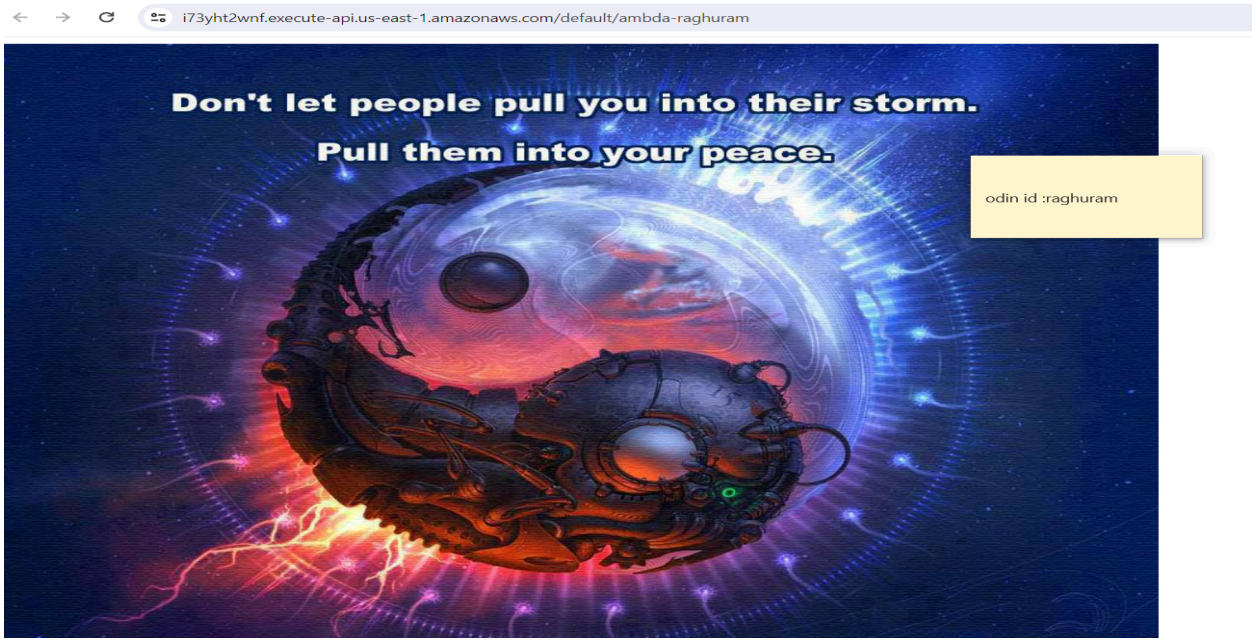
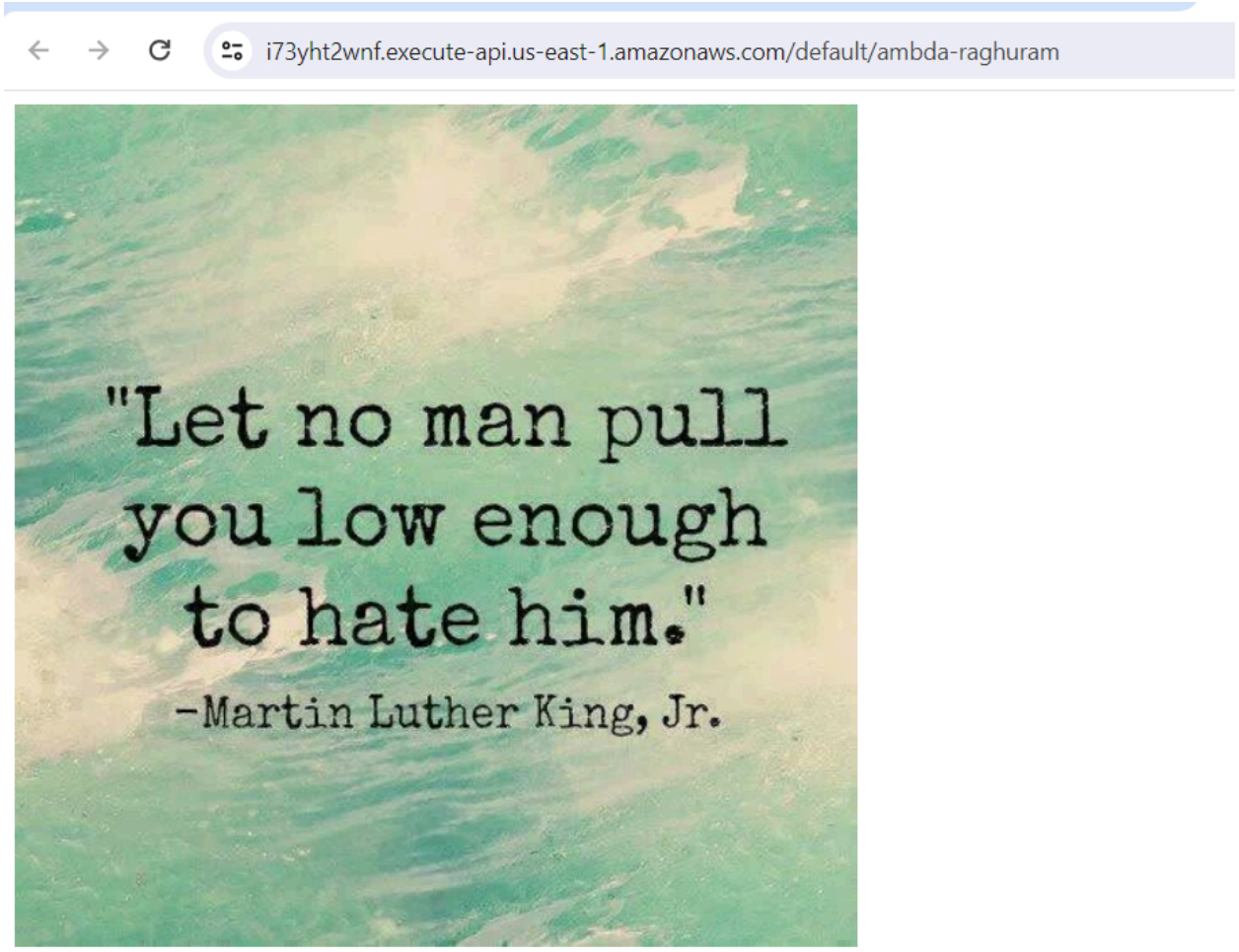
08.1a: API Gateway

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:



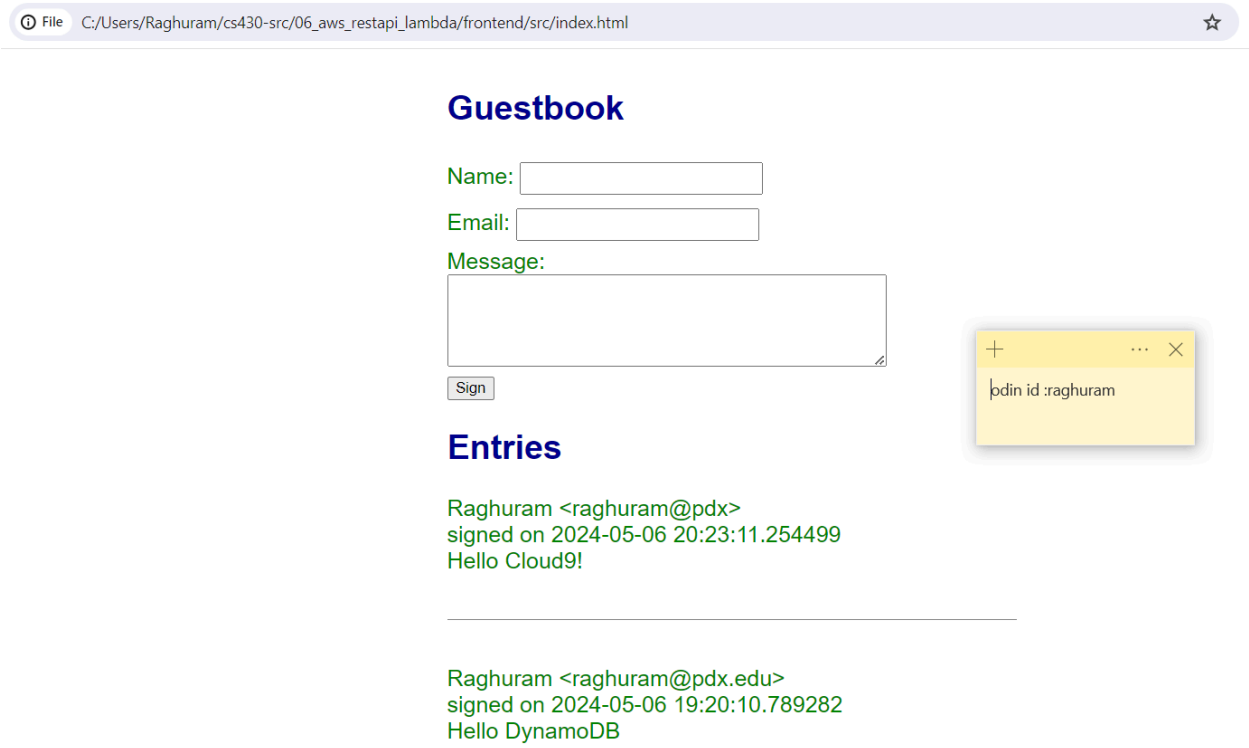
- Use curl on your Linux VM to access the API endpoint and show the results. Take a screenshot for your lab notebook.

```
The list of available updates is more than a week old.
To check for new updates run: sudo apt update

Last login: Sat May 18 07:20:10 2024 from 35.235.240.145
raghuram@course-vm:~$ curl https://v16psy3vh5.execute-api.us-east-1.amazonaws.com/default/gettime-raghuram
{"current_time": "2024-05-25 02:51:12.538293"}raghuram@course-vm:~$
```

08.2a: Lambda, API Gateway Guestbook

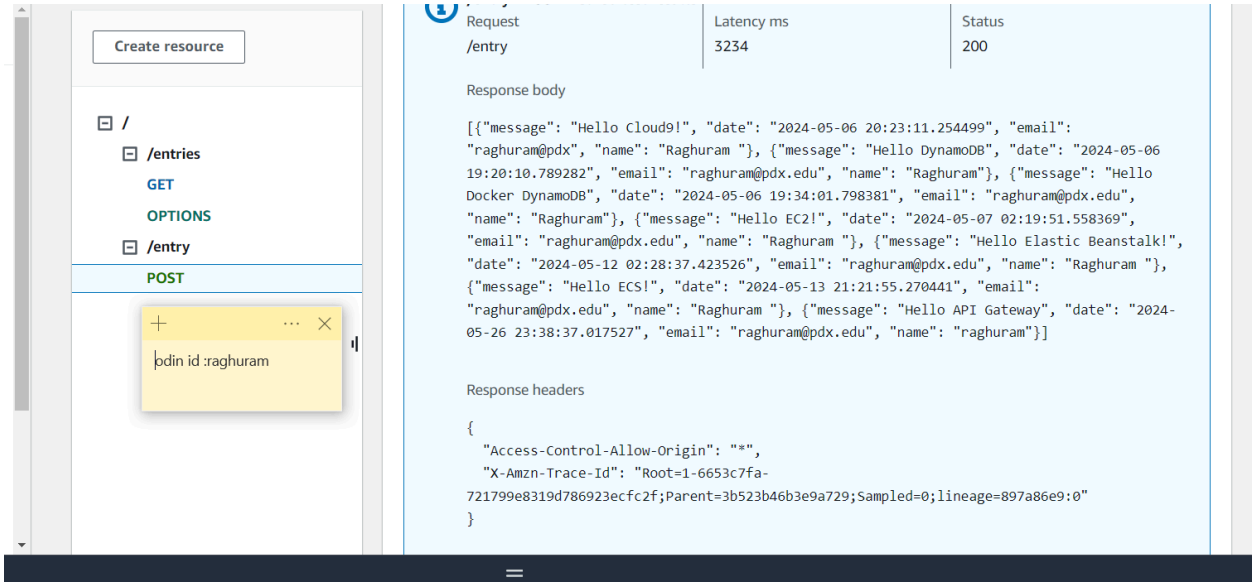
10. Deploy API to production and view entries

- Take a screenshot that shows that you can view the entries in the backend database.



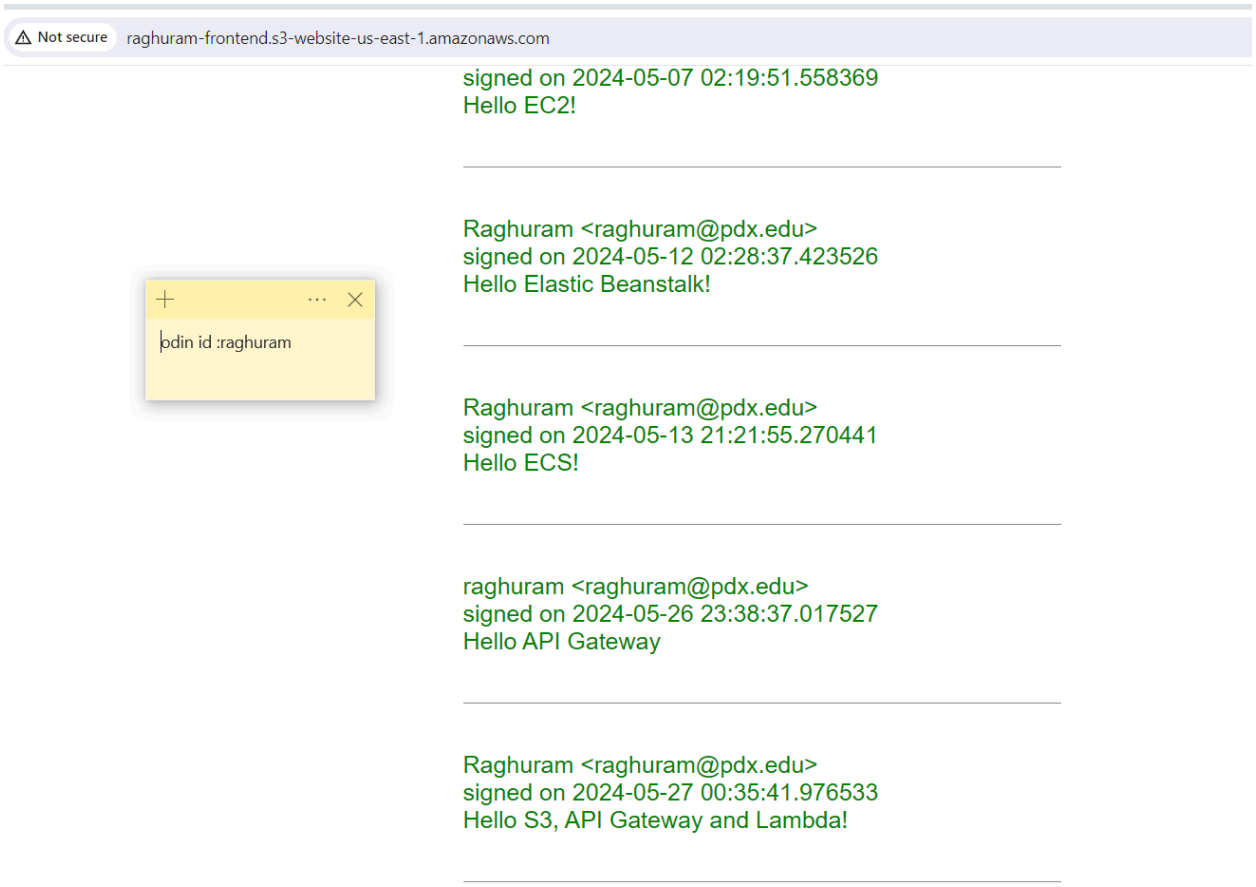
12. API endpoint for signing (2)

- Take a screenshot showing that the submission worked.



16. Configure and Deploy the Frontend

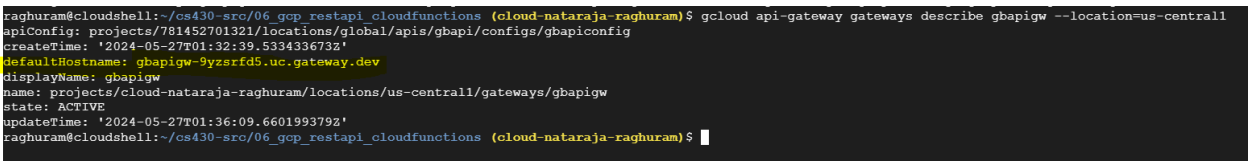
- Take a screenshot as before that shows your entry and the static website hosting URL.



08.2g: Cloud Functions, API Gateway Guestbook

10. Create the API Gateway

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



11. Test the API via Python Requests (GET)

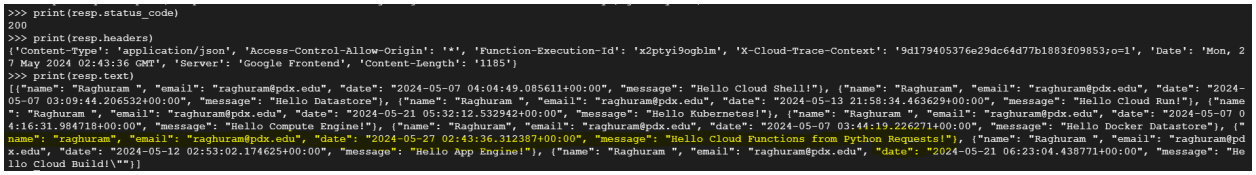
- Take a screenshot of its output

```
raghuram@cloudshell:~/cs430-src/06_gcp_restapi_cloudfunctions (cloud-nataraja-raghuram)$ python3
Python 3.10.12 (main, Nov 20 2023, 15:14:05) [GCC 11.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import requests
>>> resp = requests.get('https://us-central1-cloud-nataraja-raghuram.cloudfunctions.net/entries')
>>> print("Response HTTP status_code:", resp.status_code)
Response HTTP status_code: 200
>>> print(resp.headers)
{'Content-Type': 'application/json', 'Access-Control-Allow-Origin': '*', 'Function-Execution-Id': 'w2fs6ogiyfb4', 'X-Cloud-Trace-Context': '29a6369c8be4e8a6332650fd8b7da888;o=1', 'Date': 'Mon, 2
7 May 2024 01:41:32 GMT', 'Server': 'Google Frontend', 'Content-Length': '1032'}
>>> print(resp.text)
[{"name": "Raghuram ", "email": "raghuram@pdx.edu", "date": "2024-05-07 04:04:49.085611+00:00", "message": "Hello Cloud Shell!"}, {"name": "Raghuram", "email": "raghuram@pdx.edu", "date": "2024-
05-07 03:09:44.206532+00:00", "message": "Hello Datastore"}, {"name": "Raghuram ", "email": "raghuram@pdx.edu", "date": "2024-05-13 21:58:34.463629+00:00", "message": "Hello Cloud Run!"}, {"name
": "Raghuram ", "email": "raghuram@pdx.edu", "date": "2024-05-21 05:32:12.532942+00:00", "message": "Hello Kubernetes!"}, {"name": "Raghuram ", "email": "raghuram@pdx.edu", "date": "2024-05-07 0
4:16:31.984718+00:00", "message": "Hello Compute Engine!"}, {"name": "Raghuram", "email": "raghuram@pdx.edu", "date": "2024-05-07 03:44:19.226271+00:00", "message": "Hello Docker Datastore"}, {"
name": "Raghuram ", "email": "raghuram@pdx.edu", "date": "2024-05-12 02:53:02.174625+00:00", "message": "Hello App Engine!"}, {"name": "Raghuram ", "email": "raghuram@pdx.edu", "date": "2024-05-
21 06:23:04.438771+00:00", "message": "Hello Cloud Build!\""}]
>>> print(resp.json())
[{'name': 'Raghuram ', 'email': 'raghuram@pdx.edu', 'date': '2024-05-07 04:04:49.085611+00:00', 'message': 'Hello Cloud Shell!'}, {'name': 'Raghuram', 'email': 'raghuram@pdx.edu', 'date': '2024-
05-07 03:09:44.206532+00:00', 'message': 'Hello Datastore'}, {'name': 'Raghuram ', 'email': 'raghuram@pdx.edu', 'date': '2024-05-13 21:58:34.463629+00:00', 'message': 'Hello Cloud Run!'}, {'name
': 'Raghuram ', 'email': 'raghuram@pdx.edu', 'date': '2024-05-21 05:32:12.532942+00:00', 'message': 'Hello Kubernetes!'}, {'name': 'Raghuram ', 'email': 'raghuram@pdx.edu', 'date': '2024-05-07 0
4:16:31.984718+00:00', 'message': 'Hello Compute Engine!'}, {'name': 'Raghuram', 'email': 'raghuram@pdx.edu', 'date': '2024-05-07 03:44:19.226271+00:00', 'message': 'Hello Docker Datastore'}, {'
name': 'Raghuram ', 'email': 'raghuram@pdx.edu', 'date': '2024-05-12 02:53:02.174625+00:00', 'message': 'Hello App Engine!'}, {'name': 'Raghuram ', 'email': 'raghuram@pdx.edu', 'date': '2024-05-
21 06:23:04.438771+00:00', 'message': 'Hello Cloud Build!"'}]
```

```
KeyboardInterrupt
>>> for entry in data:
...     print("name:", entry["name"])
...     print("email:", entry["email"])
...     print("date:", entry["date"])
...     print("message:", entry["message"])
...     print("")  # Print an empty line after each entry
...
name: Raghuram
email: raghuram@pdx.edu
date: 2024-05-07 04:04:49.085611+00:00
message: Hello Cloud Shell!

name: Raghuram
email: raghuram@pdx.edu
date: 2024-05-07 03:09:44.206532+00:00
message: Hello Datastore

name: Raghuram
email: raghuram@pdx.edu
date: 2024-05-13 21:58:34.463629+00:00
message: Hello Cloud Run!

name: Raghuram
email: raghuram@pdx.edu
date: 2024-05-21 05:32:12.532942+00:00
message: Hello Kubernetes!

name: Raghuram
email: raghuram@pdx.edu
date: 2024-05-07 04:16:31.984718+00:00
message: Hello Compute Engine!

name: Raghuram
email: raghuram@pdx.edu
date: 2024-05-07 03:44:19.226271+00:00
```

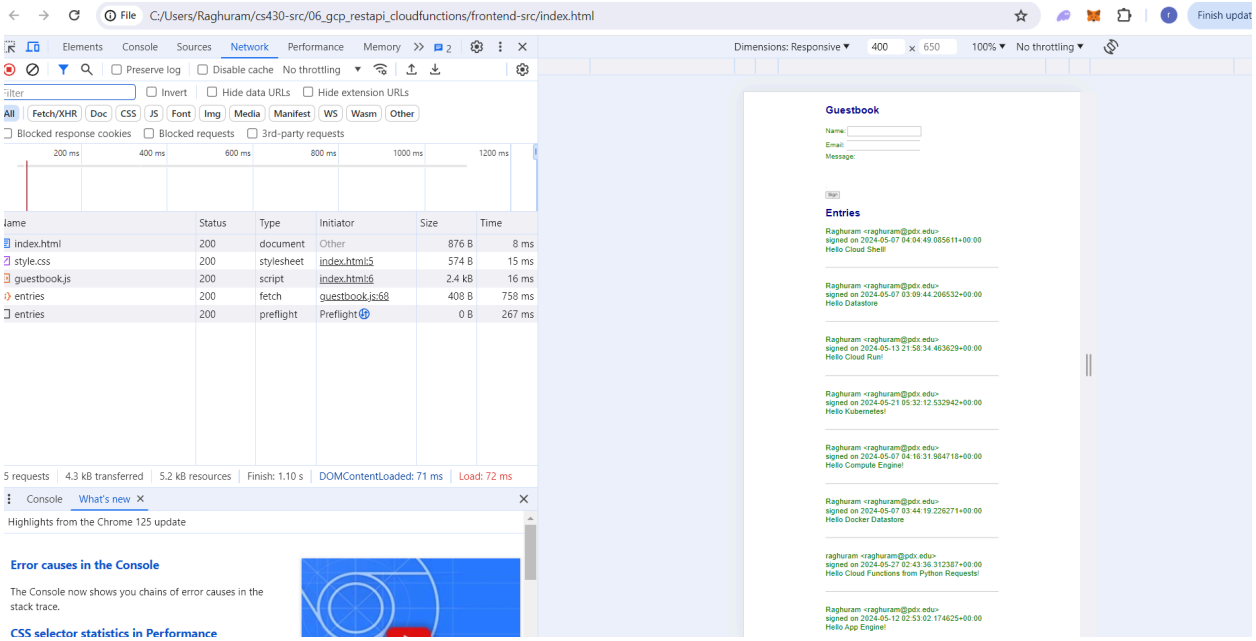
12. Test the API via Python Requests (POST)

- Take a screenshot of the output for your lab notebook

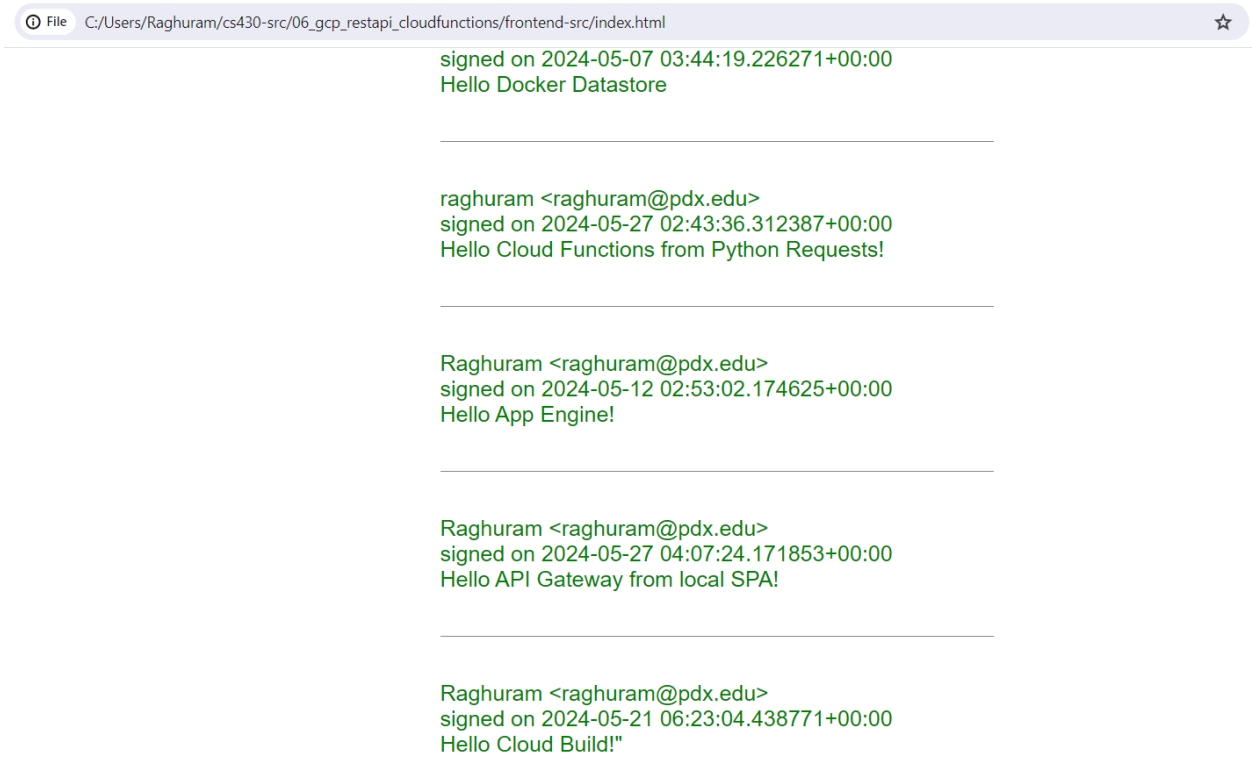


15. Version #1: Local file system

- Take a screenshot showing the preflight request to the API that allows API access, as well as the subsequent fetch request have been successful.

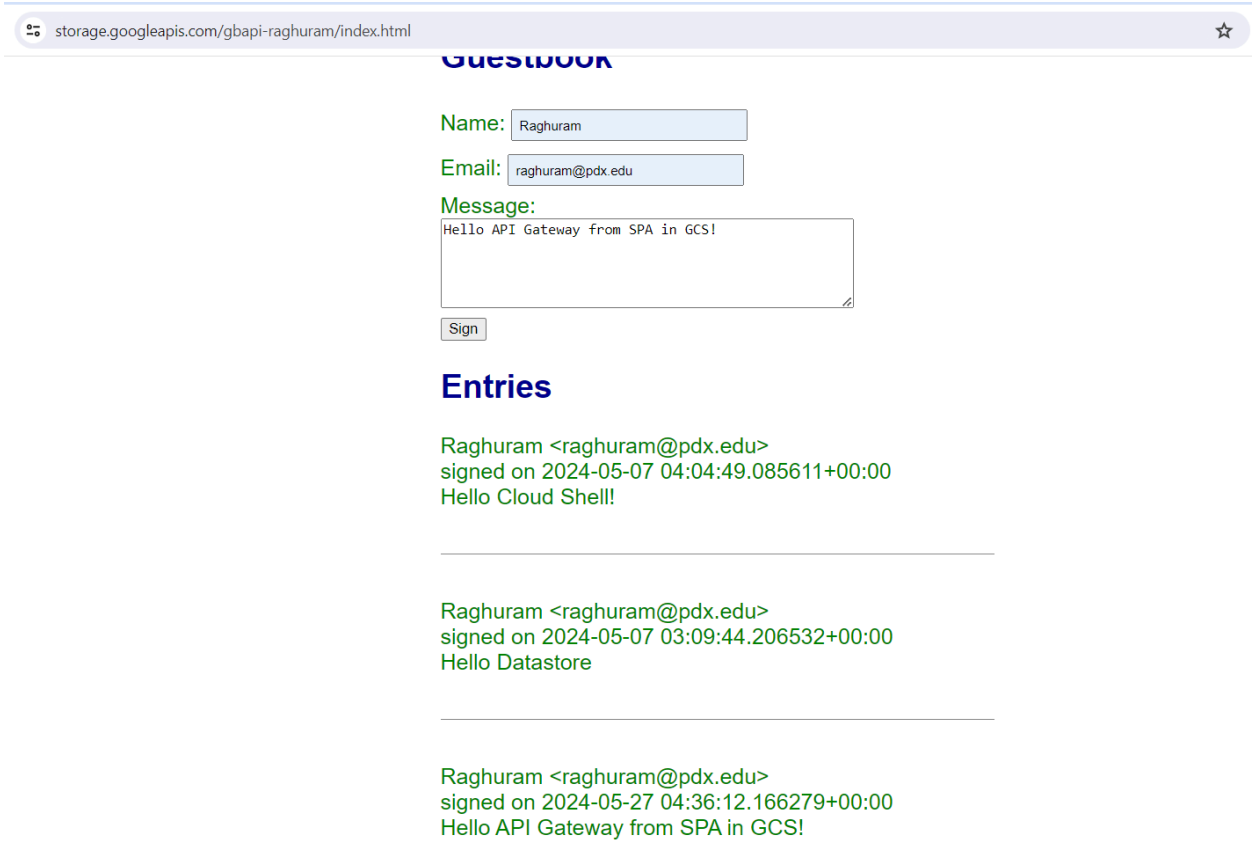


- Take a screenshot of the Guestbook including the URL.



16. Version #2: Google Cloud Storage bucket

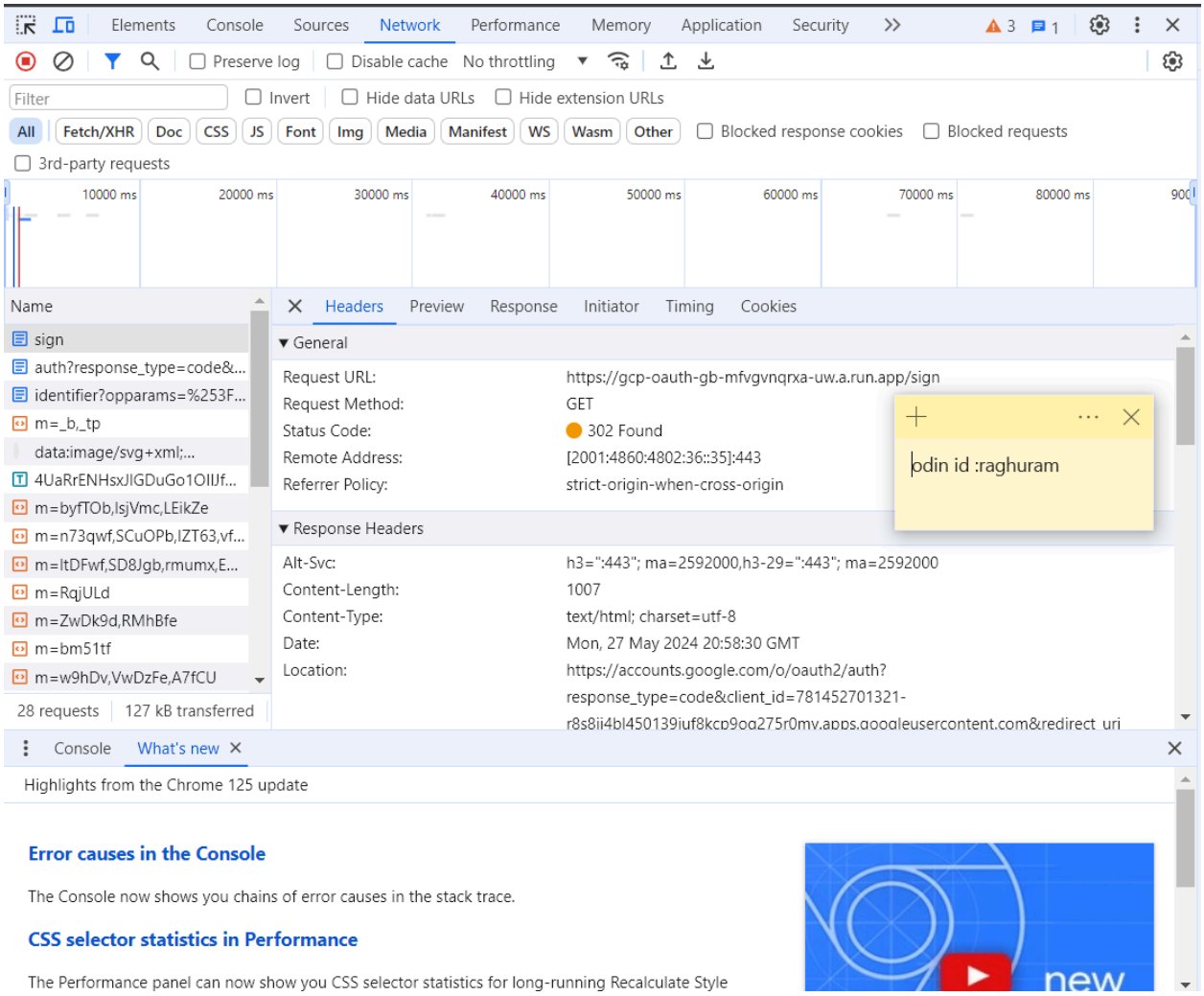
- Take a screenshot of the Guestbook including the URL.

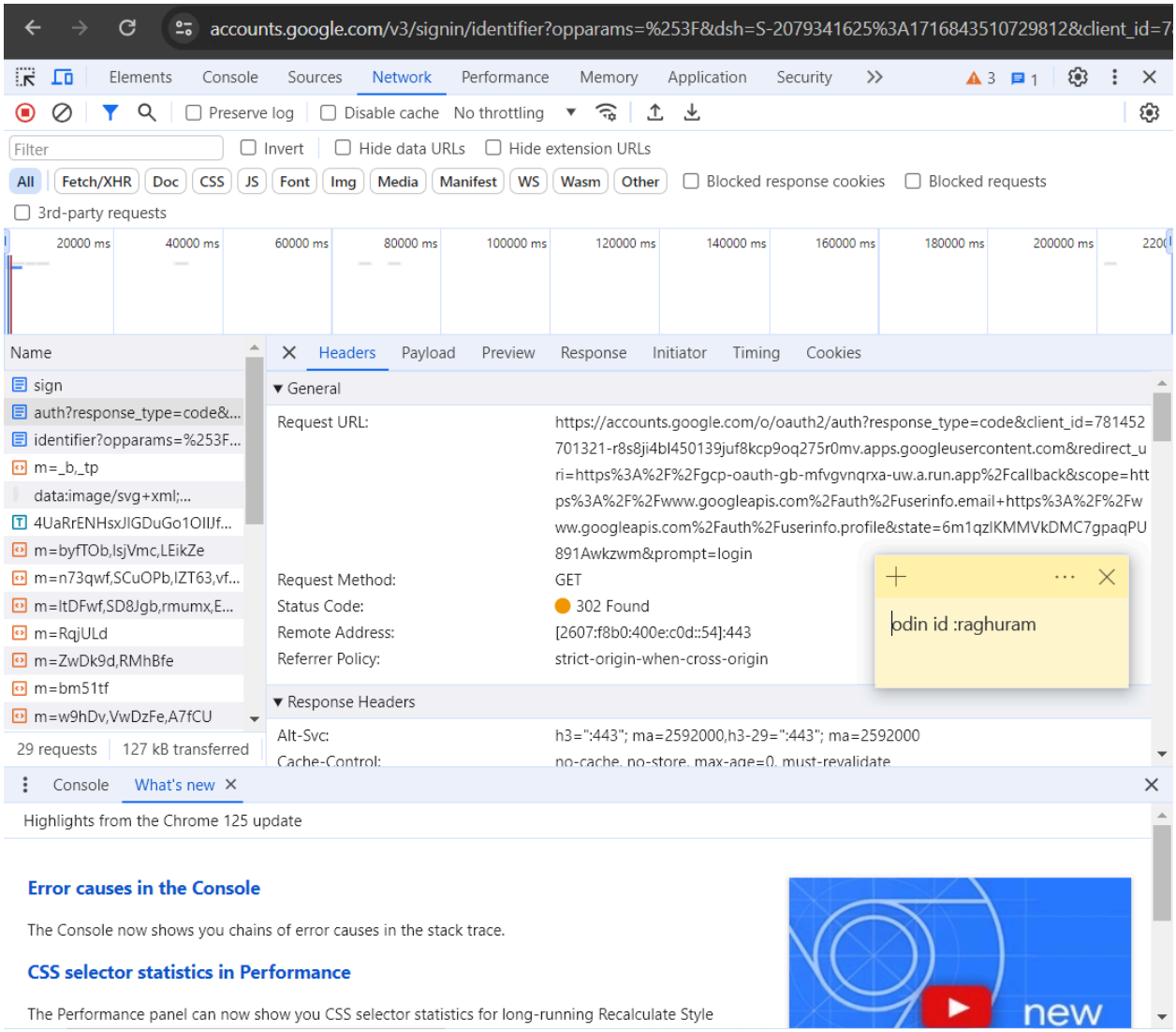


08.3g OAuth2 Guestbook

14. Visit the application

- Take a screenshot of the Headers that includes the URL and the returned HTTP status code for the first two requests for your lab notebook.


Elements
Console
Sources
Network
Performance
Memory
Application
Security
Preserve log
Disable cache
No throttling
Filter
Invert
Hide data URLs
Hide extension URLs
All
Fetch/XHR
Doc
CSS
JS
Font
Img
Media
Manifest
WS
Wasm
Other
Blocked response cookies
Blocked requests
3rd-party requests
10000 ms
20000 ms
30000 ms
40000 ms
50000 ms
60000 ms
70000 ms
80000 ms
Name
sign
auth?response_type=code&...
identifier?opparams=%253F...
m=_b,_tp
data:image/svg+xml;...
4UaRrENHsxJlGDuGo1OIlJf...
m=byfTOb,lsjVmc,LEikZe
m=n73qwf,SCuOPb,lZT63,vf...
m=ltDFwf,SD8Jgb,rmumx,E...
m=RqjULd
m=ZwDk9d,RMhBfe
m=bm51tf
m=w9hDv,VwDzFe,A7fCU
28 requests
127 kB transferred
Headers
Preview
Response
Initiator
Timing
Cookies
General
Request URL:
https://gcp-oauth-gb-mfvgvnqrxa-uw.a.run.app/sign
Request Method:
GET
Status Code:
302 Found
Remote Address:
[2001:4860:4802:36::35]:443
Referrer Policy:
strict-origin-when-cross-origin
Response Headers
Alt-Svc:
h3=":443"; ma=2592000,h3-29=":443"; ma=2592000
Content-Length:
1007
Content-Type:
text/html; charset=utf-8
Date:
Mon, 27 May 2024 20:58:30 GMT
Location:
https://accounts.google.com/o/oauth2/auth?
response_type=code&client_id=781452701321-
odin id :raghuram
Console
What's new
Highlights from the Chrome 125 update
Error causes in the Console
The Console now shows you chains of error causes in the stack trace.
CSS selector statistics in Performance
The Performance panel can now show you CSS selector statistics for long-running Recalculate Style
new



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

  google = OAuth2Session(

  client_id,

  redirect_uri="http://localhost:8000/callback",

  state=session["oauth_state"],

)

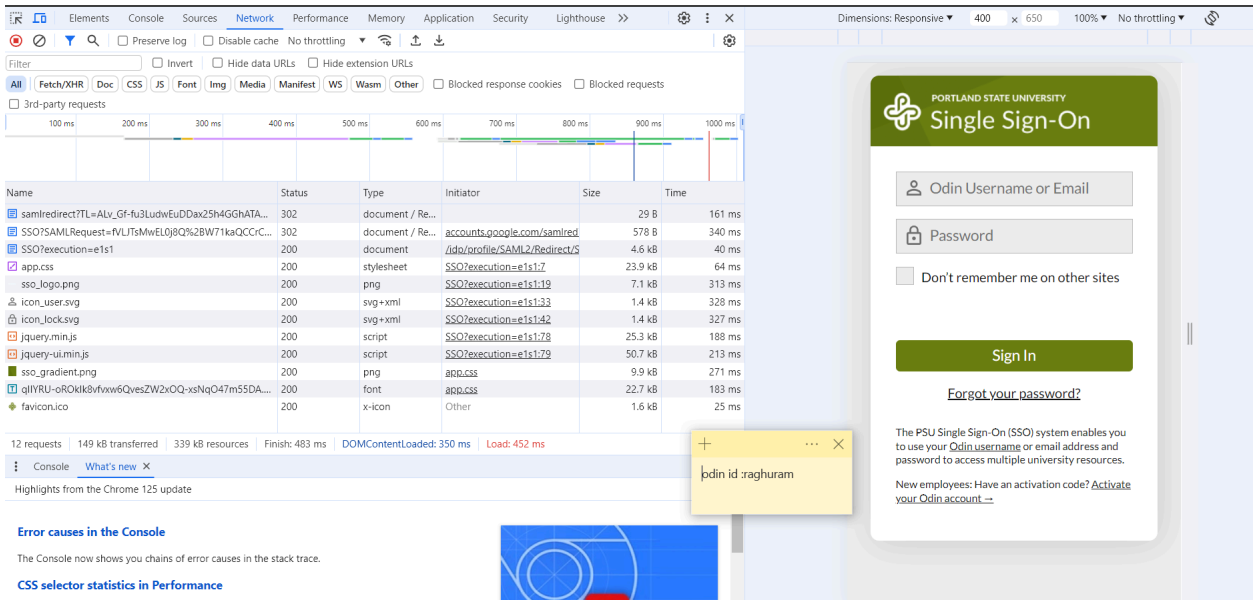
token = google.fetch_token(

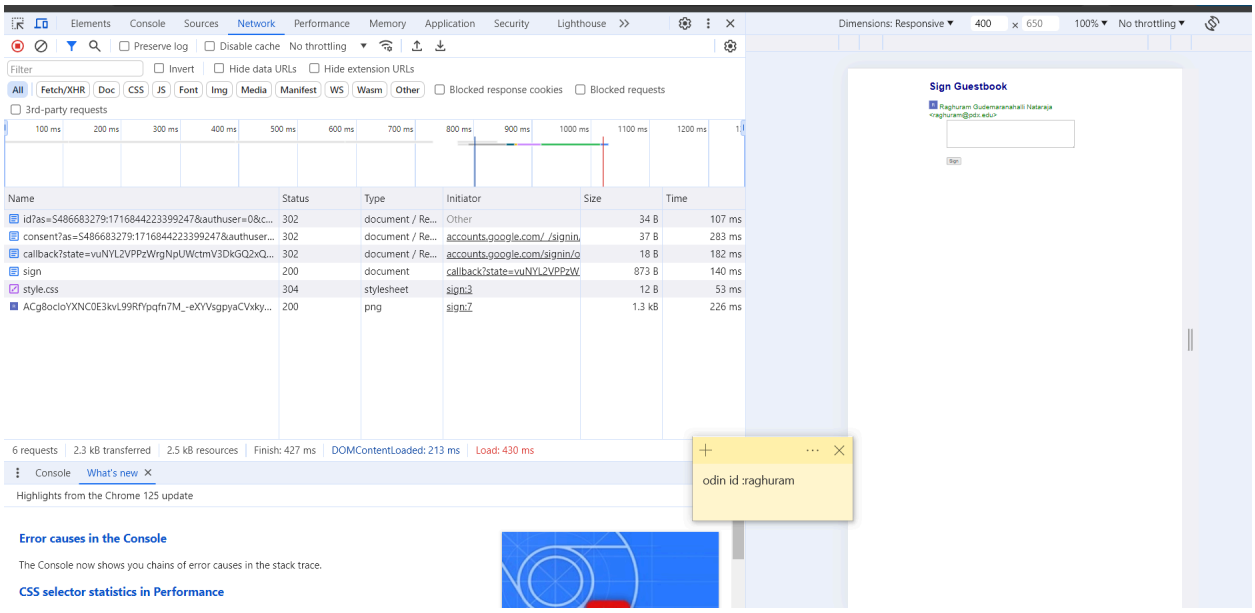
token_url, client_secret=client_secret, authorization_response=request.url

)

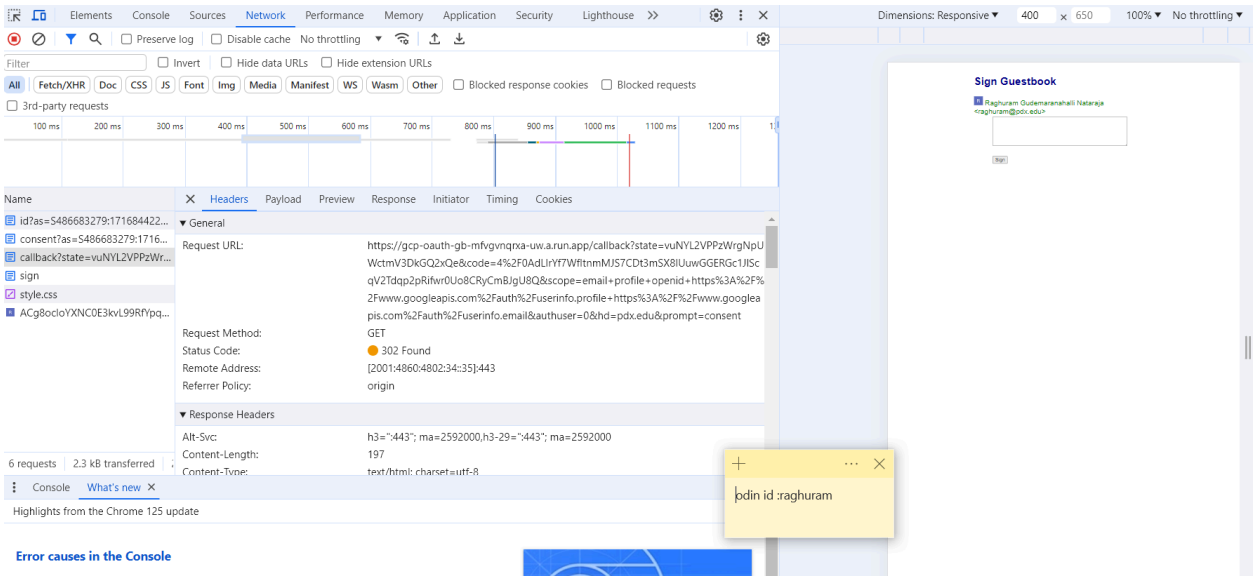
session["oauth_token"] = token

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



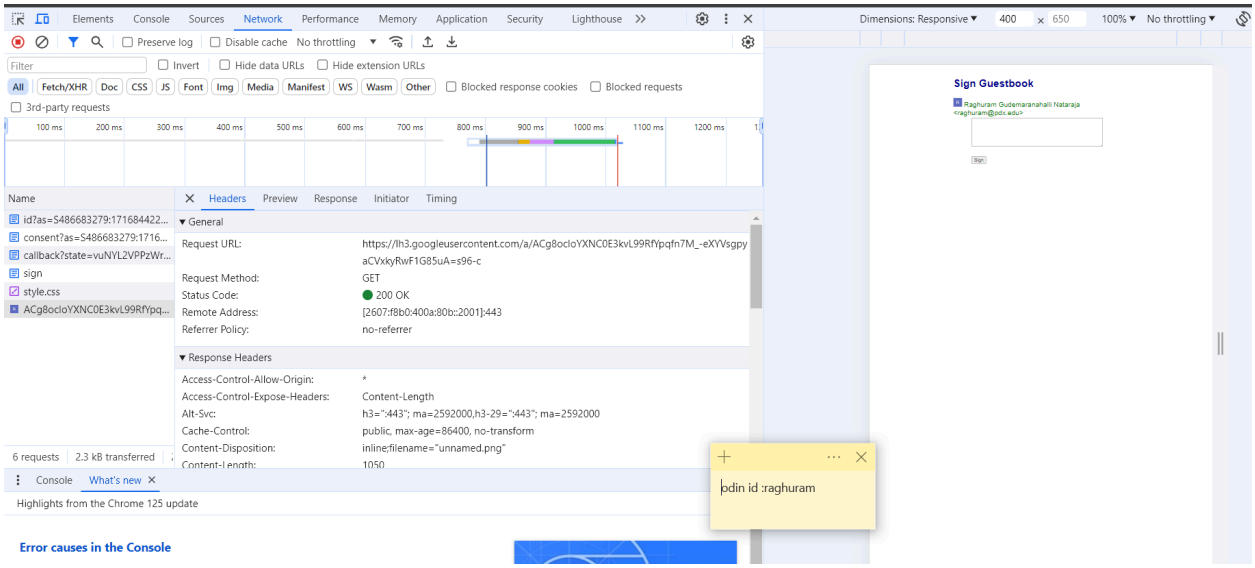


- Take a screenshot of the Headers that includes the entire Callback URL and its returned HTTP status code. What is the URI for the Location that the User sent to by the Callback?



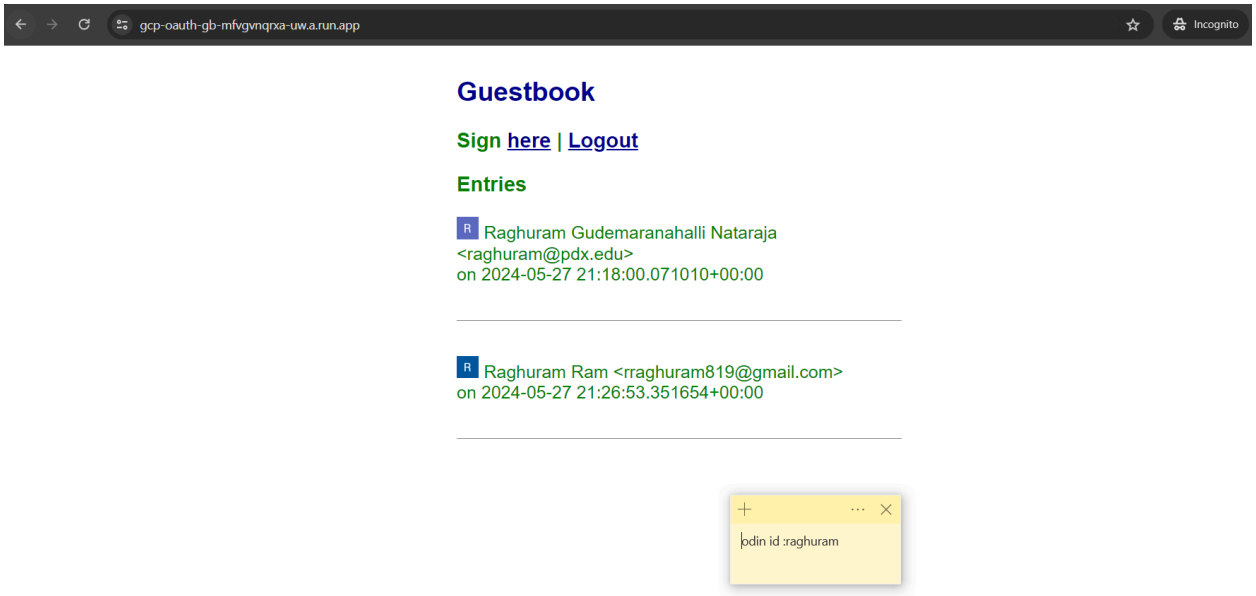
Location : /sign

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



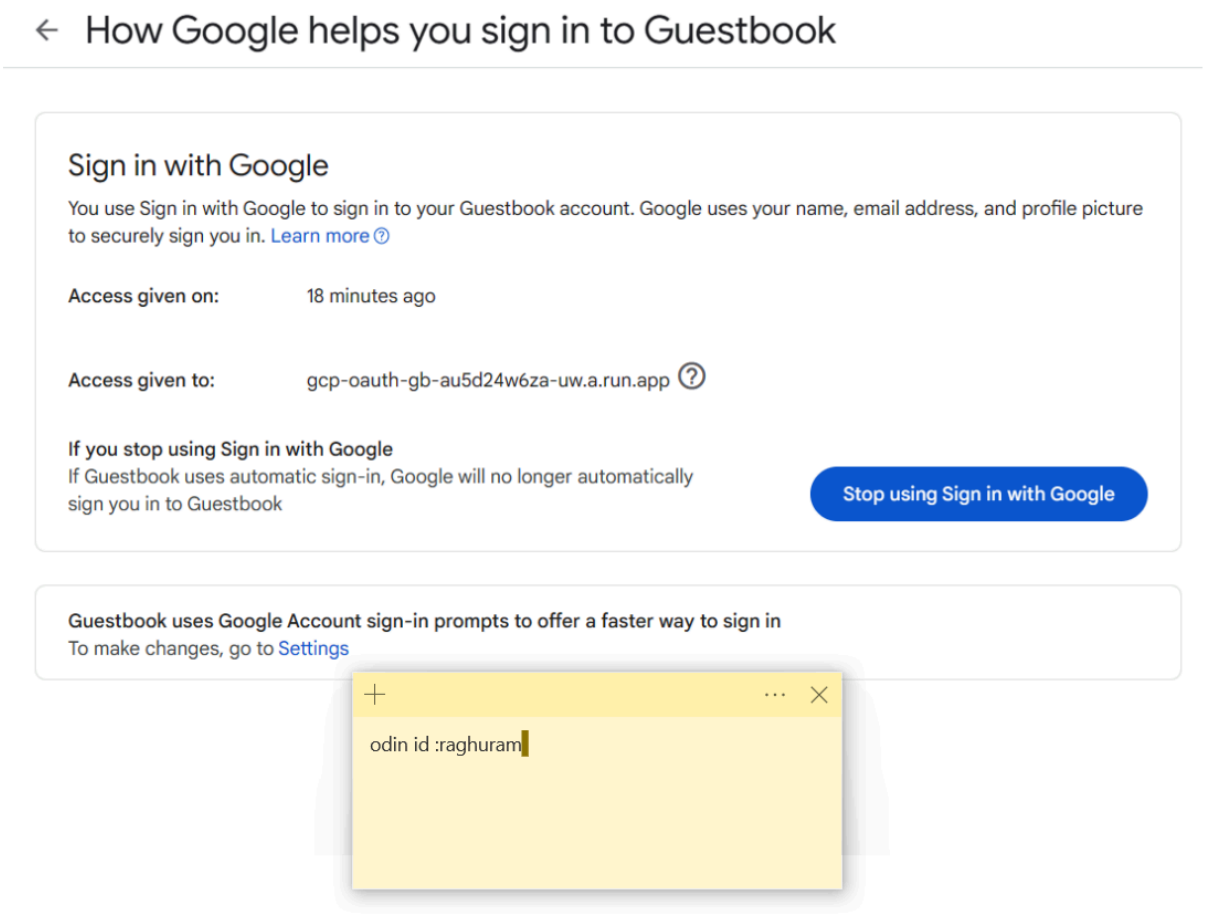
15. Secrets manager deployment

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



16. Removing access

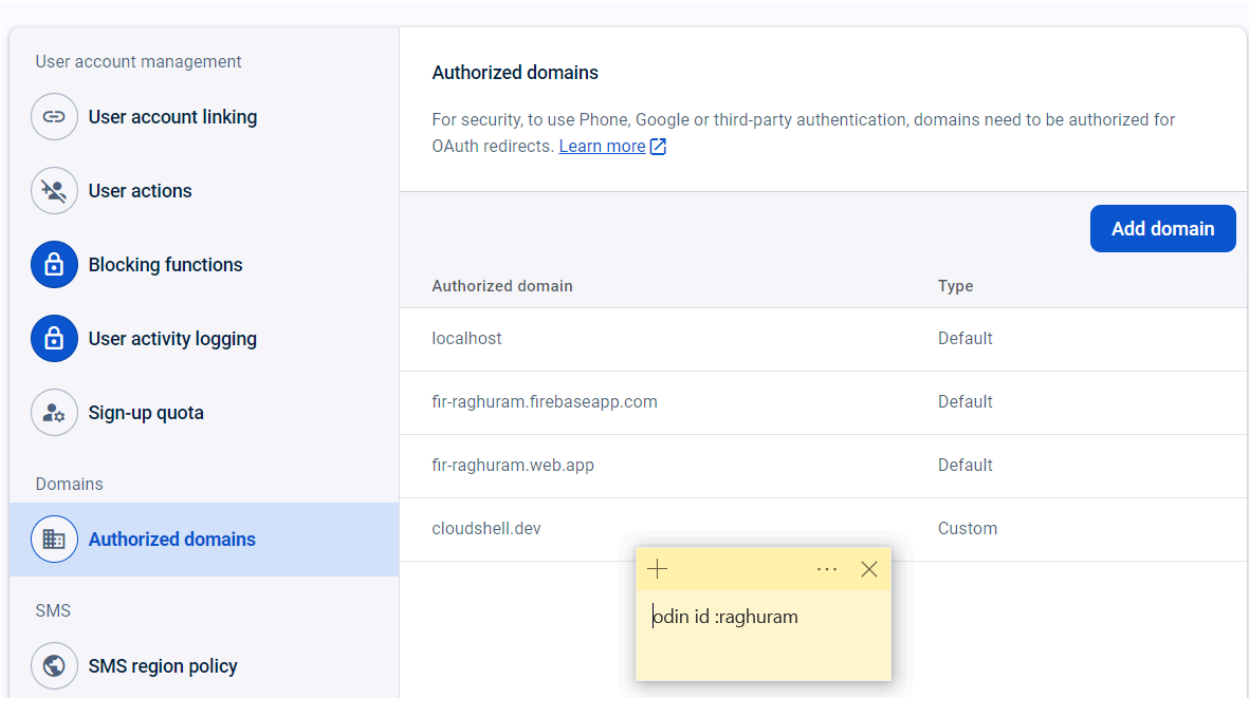
- Take a screenshot of the expanded information that includes your OdinId for your lab notebook.



08.4g: Firebase

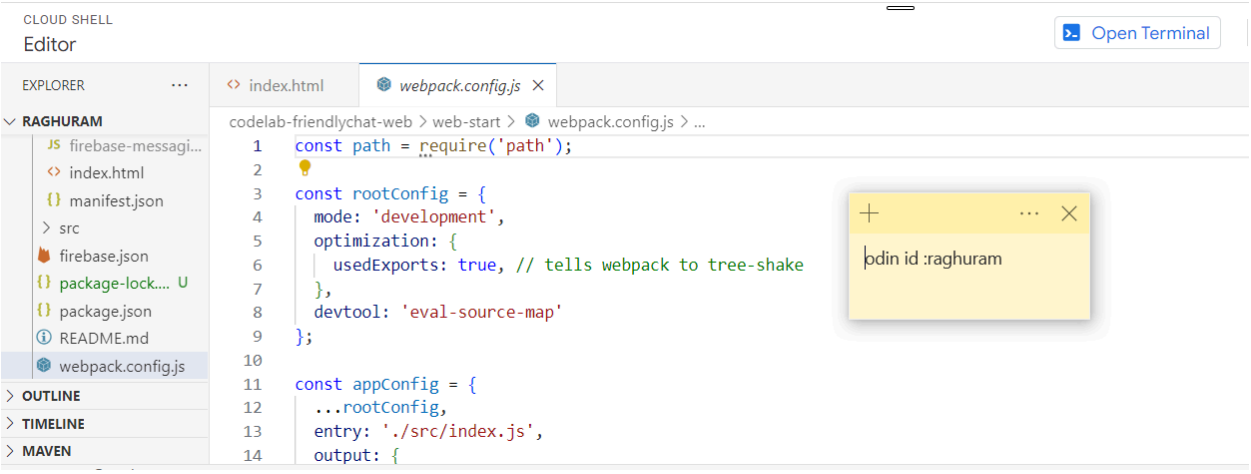
4. Authentication setup

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



8. Bundling with Webpack

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



12. Add authentication

- What missing functions deal with user authentication?

```
signIn()
```

- What missing functions deal with sending and receiving messages?

```
saveMessage(messageText)
```

```
loadMessages()
```

13. Update UI

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

```
userNameElement
```

```
userPicElement
```

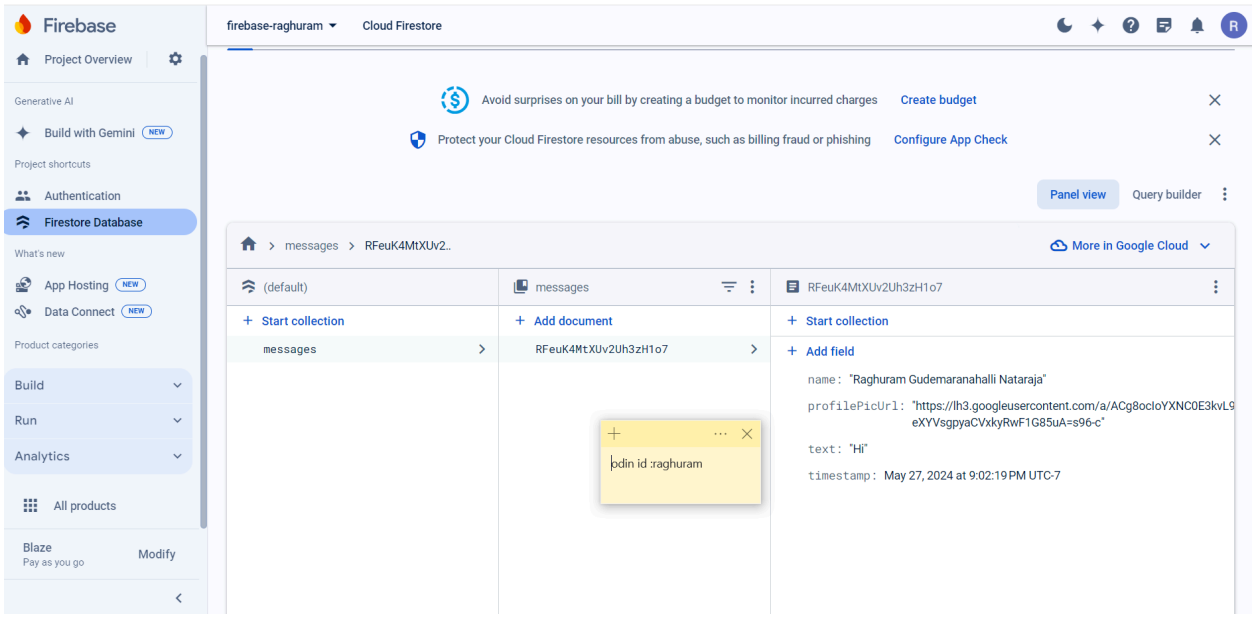
```
signOutButtonElement
```

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

```
signInButtonElement
```

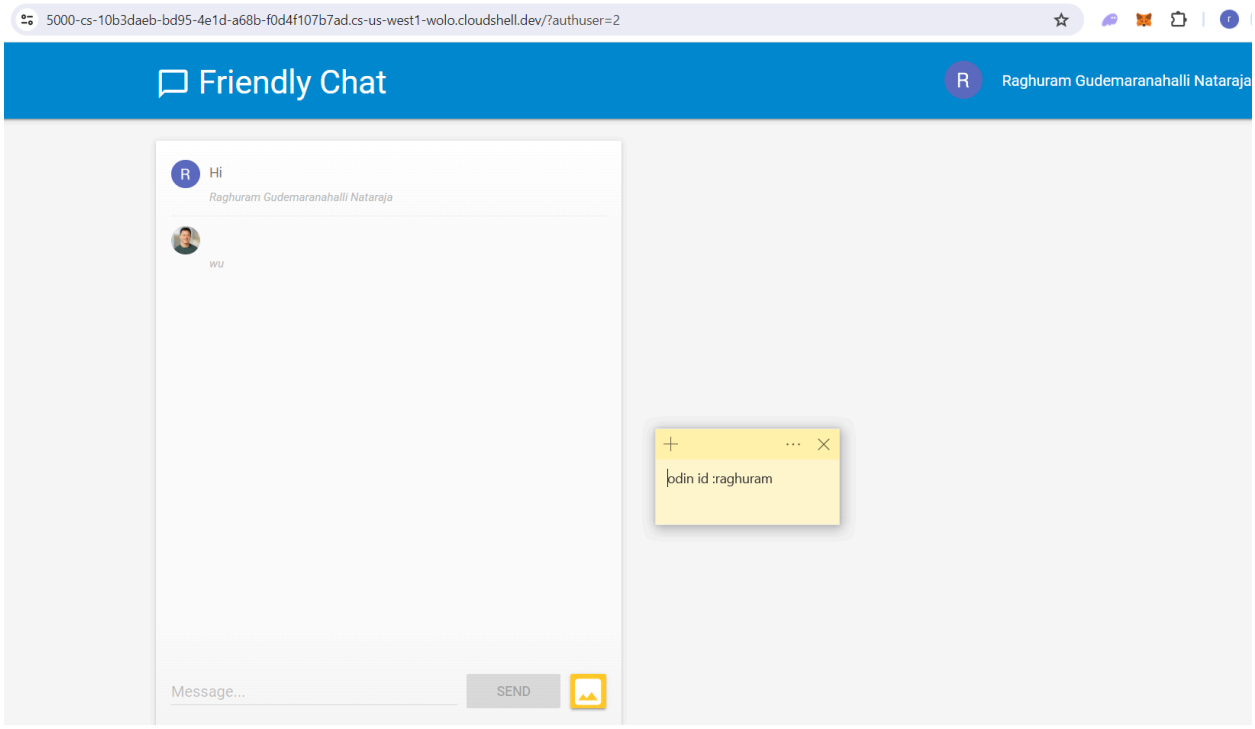
16. Test application with text messaging

- 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



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

```
var LOADING_IMAGE_URL =
'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?

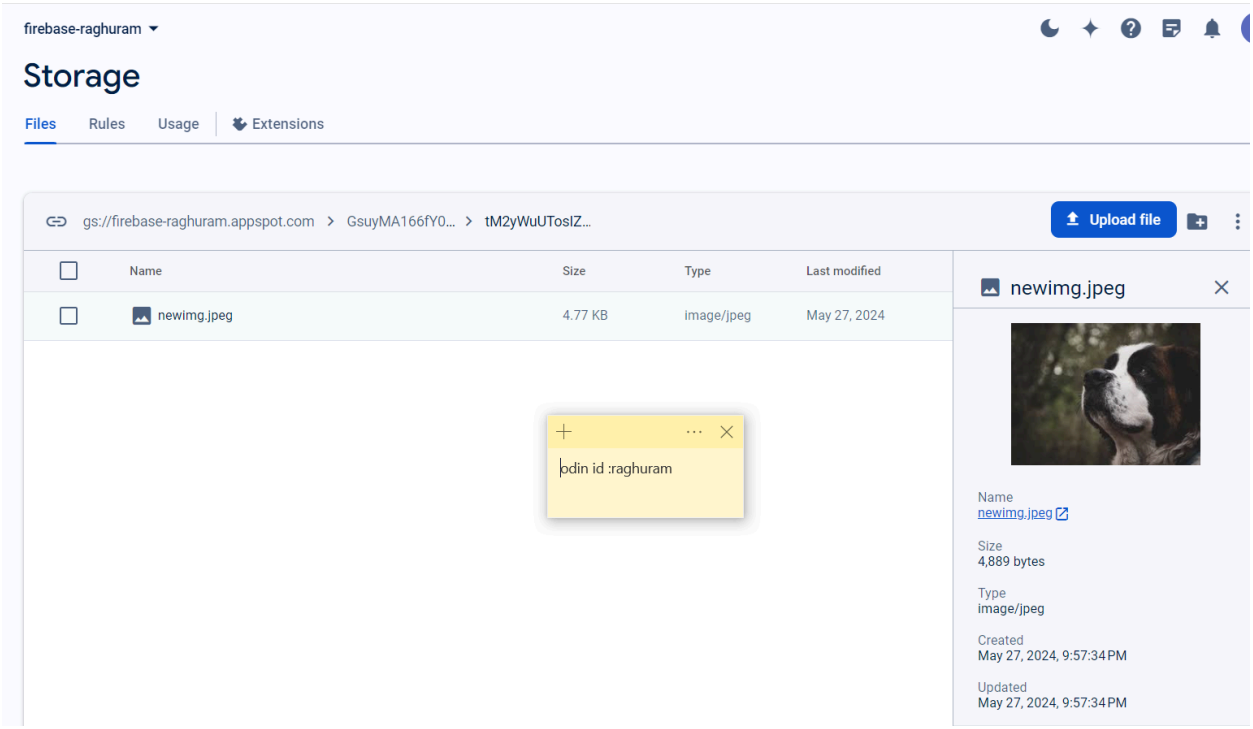
  imageUrl, storageUri are added in image url

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

  https://console.firebase.google.com/u/2/project/firebase-raghuram/storage/firebase-raghuram.appspot.com/files/~2FGsuyMA166fY01ZvzVxcInf7Gtam1~2Fpv5pXPbok4eQGvzJyH2K

  storage location:
  GsuyMA166fY01ZvzVxcInf7Gtam1/tM2yWuUTosIZAu5S9JCs/newimg.jpeg

- 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.