello Kubernetes!
>>>
```

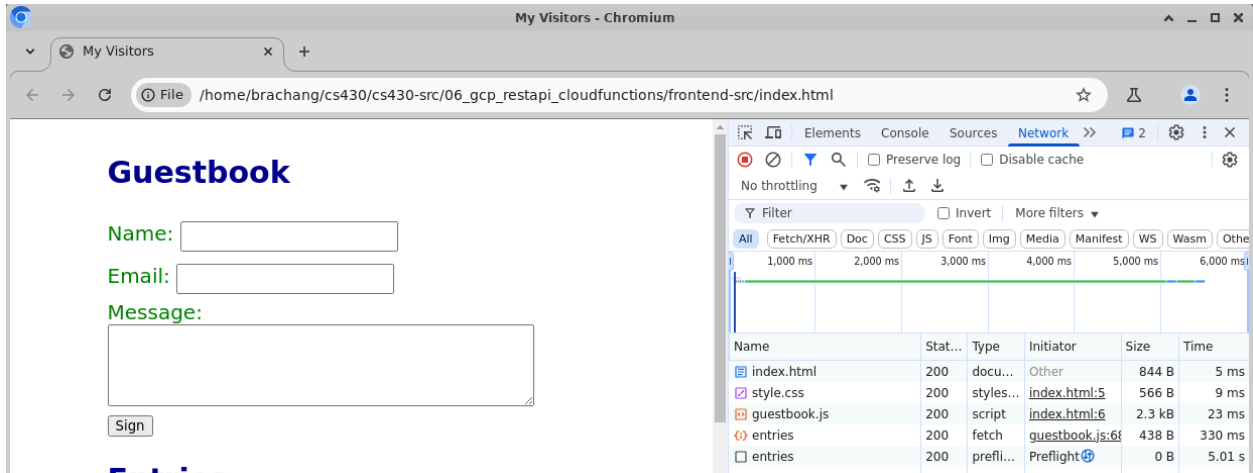
## 12. Test the API via Python Requests (POST)

Take a screenshot of the output for your lab notebook

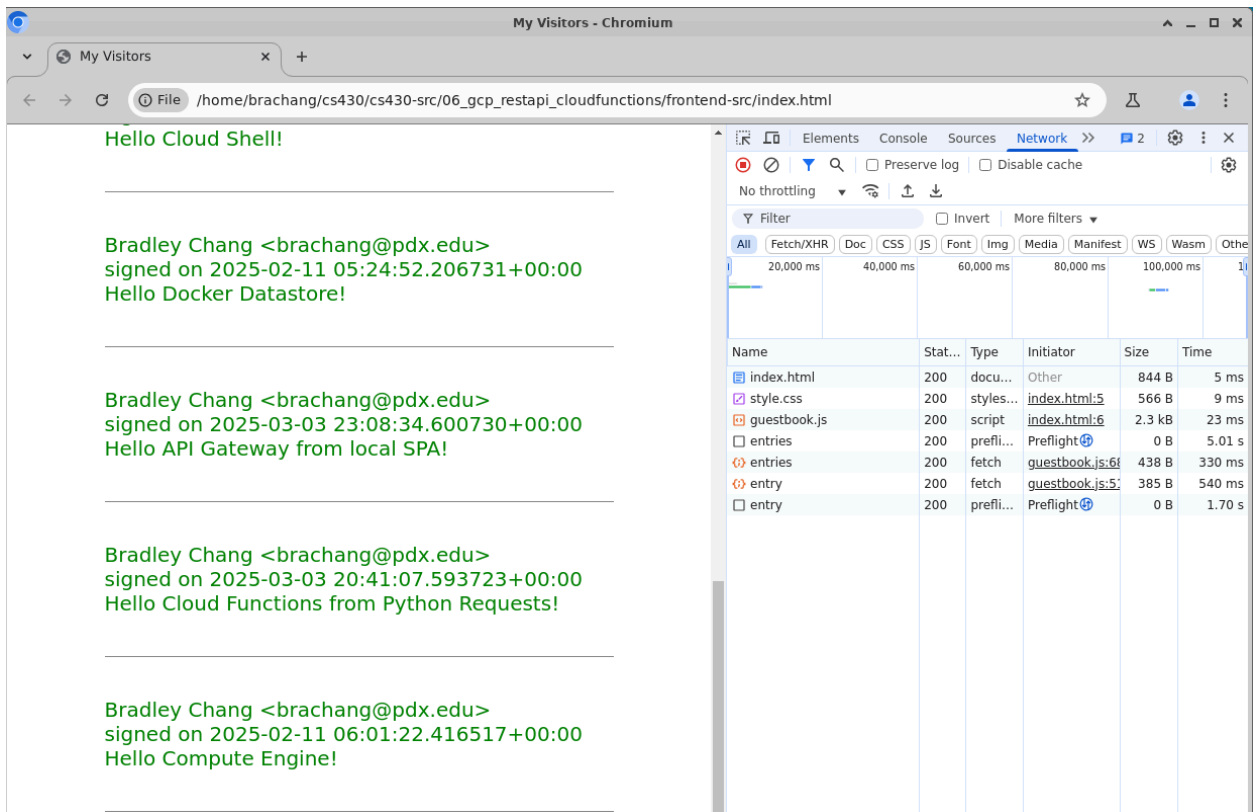
```
>>> import json
>>> mydict = {
...     'name': 'Bradley Chang',
...     'email': 'brachang@pdx.edu',
...     'message': 'Hello Cloud Functions from Python Requests!'
... }
>>> resp = requests.post('https://us-central1-cloud-chang-brachang.cloudfunctions.net/entry', json=mydict)
>>> print(resp.status_code)
200
>>> print(resp.headers)
{'content-type': 'application/json', 'access-control-allow-origin': '*', 'X-Cloud-Trace-Context': '3b351806464cb4efe753b33a8126040b;o=1', 'Date': 'Mon, 03 Mar 2025 20:41:08 GMT', 'Server': 'Google Frontend', 'Content-Length': '1224'}
>>> print(resp.text)
[{"name": "Bradley Chang", "email": "brachang@pdx.edu", "date": "2025-02-25 00:35:09.303205+00:00", "message": "Hello Kubernetes!"}, {"name": "Bradley Chang", "email": "brachang@pdx.edu", "date": "2025-02-16 20:25:26.934303+00:00", "message": "Hello App Engine!"}, {"name": "Bradley Chang", "email": "brachang@pdx.edu", "date": "2025-02-11 05:14:46.423535+00:00", "message": "Hello Datastore!"}, {"name": "Bradley Chang", "email": "brachang@pdx.edu", "date": "2025-02-25 01:02:29.159730+00:00", "message": "Hello Cloud Build!"}, {"name": "Bradley Chang", "email": "brachang@pdx.edu", "date": "2025-02-17 20:03:05.763443+00:00", "message": "Hello Cloud Run!"}, {"name": "Bradley Chang", "email": "brachang@pdx.edu", "date": "2025-02-11 05:41:12.016337+00:00", "message": "Hello Cloud Shell!"}, {"name": "Bradley Chang", "email": "brachang@pdx.edu", "date": "2025-02-11 05:24:52.206731+00:00", "message": "Hello Docker Datastore!"}, {"name": "Bradley Chang", "email": "brachang@pdx.edu", "date": "2025-02-11 06:01:22.416517+00:00", "message": "Hello Compute Engine!"}]
```

## 15. Version #1: Local file system

Take a screenshot showing the preflight request to the API that allows API access, as well as the subsequent fetch request have been successful. Note that this request may not appear in Firefox browsers.



Take a screenshot of the Guestbook including the URL.

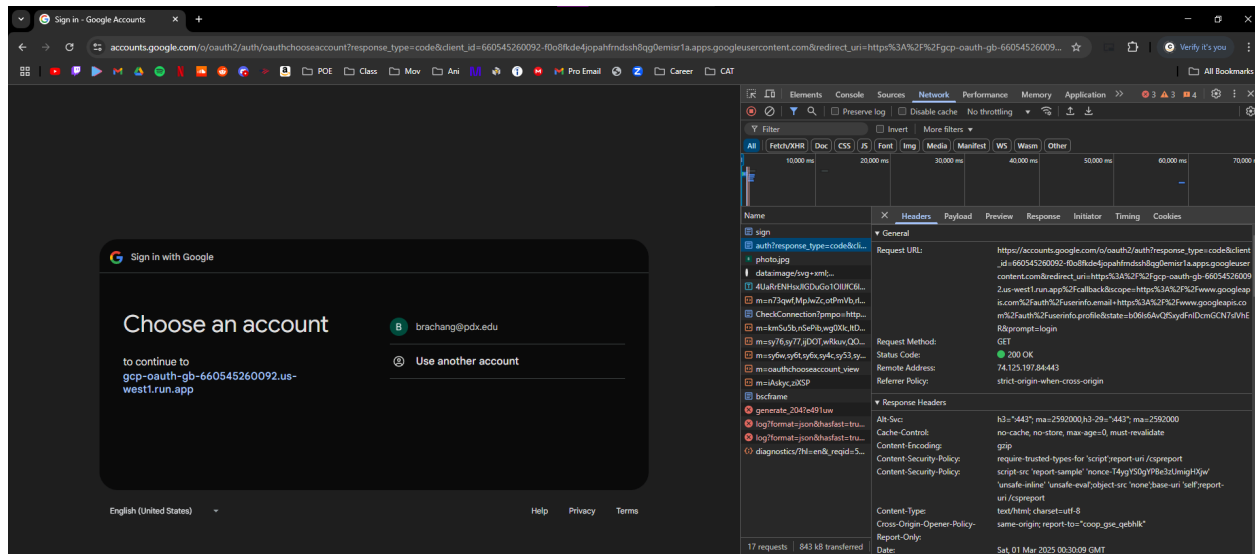


## 16. Version #2: Google Cloud Storage bucket

Take a screenshot of the Guestbook including the URL.





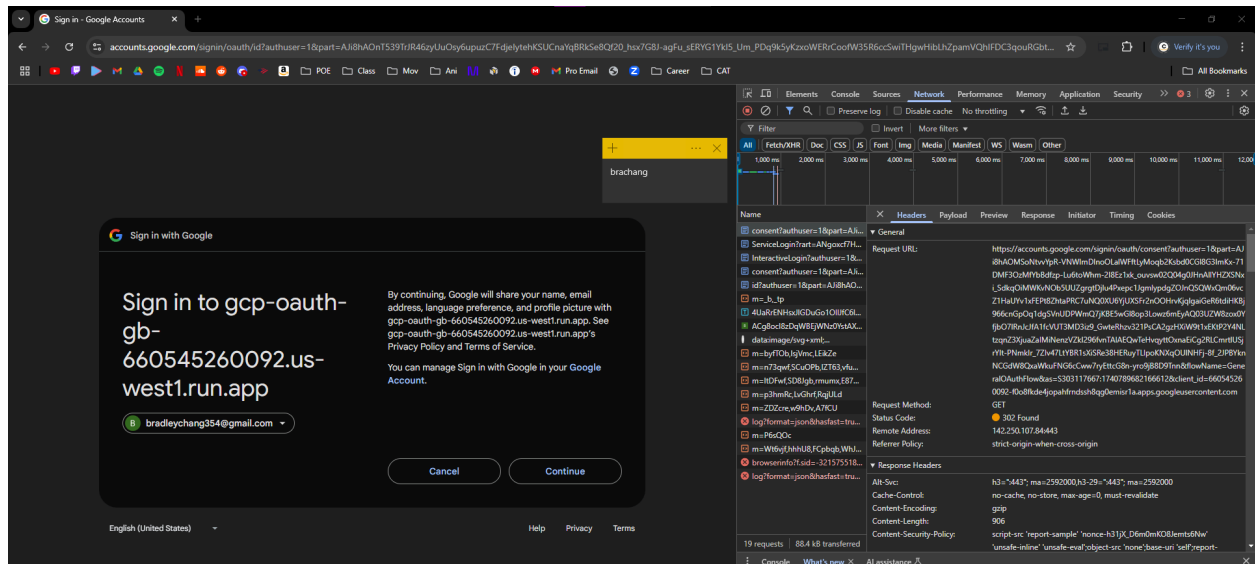


Based on the description of the source code, what lines of code in our application are responsible for the second request shown?

This line in sign.py:

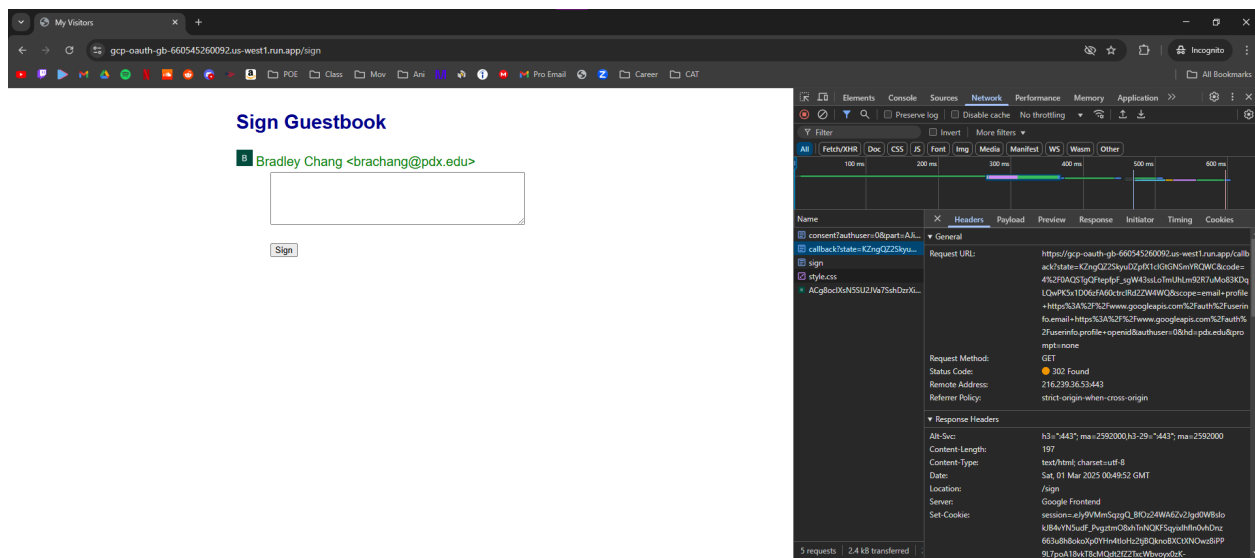
```
authorization_url, state = google.authorization_url(authorization_base_url, prompt='login')
session['oauth_state'] = state
return redirect(authorization_url)
```

Take a screenshot of the permissions you (as a user) are granting the Guestbook access to.

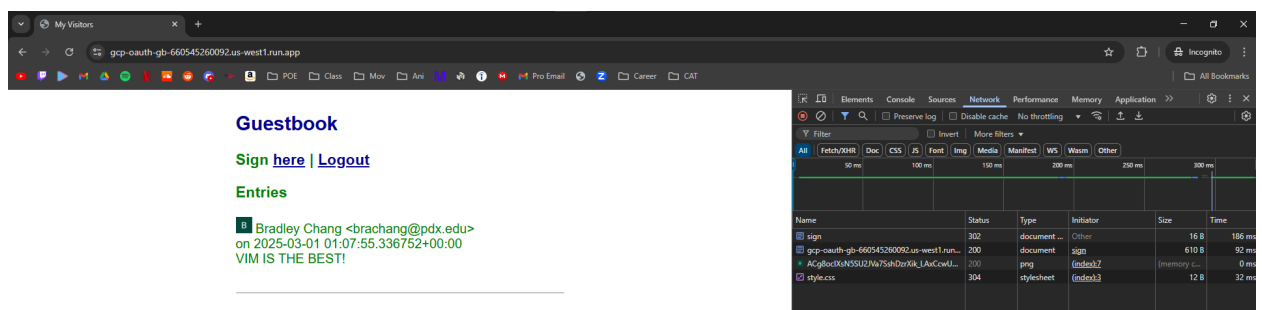
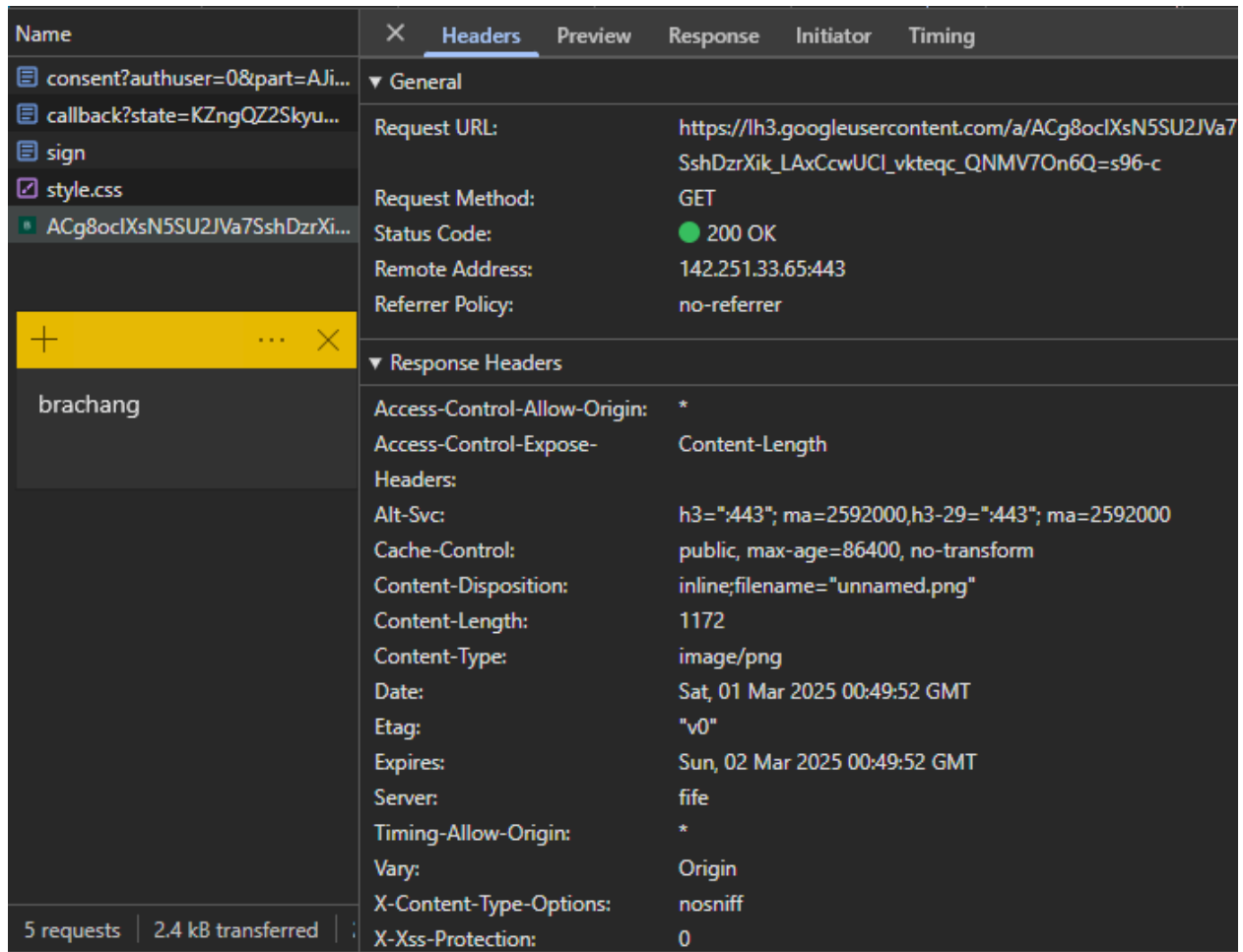


Using a different email since I accidentally already signed in and consented on my pdx one.

Take a screenshot of the Headers that includes the entire Callback URL and its returned HTTP status code. What is the URI for the Location that the User sent to by the Callback?

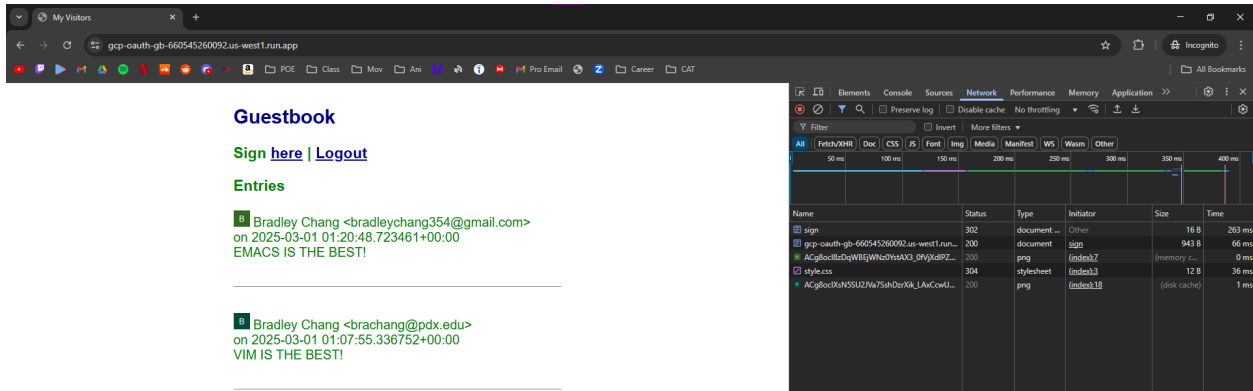


Find the request within Developer Tools that fetches the embedded image and take a screenshot of its URL.



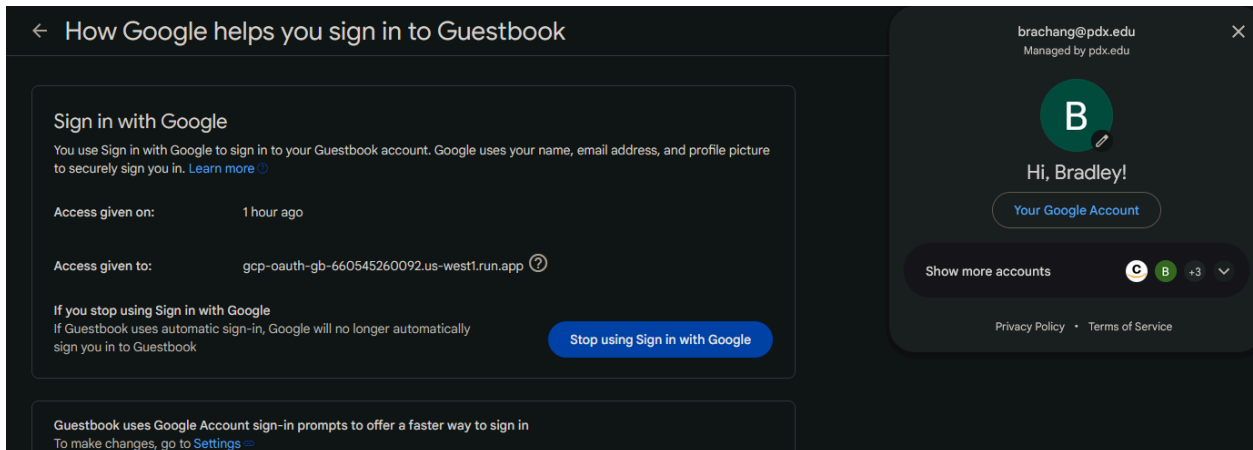
## 15. Secrets manager deployment

Take a screenshot showing multiple authenticated accounts have been able to sign the Guestbook.



## 16. Removing access


Take a screenshot of the expanded information that includes your `OdinId` for your lab notebook.



## 8.4g: Firebase

### 4. Authentication setup

What other domains are given access to this Firebase project by default?

Authorized domains		
For security, to use Phone, Google or third-party authentication, domains need to be authorized for OAuth redirects. <a href="#">Learn more</a>		
		Add domain
Authorized domain	Type	
localhost	Default	
fir-brachang-32c0a.firebaseio.com	Default	
fir-brachang-32c0a.web.app	Default	
cloudshell.dev	Custom	

There's localhost, firebaseapp.com, and web.app

## 8. Bundling with Webpack

Take a screenshot of the first 10 lines of the produced file.

```

JS main.js x
public > scripts > JS main.js > ...
1  /*
2   * ATTENTION: An "eval-source-map" devtool has been used.
3   * This devtool is neither made for production nor for readable output files.
4   * It uses "eval()" calls to create a separate source file with attached SourceMaps in the browser devtools.
5   * If you are trying to read the output file, select a different devtool (https://webpack.js.org/configuration/devtool/)
6   * or disable the default devtool with "devtool: false".
7   * If you are looking for production-ready output files, see mode: "production" (https://webpack.js.org/configuration/mode/).
8   */
9  /******/ (() => { // webpackBootstrap
10 /******/  "use strict";
11 /******/  var webpack_modules = ({

```

What missing functions deal with user authentication?

signIn()

signOutUser()  
initFirebaseAuth()  
getProfilePicUrl()  
getUserName()  
isUserSignedIn()

**What missing functions deal with sending and receiving messages?**

saveMessage(messageText)  
loadMessages()  
saveImageMessage(file)  
saveMessagingDeviceToken()

**What are the names of the elements that are hidden when the user is signed out?**

userNameElement.setAttribute('hidden', 'true');  
userPicElement.setAttribute('hidden', 'true');  
signOutButtonElement.setAttribute('hidden', 'true');  
Hides their user profile and the sign out button.

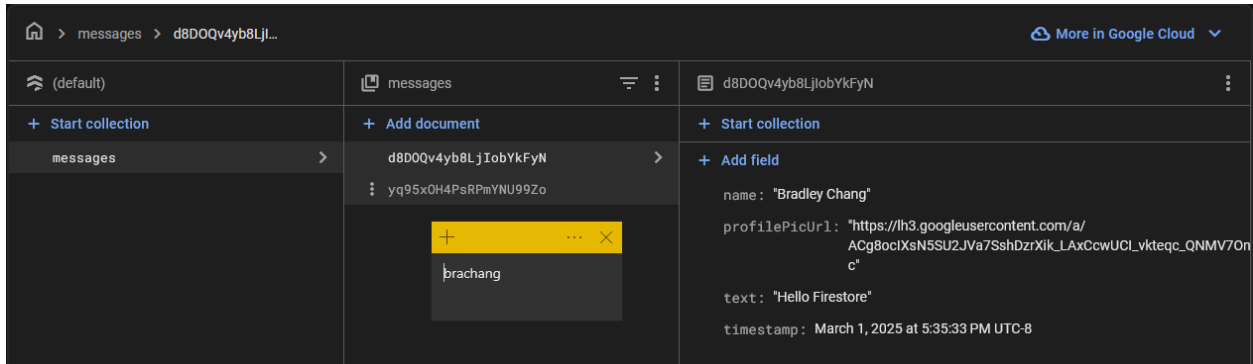
**What is the name of the element that is not hidden when the user is signed out?**

signInButtonElement.removeAttribute('hidden');  
Shows the sign in button.

## 16. Test application with text messaging

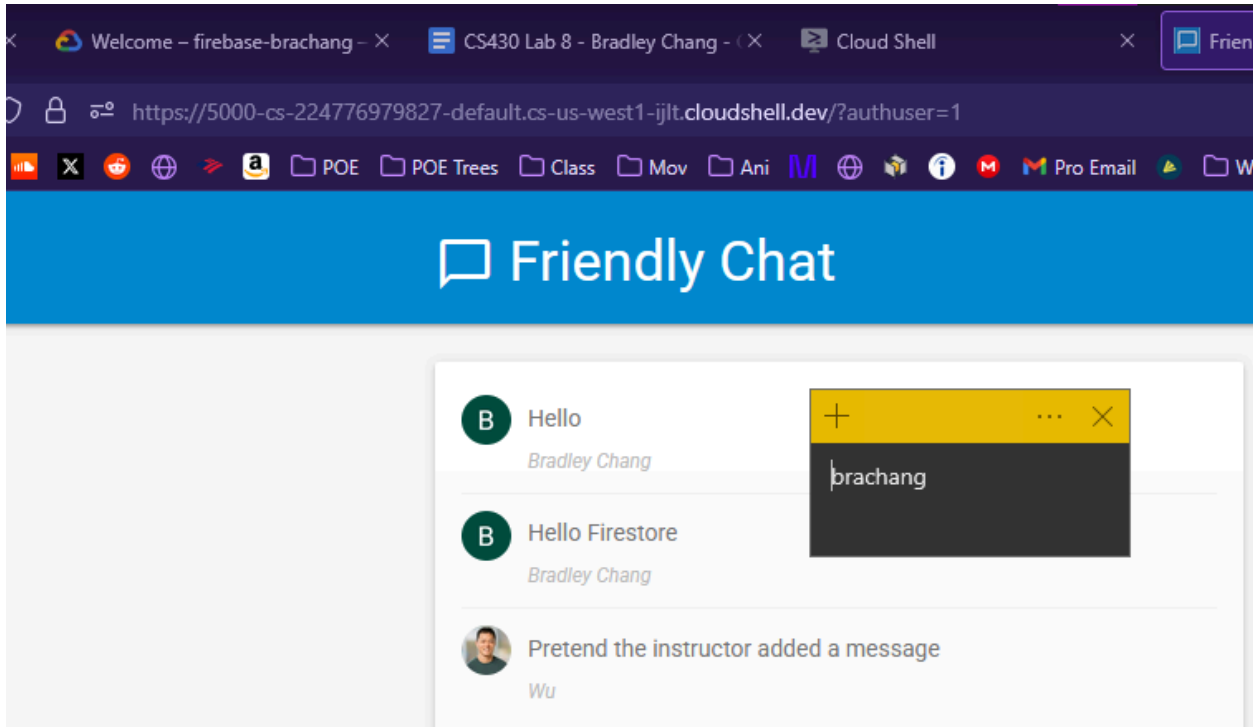
**Include a screenshot of the message and its fields in the database for your lab notebook**





## 17. Manual message insertion

Include a screenshot of the application with its two messages for your lab notebook



## 18. Add image messaging

What is the URL of the image that is first shown in the UI as the message is loading?

<https://www.google.com/images/spin-32.gif?a>

## 19. Test application with image messaging

### How do the fields in an image document differ from that of the text document?

It now contains an imageUrl and storageUri.

### What URL and storage location can the image be found at?

imageUrl:

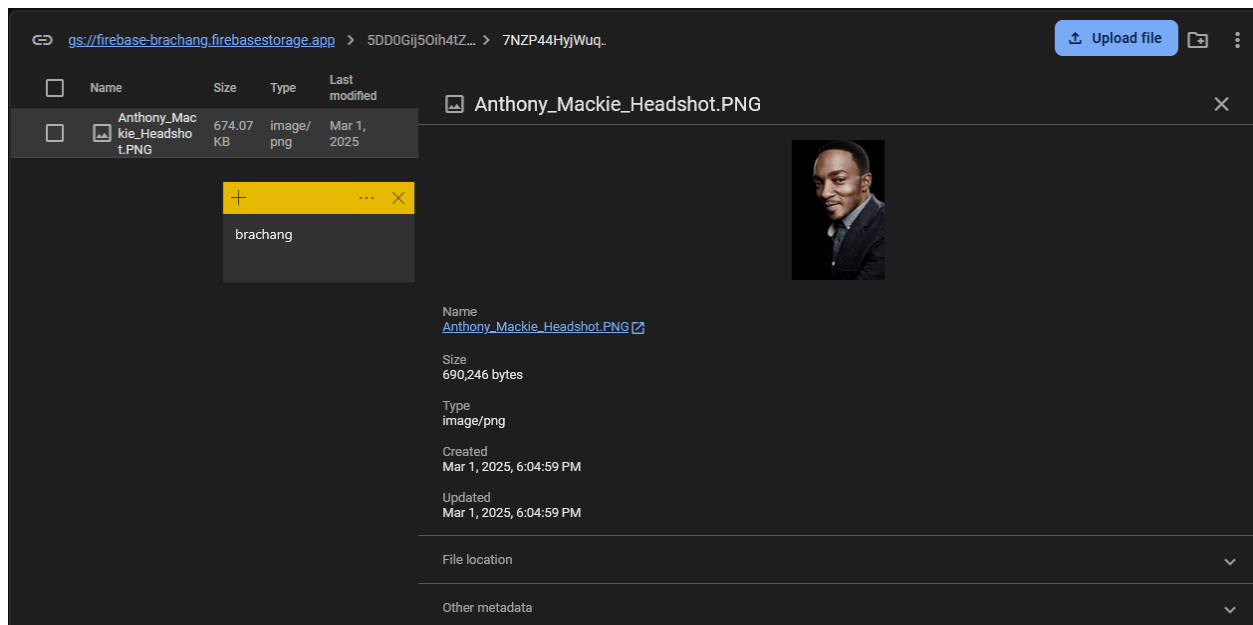
"https://firebasestorage.googleapis.com/v0/b/firebase-brachang.firebaseio.com/o/5DD0Gij5Oih4tZ9z2Xt9fOR0VgO2%2F7NZP44HyjWuqbZCd8eOA%2FAnthony\_Mackie\_Headshot.PNG?alt=media&token=0225d536-d844-4ecc-af0e-5ef7bc81eefd"

(string)

storageUri

"5DD0Gij5Oih4tZ9z2Xt9fOR0VgO2/7NZP44HyjWuqbZCd8eOA/Anthony\_Mackie\_Headshot.PN  
G"

**Take a screenshot of the image in the storage bucket for your lab notebook.**



## 20. Deploy application

What directory is the application going to be served from?

./public

Take a screenshot of the message including the URL for your lab notebook.

```
brachang@cloudshell:~/codelab-friendlychat-web/web-start (firebase-brachang)$ firebase deploy --except functions
== Deploying to 'firebase-brachang'...

i  deploying hosting
i  hosting[fir-brachang-32c0a]: beginning deploy...
i  hosting[fir-brachang-32c0a]: found 8 files in ./public
✓  hosting[fir-brachang-32c0a]: file upload complete
i  hosting[fir-brachang-32c0a]: finalizing version...
✓  hosting[fir-brachang-32c0a]: version finalized
i  hosting[fir-brachang-32c0a]: releasing new version...
✓  hosting[fir-brachang-32c0a]: release complete


✓  Deploy complete!

Project Console: https://console.firebase.google.com/project/firebase-brachang/overview
Hosting URL: https://fir-brachang-32c0a.web.app
brachang@cloudshell:~/codelab-friendlychat-web/web-start (firebase-brachang)$
```

## Friendly Chat

**B** Hello  
*Bradley Chang*

**B** Hello Firestore  
*Bradley Chang*

 Pretend the instructor added a message  
*Wu*

**B** 

*Bradley Chang*

**G** Hello there - Gustavo  
*Gustavo Cotom*