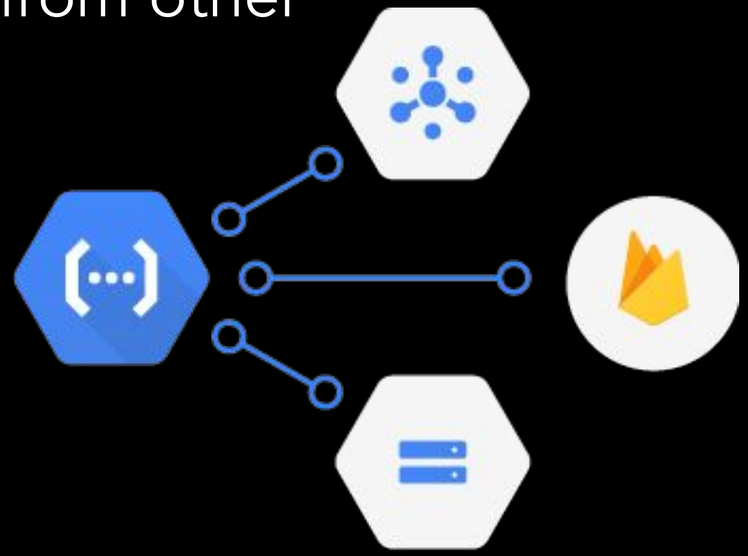
GOOGLE CLOUD PLATFORM

CLOUD FUNCTIONS

WHAT ARE GOOGLE CLOUD FUNCTIONS

- Google Cloud Functions is a serverless execution environment for building and connecting cloud services.
- Fully-managed, serverless environment where Google handles infrastructure, operating systems, and runtime environments
- Each Cloud Function runs in its own isolated secure execution context, scales automatically, and has a lifecycle independent from other functions.
- Function-as-a-service approach
- Use familiar languages like Javascript and Node.js



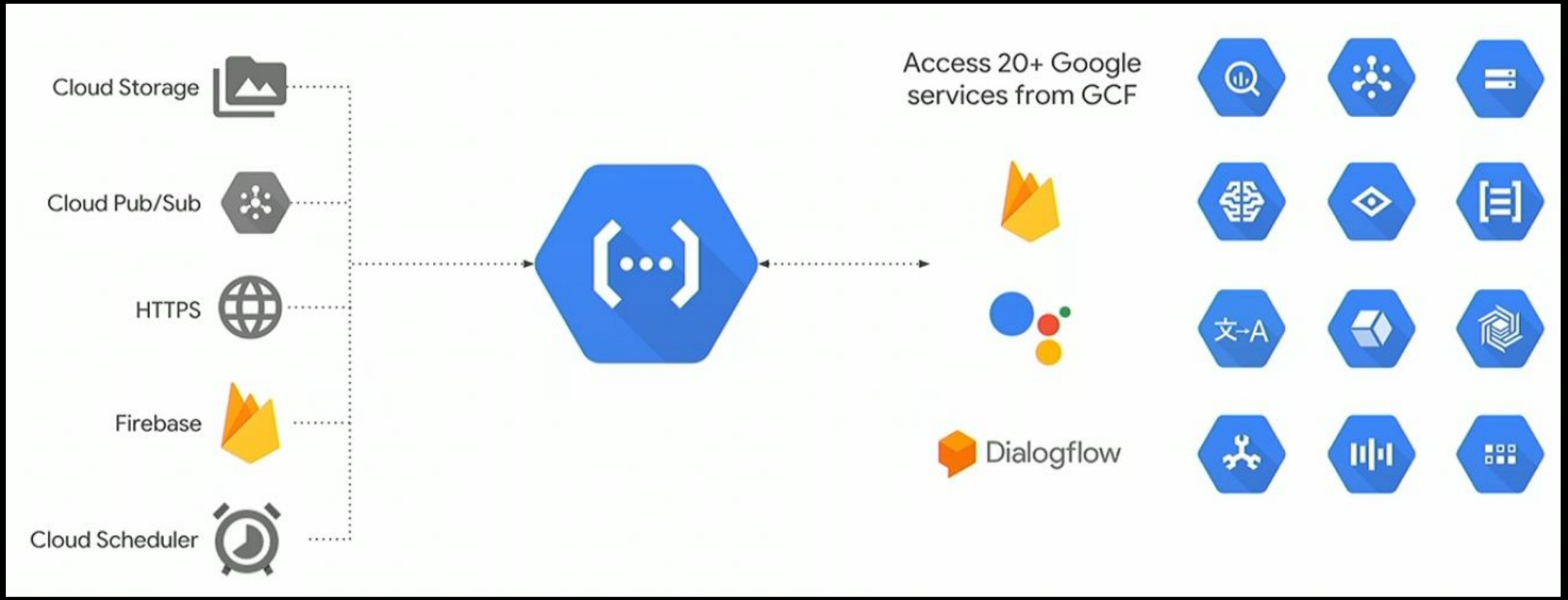
CONNECT AND EXTEND CLOUD SERVICES

- Cloud Functions provides a connective layer of logic that lets you write code to connect and extend cloud services.
- Listen and respond to a file upload to Cloud Storage, a log change, or an incoming message on a Cloud Pub/Sub topic.
- Augments existing cloud services and allows you to address an increasing number of use cases with arbitrary programming logic.
- Has access to the Google Service Account credential and are thus seamlessly authenticated with the majority of Google Cloud Platform services.
- Cloud Functions are supported by numerous Google Cloud client libraries, which further simplify these integrations.

EVENTS

- Events are things that happen within your cloud environment that you might want to take action on.
- These might be changes to data in a database, files added to a storage system, or a new virtual machine instance being created.
- Currently, Cloud Functions supports events from the following providers:
  - HTTP—invoke functions directly via HTTP requests.
  - Cloud Storage
  - Cloud Pub/Sub
  - Firebase (DB, Storage, Analytics, Auth)
  - Stackdriver Logging—forward log entries to a Pub/Sub topic by creating a sink. You can then trigger the function.
- When an event triggers the execution of your Cloud Function, data associated with the event is passed via the function's parameters.
- The type of event determines the parameters passed to your function.

HOW IT ALL FITS



TRIGGERS

- Creating a response to an event is done with a *trigger*.
- A trigger is a declaration that you are interested in a certain event or set of events.
- Binding a function to a trigger allows you to capture and act on events.
- Functions and triggers are bound to each other on a many-to-one basis during deployment

TRIGGER	COMMAND-LINE FLAG
HTTP	--trigger-http
Google Cloud Storage	--trigger-bucket *BUCKET_NAME*
Google Cloud Pub/Sub	--trigger-topic *TOPIC_NAME*
Other sources (e.g. Firebase)	--trigger-event *EVENT_TYPE* --trigger-resource *RESOURCE*

SERVERLESS

- Cloud Functions removes the work of managing servers, configuring software, updating frameworks, and patching operating systems.
- The software and infrastructure are fully managed by Google so that you just add code.
- Furthermore, provisioning of resources happens automatically in response to events.
- This means that a function can scale from a few invocations a day to many millions of invocations without any work from you.

Compute time in 100ms increments

No hourly, daily, or monthly minimums

Free tier

SERVERLESS

Customer examples

Meetup

SMART PARKING

HomeAway

Turner

semios

Guesswork

Incentro

Serverless application backends

Integration with third party services and API's

Mobile backends

IoT backends

Real-time data processing

File processing

Stream processing

Event-driven extract, transform, load (ETL)

Intelligent applications

Virtual assistants and chatbots

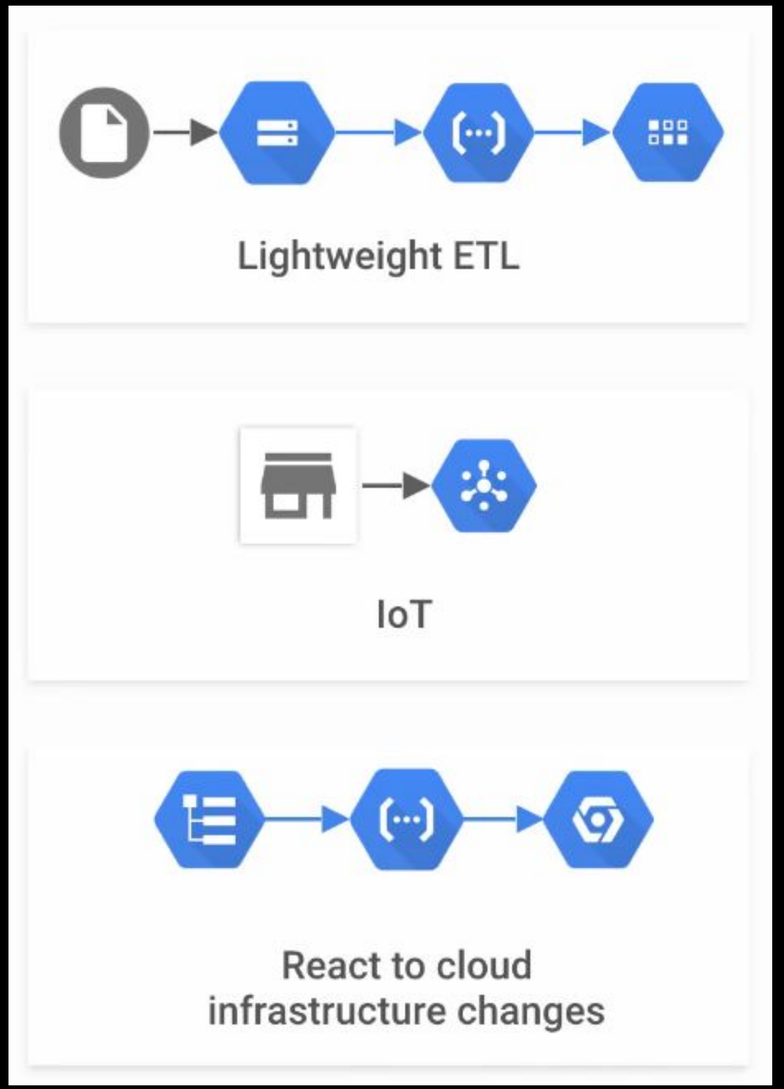
Video and image analysis

Sentiment analysis

NEXT 18

USE CASES

USE CASE	DESCRIPTION
Data processing / ETL	Listen and respond to Cloud Storage
Webhooks	Respond to events via a simple HTTP trigger
Lightweight APIs	Compose applications from lightweight, loosely coupled bits of logic
Mobile backend	Listen and respond to events from Firebase Analytics, Realtime Database, Authentication, and Storage.
IoT	Process, transform and store data for Iot events from Pub/Sub.



EXECUTION ENVIRONMENT

- Runtimes
  - Cloud Functions supports multiple languages and runtimes.
  - Updates to the language runtimes are generally done automatically.

- **Concurrency**
  - Cloud Functions can start multiple function instances to scale your function up to meet the current load.
  - These instances run in parallel, which results in having more than one parallel function execution.
  - However, each function instance handles only one concurrent request at a time.
  - Concurrent requests are processed by different function instances and do not share variables or local memory.

EXECUTION ENVIRONMENT

- **Stateless functions**
  - Functions must be stateless - one function invocation should not rely on in-memory state set by a previous invocation.
  - If you need to share state across function invocations, your function should use a service such as Cloud Datastore, Cloud Firestore or Cloud Storage to persist data.

- **Cold starts**
  - A new function instance is started in two cases:
    - When you deploy your function.
    - When a new function instance is automatically created to scale up to the load, or occasionally to replace an existing instance.
  - Starting a new function instance involves loading the runtime and your code.
  - Requests that include function instance startup, can be slower than requests hitting existing function instances.

EXECUTION ENVIRONMENT

- **Function instance lifespan**
  - The environment running a function instance is typically resilient and reused by subsequent function invocations
  - When one function execution ends, another function invocation can be handled by the same function instance.

- **Function scope versus global scope**
  - A single function invocation results in executing only the body of the function declared as the entry point.
  - The global scope in the function file, which is expected to contain the function definition, is executed on every cold start, unless the instance has already been initialized.
  - You should not depend on the total number of or timing of global scope executions as they depend on the autoscaling.

EXECUTION ENVIRONMENT

- **Function execution timeline**
  - A function has access to the resources requested (CPU and memory) only for the duration of function execution
  - Code run outside of the execution periods is not guaranteed to execute, and it can be stopped at any time

- **Execution guarantees**
  - Functions are typically invoked once for each incoming event
  - Cloud Functions does not guarantee a single invocation in all cases because of differences in error scenarios
  - HTTP functions are invoked at *most* once.
  - Background functions are invoked at *least* once.

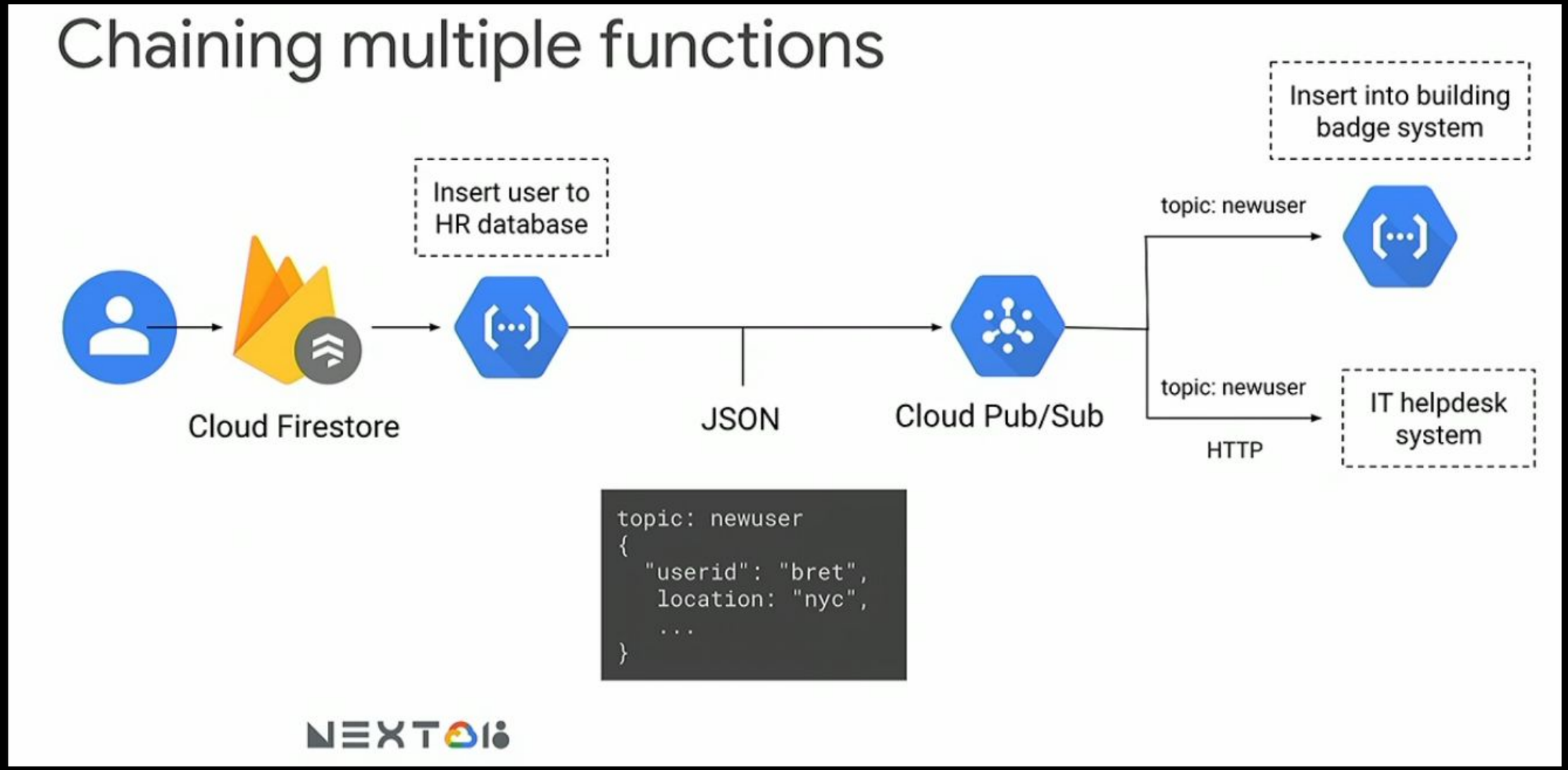
EXECUTION ENVIRONMENT

- **Errors**
  - HTTP functions should return appropriate HTTP status codes which denote an error.
  - Background functions should log and return an error message
- **Timeout**
  - Function execution time is limited by the timeout duration, which you can specify at function deployment time
  - A function times out after 1 minute by default, but you can extend this period up to 9 minutes.
- **Network**
  - Your function can access the public internet using standard libraries offered by the runtime or third-party providers.
  - Your code should either use a library that handles closed connections well as they close automatically after 2 minutes of no activity.

EXECUTION ENVIRONMENT

- **Multiple functions**
  - Every deployed function is isolated from all other functions - even those deployed from the same source file.
  - In particular, they don’t share memory, global variables, file systems, or other state.
  - To share data across deployed functions, you can use storage services like Cloud Datastore, Cloud Firestore, or Cloud Storage.
  - Alternatively, you can invoke one function from another, using their appropriate triggers.

CHAINING MULTIPLE FUNCTIONS



CLOUD FUNCTIONS AND FIREBASE

- For Firebase developers, Cloud Functions for Firebase provides a way to extend the behavior of Firebase and integrate Firebase features through the addition of server-side code.
  - Cloud Functions for Firebase is optimized for Firebase developers:
    - Firebase SDK to configure your functions through code
    - Integrated with Firebase Console and Firebase CLI
    - The same triggers as Google Cloud Functions, plus Firebase Realtime Database, Firebase Authentication, and Firebase Analytics triggers