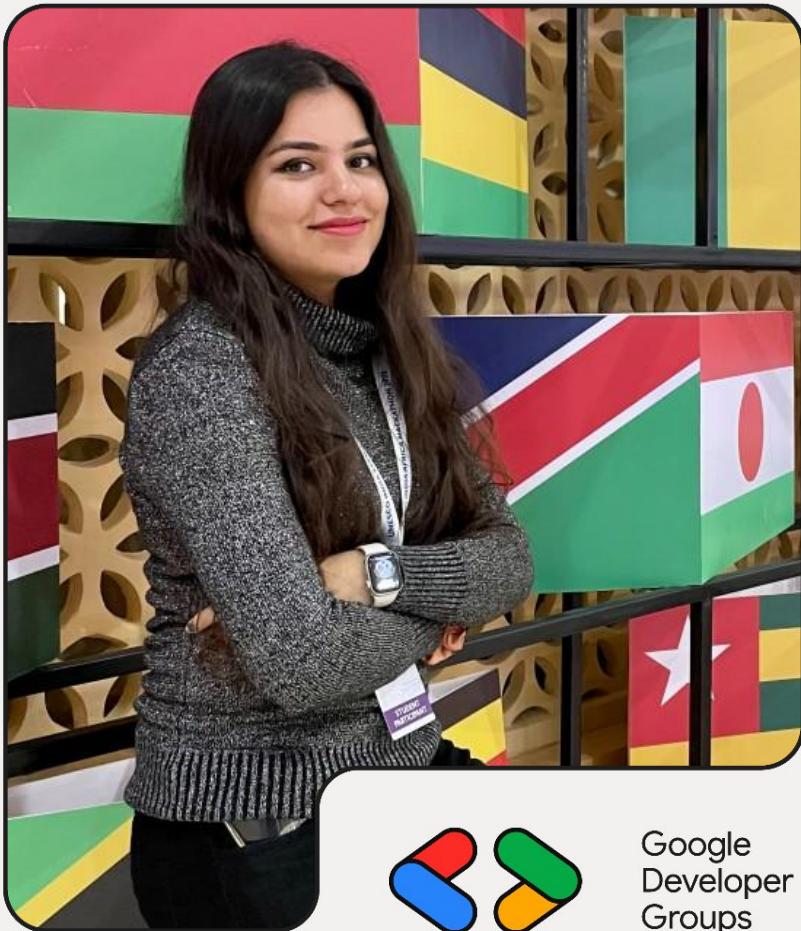


"From Static to Dynamic: Leveraging Short-Lived Credentials for Better Security in GCP

Sakshi Nasha
Learner



Google
Developer
Groups



\$whoami

Learner

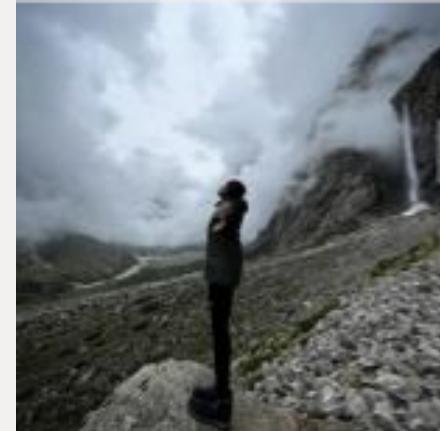
Innovator : SIH Winner, UIA

Public speaker

Athlete at heart :



Off the grid? Catch me just soaking up **nature** to recharge for the next big idea.



ਕੀ ਤੁਹਾਨੂੰ ਨੀਂਦ ਆ ਰਹੀ ਹੈ
ki tuhanu neend aa rahi
hai



Google Developer Groups



friday hai aaj



Google Developer Groups



AGENDA

1

What is Security and Authentication

2

Service Account in GCP

3

Access Control: Granting Roles

4

What is Static Creds and Dynamic Creds

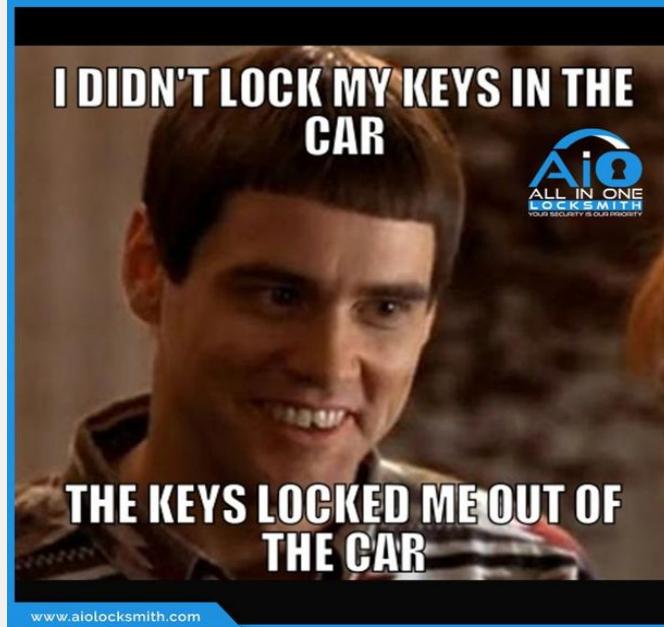
5

Generate Short-Lived Credentials in GCP

6

Short Lived Credentials - Game Changer

What is Security ?



Google Developer Groups

Authentication v/s Authorization



Google Developer Groups

Authentication v/s Authorization



Google Developer Groups

AUTHENTICATION



Who are you?

Verify the user's identity

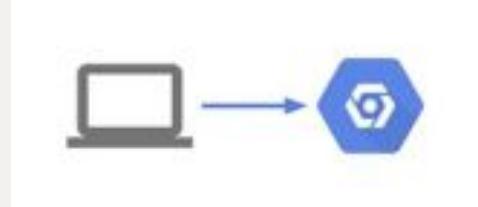
AUTHORIZATION



What are you allowed to do?

Determine user permissions

Authentication in GCP



- identity is confirmed through the use of some kind of credential.
- WHO AM I
- Proving that you are who you say you are.
- **gcloud auth login**



ਮੈਂ ਕਾਨ ਵੱਡੇ



Google Developer Groups

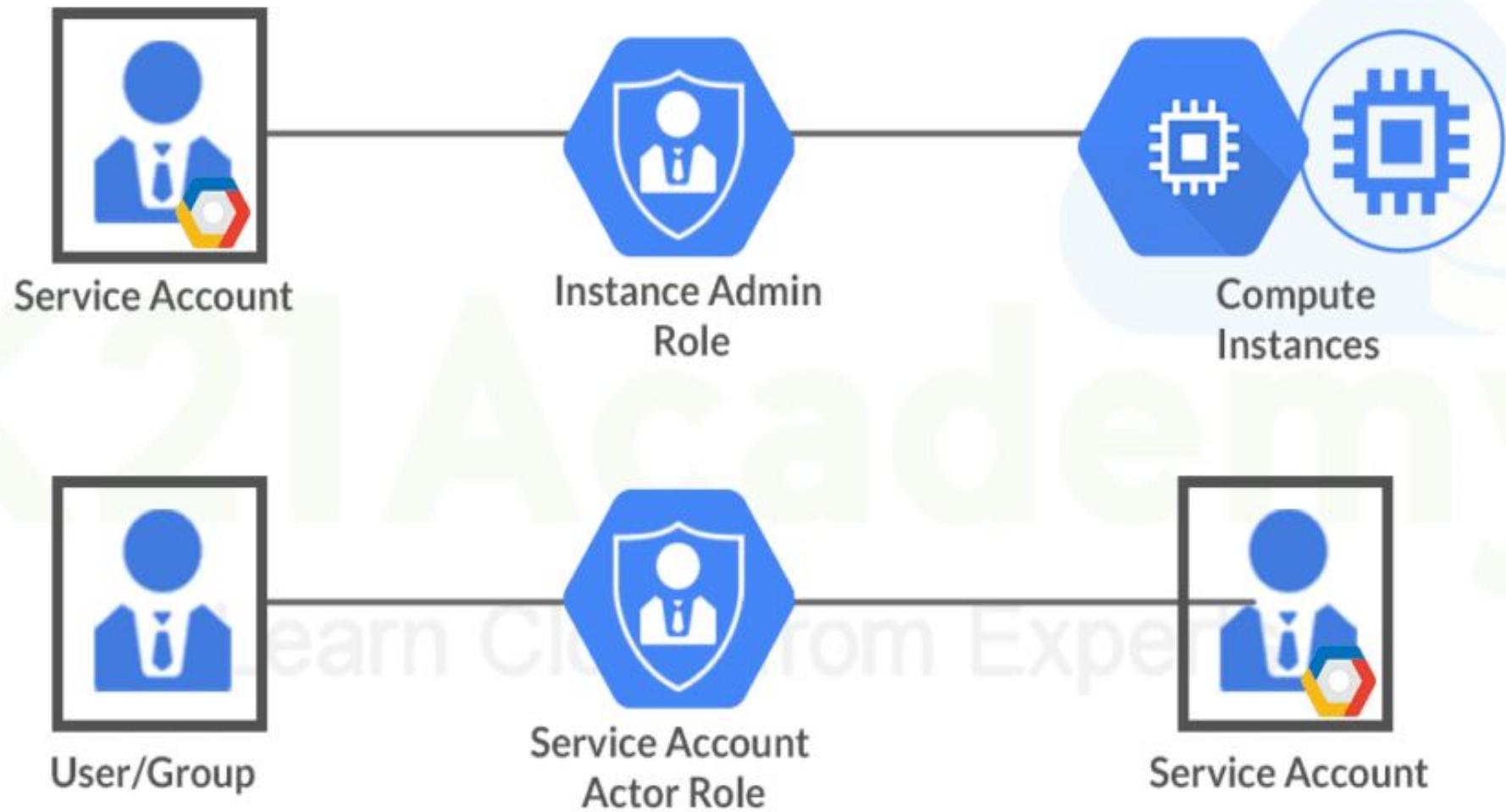
Service Account in GCP

- special type of Google account that grant permissions to virtual machines instead of end users.
- Ensure authorized API calls by authenticating itself
- When an application authenticates as a service account, it has access to all resources that the service account has permission to access.

GRANTING ROLES : Storage Object Viewer role

let a service account access resources in your project by granting it a role





Static v/s Dynamic Creds

- **Static secrets** are secrets that are pre-defined and do not change unless explicitly modified by an administrator or authorized user.

Examples: API keys, database passwords, or encryption keys unless intentionally rotated.

- **Dynamic secrets** are credentials or tokens that are generated programmatically and have a short lifespan. They are typically generated on-demand and automatically revoked after a specified time period or usage.

Examples : Temporary Access, Session Token or Short-lived tokens

Creating a Credential (Access key) with the GCP Service Accounts

The screenshot shows the Google Cloud IAM & Admin interface. The left sidebar includes links for IAM & Admin, IAM, PAM, Principal Access Boundary, Organizations (PREVIEW), Identity & Organization, Policy Troubleshooter, Policy Analyzer (NEW), Organization Policies, Service Accounts (selected), Workload Identity Federation, and Workforce Identity Federation. The main content area shows the Compute Engine default service account under the Keys tab. It displays a warning about service account keys being a security risk and a note about Google automatically disabling keys in public repositories. A section for adding a new key pair is present, along with a table of existing keys. One key is listed: Type (Service Account Key), Status (Active), Creation date (Apr 26, 2024), and Expiration date (Jan 1, 10000). A trash icon is shown next to the expiration date.

Google Cloud Search (Search /) for resources, docs, products, and more

Compute Engine default service account

DETAILS PERMISSIONS KEYS METRICS LOGS

Keys

⚠️ Service account keys could pose a security risk if compromised. We recommend you avoid downloading service account keys and instead use the [Workload Identity Federation](#). Learn more about the best way to authenticate service accounts on Google Cloud [Learn more](#).

ℹ️ Google automatically disables service account keys detected in public repositories. You can customize this behavior by using the 'iam.serviceAccountKeyExposureResponse' organization policy. [Learn more](#).

Add a new key pair or upload a public key certificate from an existing key pair.

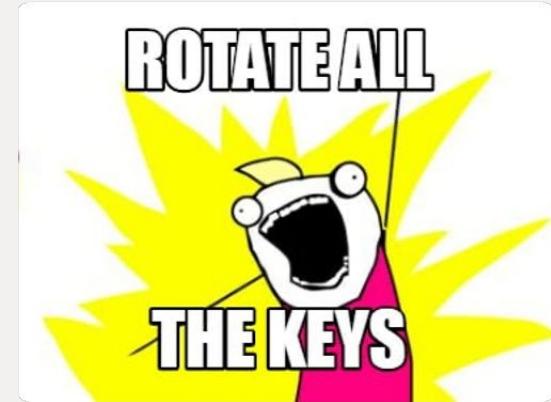
Block service account key creation using [organization policies](#).
[Learn more about setting organization policies for service accounts](#)

ADD KEY

Type	Status	Key	Creation date	Expiration date
Service Account Key	Active	[REDACTED]	Apr 26, 2024	Jan 1, 10000

Current Challenges faced with managing and storing secrets

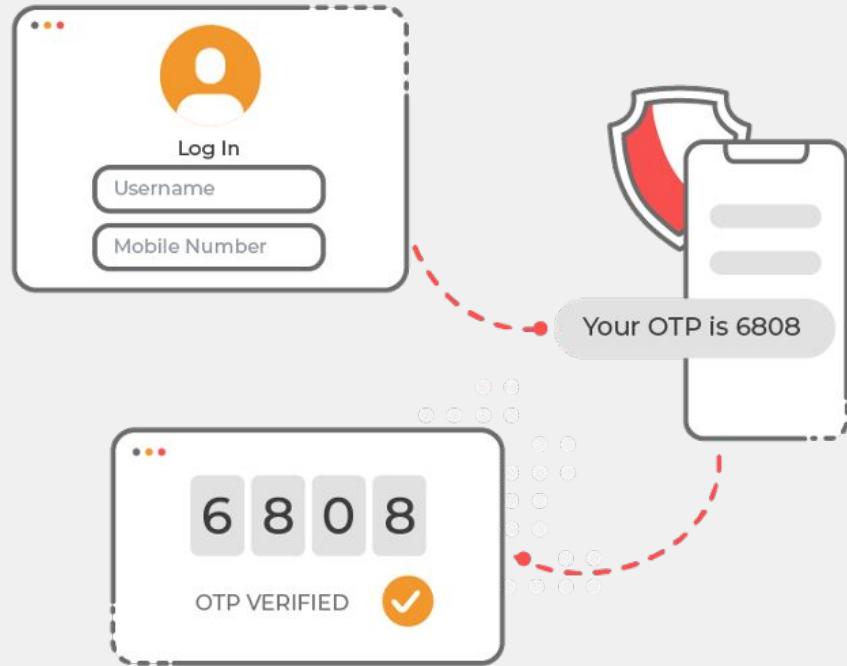
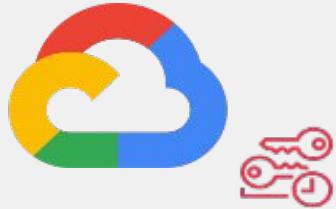
- Leakage, unauthorized access to secrets
- Balancing - Cost, Flexibility, Scalability : licensing fees, operational expenses.
- Manual Processes : key rotation, redistribution
- Integration Complexity : diverse ecosystem



Hard coding credentials

```
main.go  x
1 package main
2
3 import (
4     "fmt"
5     "os"
6 )
7
8 func main() {
9     databaseName := "53CR3TD4T4B453" ← Red arrow points here
10    secretKey := "5UP3R53CR3T"
11    secretPhrase := "Always know where your towel is. — Douglas Adams, The Hitchhiker's Guide to the Galaxy"
12
13    var dbName string
14    var dbPass string
15
16    fmt.Println("Please enter database name:")
17    fmt.Scanf("%s", &dbName)
18
19    fmt.Println("Please enter database password:")
20    fmt.Scanf("%s", &dbPass)
21
22    if dbName == databaseName && dbPass == secretKey {
23        fmt.Println("Welcome to the database!")
24        fmt.Println("Your secret phrase is: ", secretPhrase)
25        os.Exit(0)
26    }
27    fmt.Println("Sorry, wrong database name or password")
28
29 }
```

**TODAY, we will
generate a
short-lived token
for GCP
application**



Create short-lived credentials for a service account

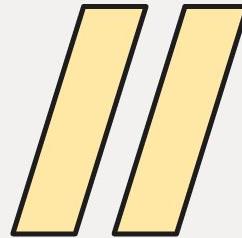
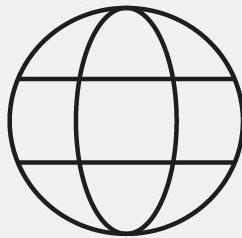
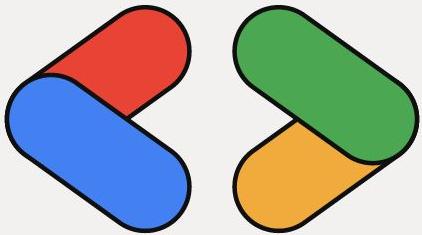
- You can generate an OAuth 2.0 access token by using the gcloud CLI, the REST API, or the Cloud Client Libraries.
- We will leverage use of The Security Token Service STS : creates short-lived access token to Google Cloud resources.

Create short-lived credentials for a service account

- POST <https://sts.googleapis.com/v1/token>
 - Response :

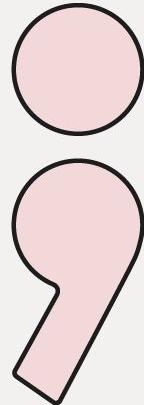
```
{ "access_token": "*****",
"issued_token_type": "urn:ietf:params:oauth:token-type:access_token",
"token_type": "Bearer",
"expires_in": 3216
}
```

The lifetime of the token is min = 1 hour (3,600 seconds) to max = 12 hours (43,200 seconds)



HOW Dynamic Creds is a Game Changer ?

- Examples : Financial applications (ICICI Bank, HDFC Bank), Reset Password Link, Gmail Login



Google
Developer
Groups

Perks of using Short lived Tokens

1



Enhancing Security

-
- Short-lived tokens have a **limited lifespan**, reducing the exposure window for potential attacks.
 - Regular token expiration forces users to **re-authenticate**, ensuring better security.

2



Mitigating Token Abuse

-
- By making tokens short lived, we limit the time an attacker can use to abuse a stolen token.
 - Thus, **minimizing the risk window** significantly

3



Least Privilege Principle

-
- A user should only have access to what they absolutely **need** and not what they want.
 - When permissions **change** (e.g., user roles or access levels), short-lived tokens automatically reflect the updates upon renewal.

Connect with me



Google Developer Groups



**“DREAM, BELIEVE and
Make it happen”**



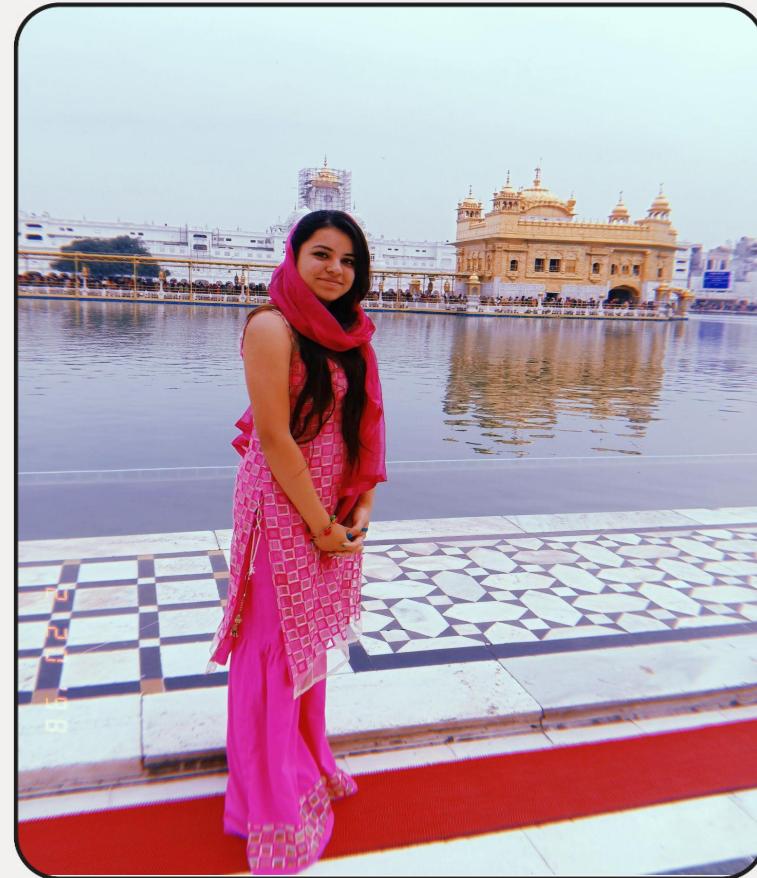


Google Developer Groups

**“Asli punjabi jithe vi
jaave chhaa jave”**



@Avantika_0902

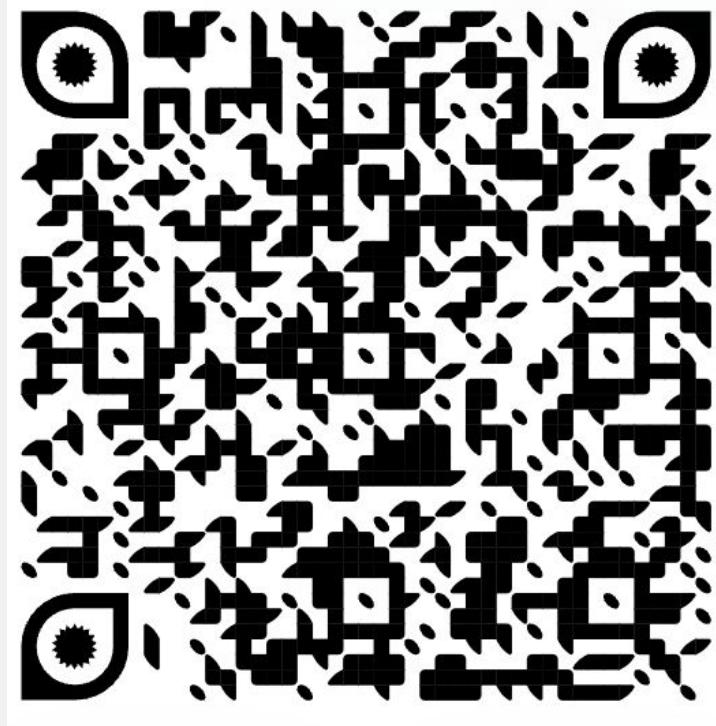


Thank you all
for joining! ✨



Google Developer Groups

<https://bit.ly/SakshiFeed>



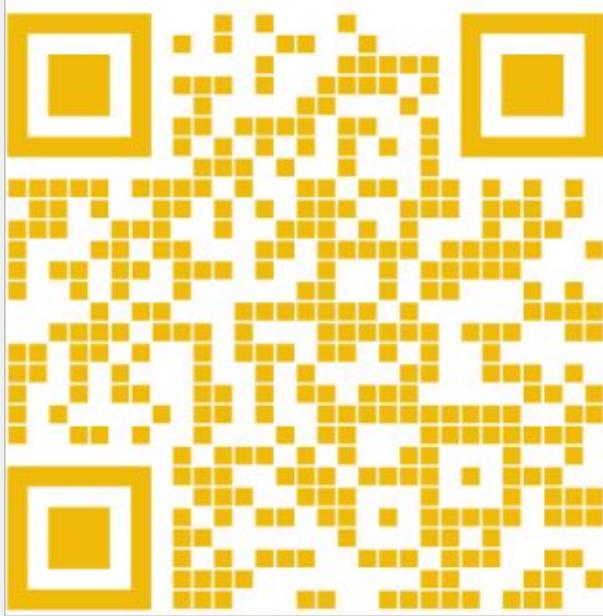


Tussi
ja rahe
ho?



Google Developer Groups

Future References



Github Repo

(DOWNSCOPE the short-lived credential)



My Blog



Google Developer Groups

Smart India Hackathon Winner



Google Developer Groups

**“GDG jalandhar nu invite kita
tuhada dhanvad, Vada changa
laga si”**

Janta : tuhada dhanvad



Google Developer Groups