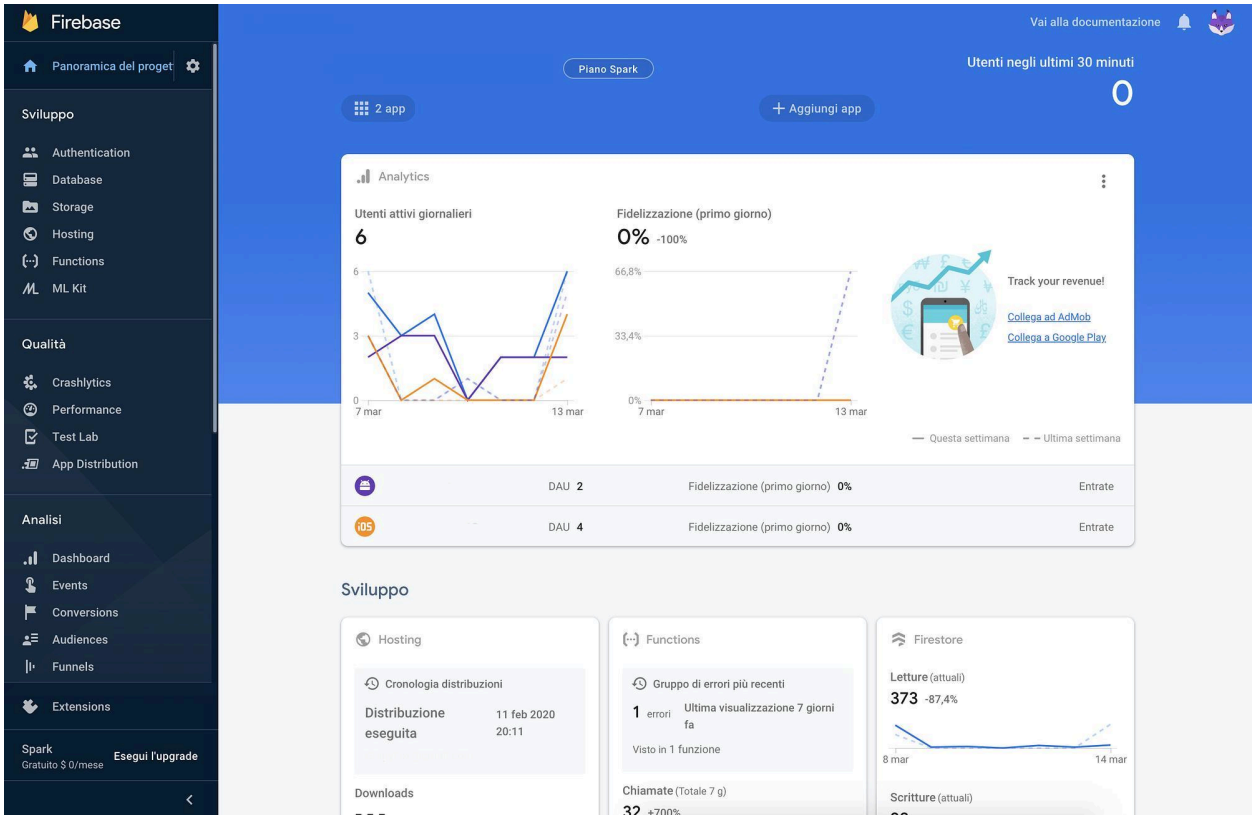
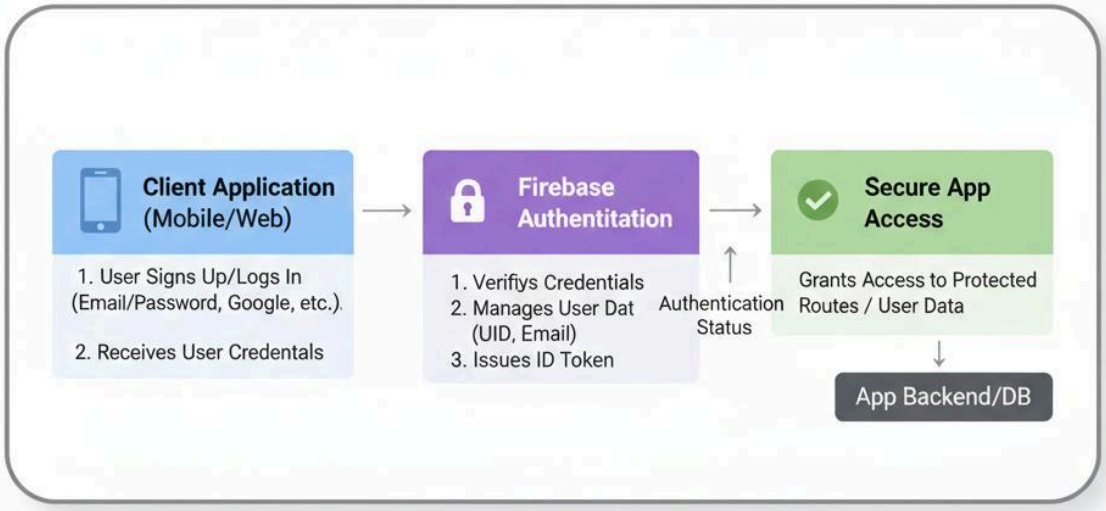
How to Use Firebase for Cybersecurity Applications

A Practical Tutorial for Students, Developers, and Security Professionals

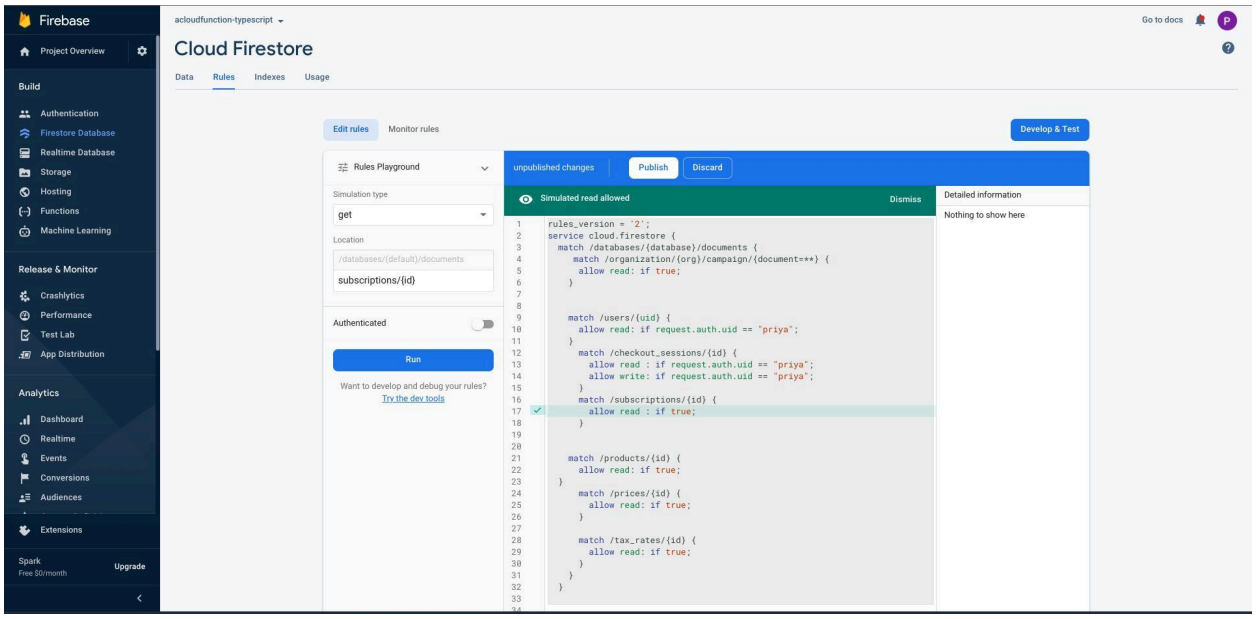


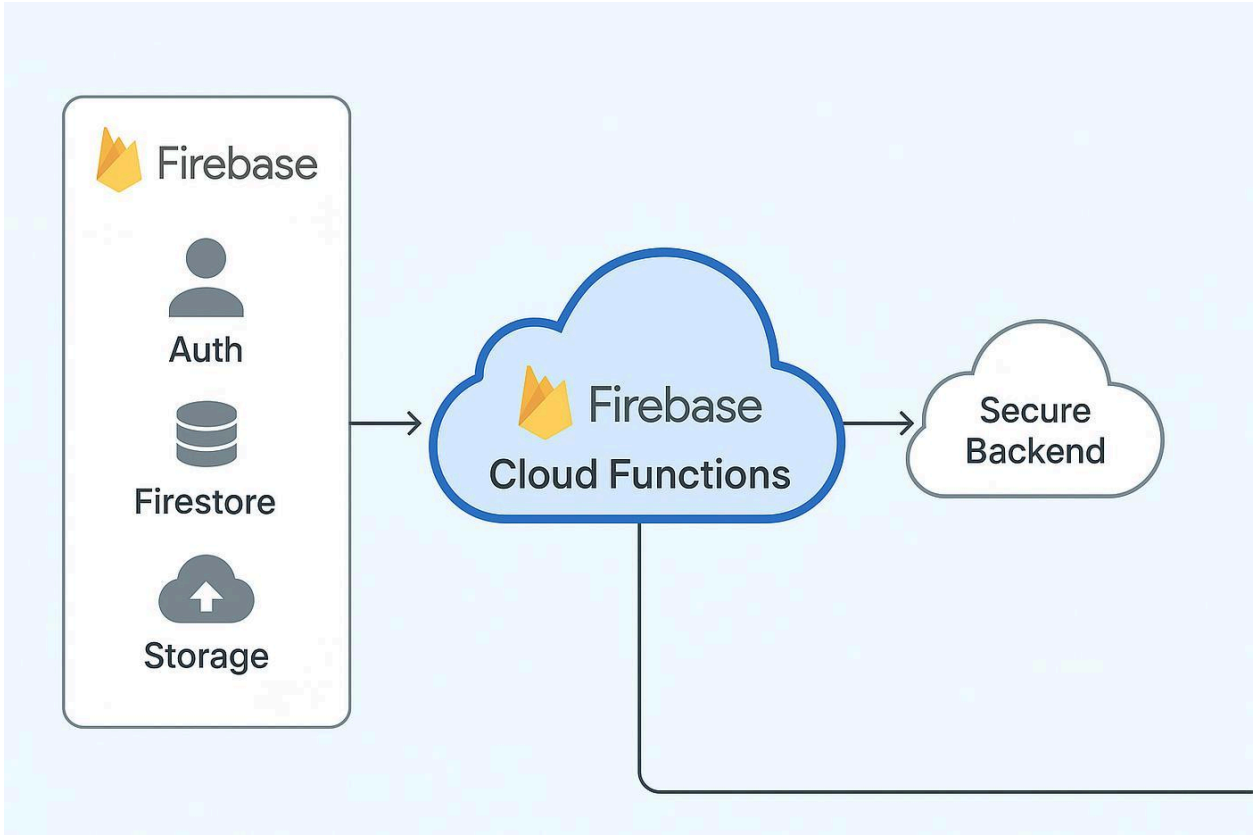
Using Firebase Authentitation

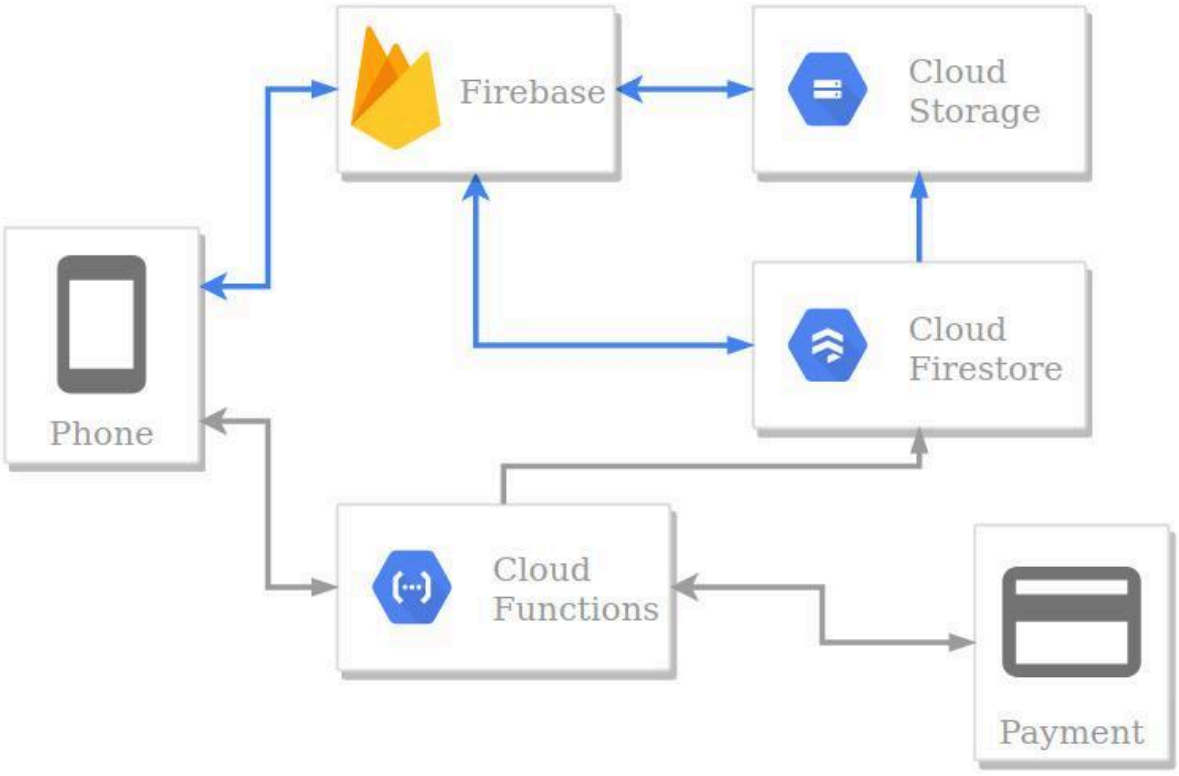
Simplified User Security

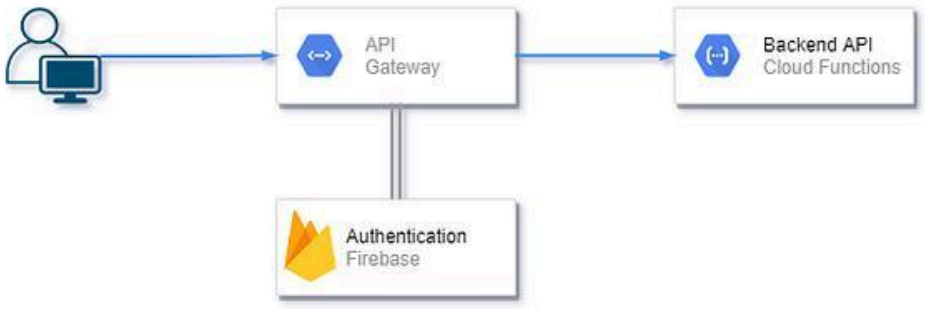


Easy, Secure User Management


Firebase
Project Overview
Build
Authentication
Firestore Database
Realtime Database
Storage
Hosting
Functions
Machine Learning
Release & Monitor
Crashlytics
Performance
Test Lab
App Distribution
Analytics
Dashboard
Realtime
Events
Conversions
Audiences
Extensions
Spark
Free $0/month
Upgrade
acloudfunction-typescript
Go to docs
Cloud Firestore
Data
Rules
Indexes
Usage
Edit rules
Monitor rules
Develop & Test
Rules Playground
Simulation type
get
Location
/databases/(default)/documents
subscriptions/{id}
Authenticated
Run
Want to develop and debug your rules?
Try the dev tools
unpublished changes
Publish
Discard
Simulated read allowed
Dismiss
Detailed information
Nothing to show here
rules_version = '2';
service cloud.firestore {
match /databases/{database}/documents {
match /organization/{org}/campaign/{document=**} {
allow read: if true;
}
match /users/{uid} {
allow read: if request.auth.uid == "priya";
}
match /checkout_sessions/{id} {
allow read : if request.auth.uid == "priya";
allow write: if request.auth.uid == "priya";
}
match /subscriptions/{id} {
allow read : if true;
}
match /products/{id} {
allow read: if true;
}
match /prices/{id} {
allow read: if true;
}
match /tax_rates/{id} {
allow read: if true;
}
}
}


Firebase
Auth
Firestore
Storage
Firebase
Cloud Functions
Secure
Backend





Introduction

Cybersecurity applications require secure user authentication, encrypted communications, access control, real-time monitoring, and event logging. Google's **Firebase** provides many cloud-based services that help developers build secure applications without managing their own backend infrastructure.

This tutorial demonstrates how Firebase can be used to develop cybersecurity applications such as:

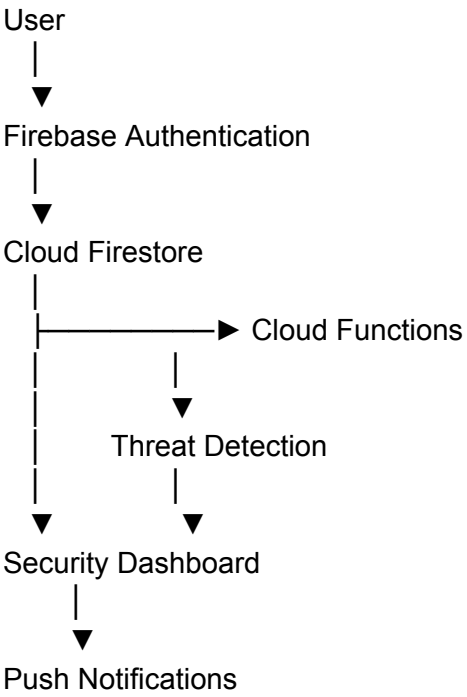
- Intrusion Detection Systems (IDS)
- Security Monitoring Dashboards
- Digital Forensics Tools
- Threat Intelligence Applications
- Malware Reporting Systems
- Incident Response Platforms
- USB Device Monitoring Applications
- Security Awareness Applications

What is Firebase?

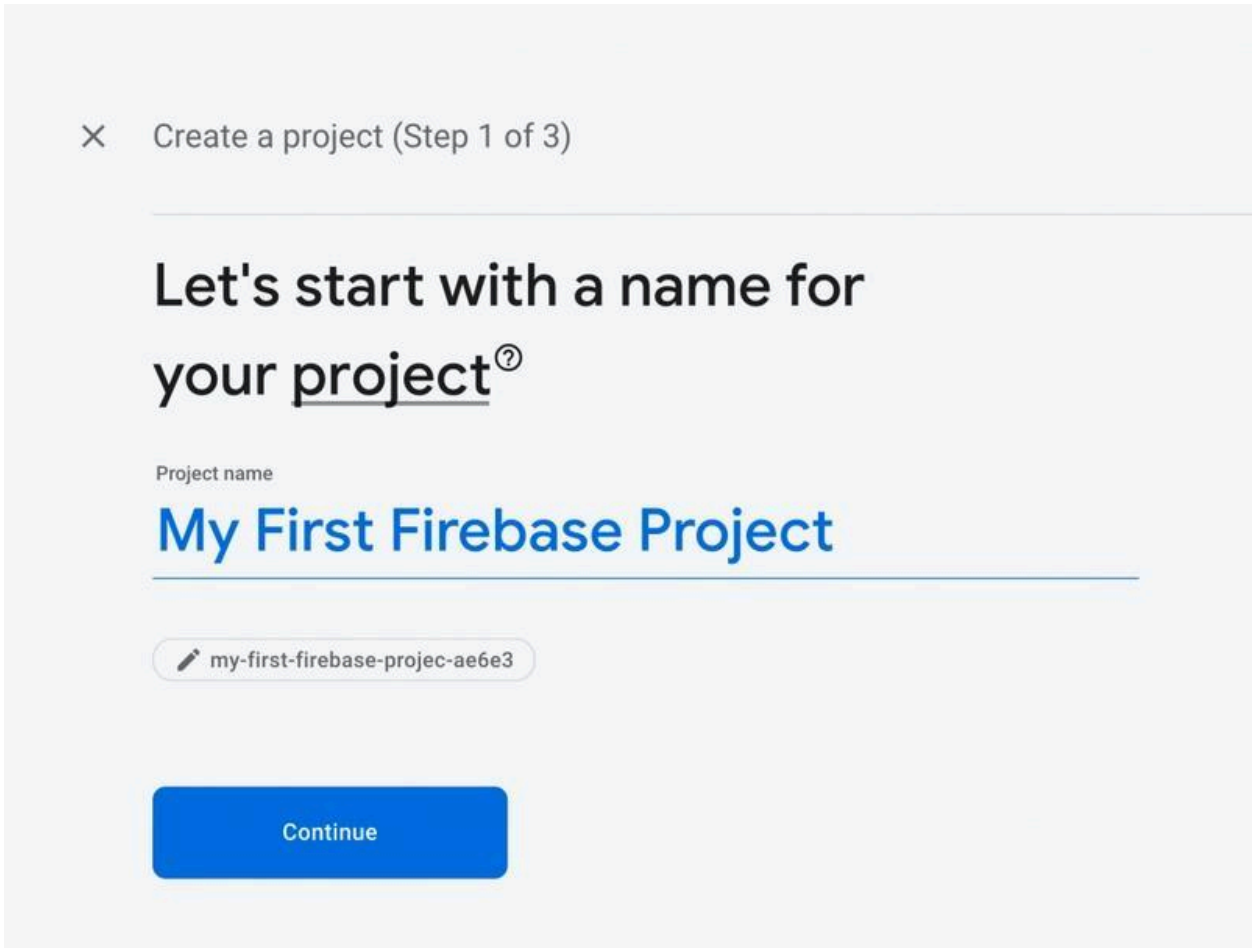
Firebase is a Backend-as-a-Service (BaaS) platform from Google that provides cloud services including:

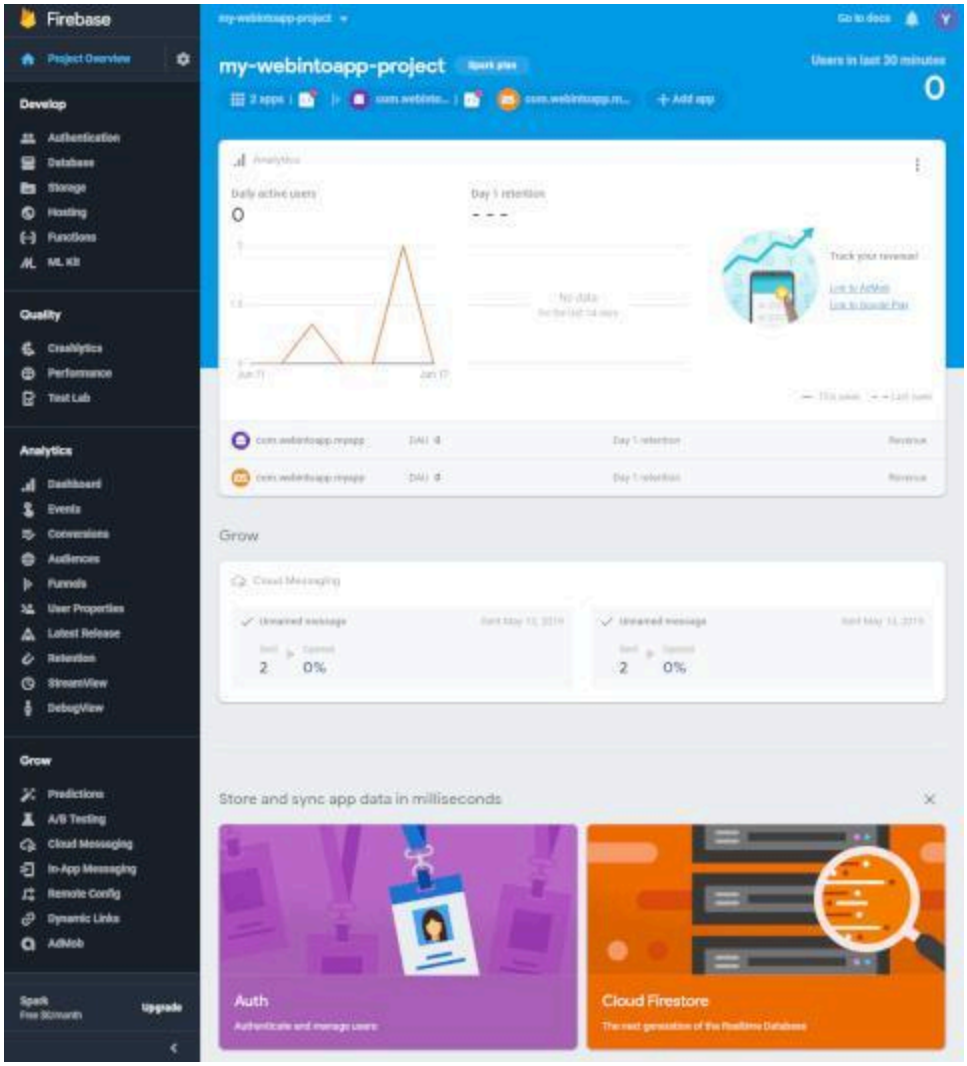
Service	Cybersecurity Use
Authentication	Secure login
Firestore Database	Store security logs
Cloud Functions	Automate incident response
Storage	Save malware samples and reports
Cloud Messaging	Send attack alerts
Analytics	Detect suspicious behavior
App Check	Protect APIs from abuse
Crashlytics	Monitor application failures

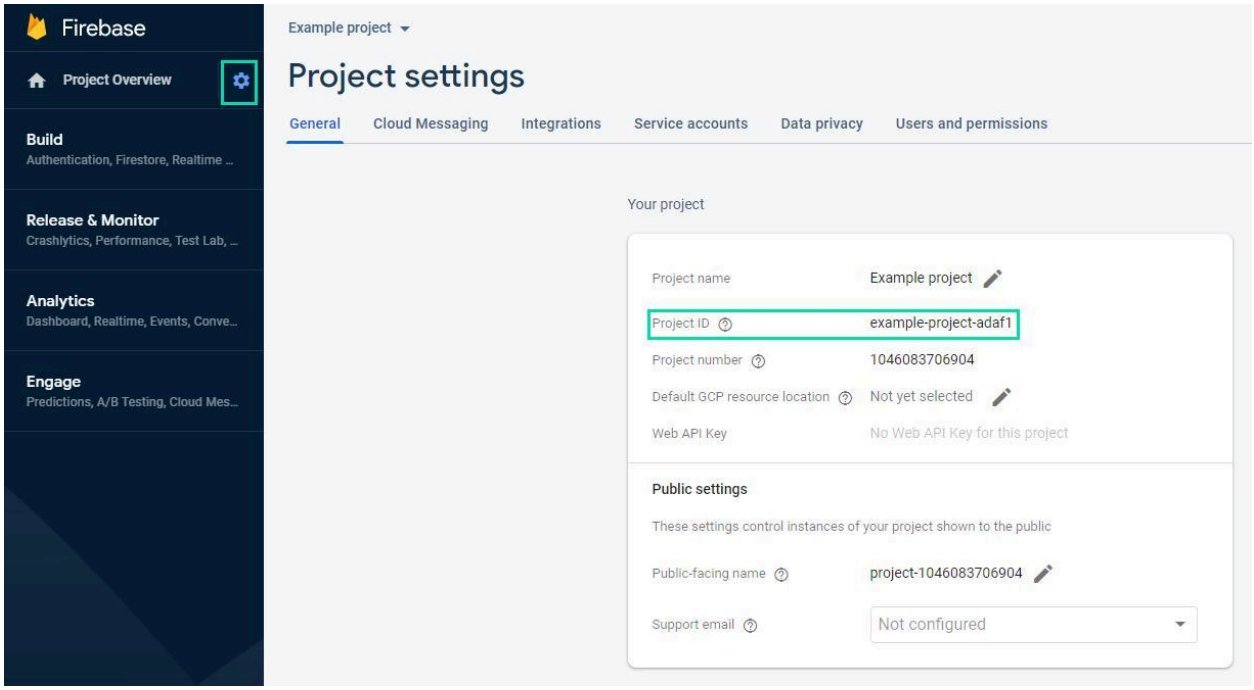
Cybersecurity Architecture

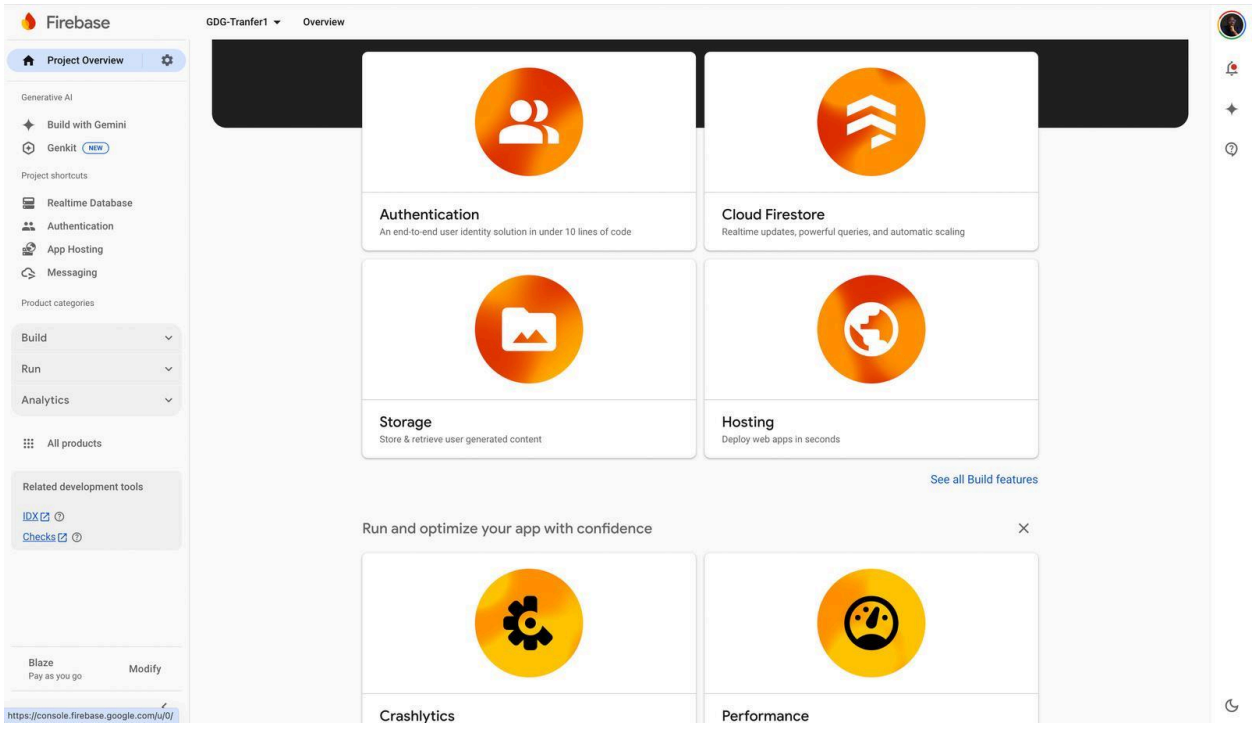


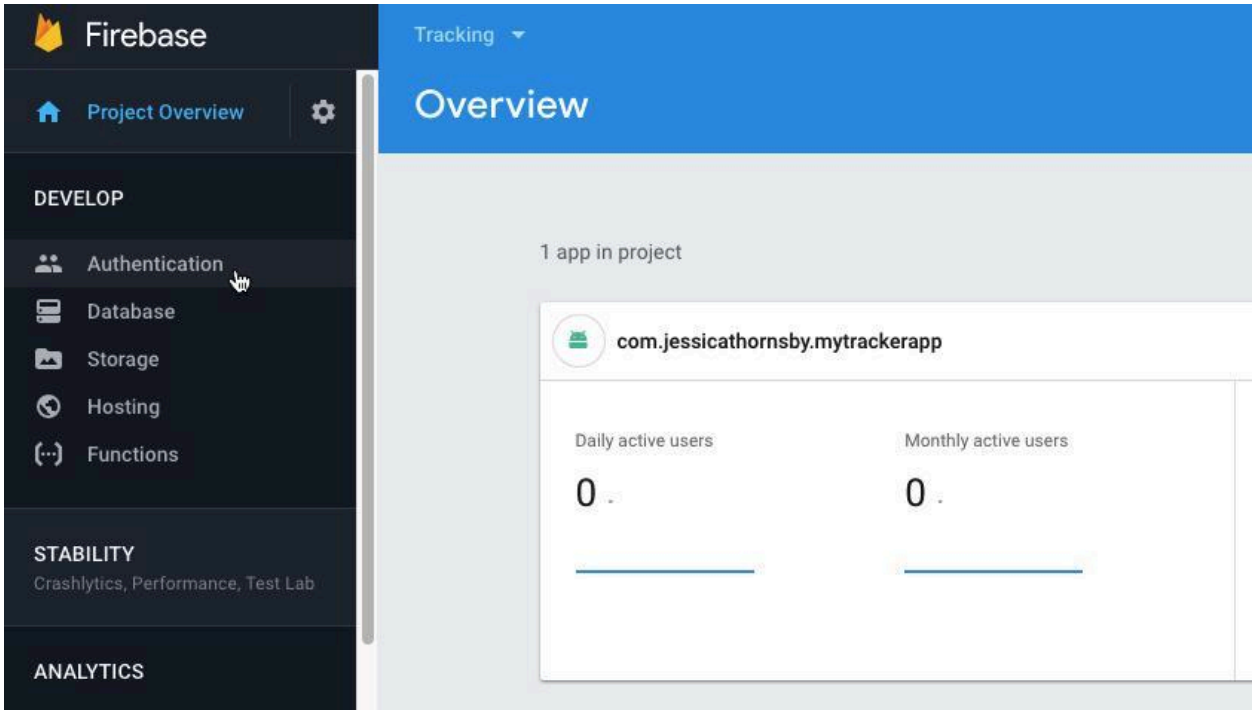
Step 1: Create a Firebase Project


Create a project (Step 1 of 3)
Let's start with a name for your project
Project name
My First Firebase Project
my-first-firebase-projec-ae6e3
Continue


Firebase
Project Overview
Develop
Authentication
Database
Storage
Hosting
Functions
ML Kit
Quality
Crashlytics
Performance
Test Lab
Analytics
Dashboard
Events
Conversions
Audiences
Funnels
User Properties
Latest Release
Retention
StreamView
DebugView
Grow
Predictions
A/B Testing
Cloud Messaging
In-App Messaging
Remote Config
Dynamic Links
AdMob
Spark
Free $0/month
Upgrade
my-webintoapp-project
Go to docs
my-webintoapp-project
Spark plan
Users in last 30 minutes
0
+ Add app
Daily active users
0
Day 1 retention
Grow
Store and sync app data in milliseconds
Auth
Cloud Firestore


Firebase
Project Overview
Build
Authentication, Firestore, Realtime ...
Release & Monitor
Crashlytics, Performance, Test Lab, ...
Analytics
Dashboard, Realtime, Events, Conve...
Engage
Predictions, A/B Testing, Cloud Mes...
Example project
Project settings
General
Cloud Messaging
Integrations
Service accounts
Data privacy
Users and permissions
Your project
Project name
Example project
Project ID
example-project-adaf1
Project number
1046083706904
Default GCP resource location
Not yet selected
Web API Key
No Web API Key for this project
Public settings
These settings control instances of your project shown to the public
Public-facing name
project-1046083706904
Support email
Not configured


Firebase
GDG-Tranfer1
Overview
Project Overview
Generative AI
Build with Gemini
Genkit
NEW
Project shortcuts
Realtime Database
Authentication
App Hosting
Messaging
Product categories
Build
Run
Analytics
All products
Related development tools
IDX
Checks
Blaze
Pay as you go
Modify
https://console.firebase.google.com/u/0/
Authentication
An end-to-end user identity solution in under 10 lines of code
Cloud Firestore
Realtime updates, powerful queries, and automatic scaling
Storage
Store & retrieve user generated content
Hosting
Deploy web apps in seconds
See all Build features
Run and optimize your app with confidence
Crashlytics
Performance



Open Firebase Console

Visit:

https://console.firebase.google.com

Click

Create Project

Example

CyberSecurityMonitor

Choose

- Enable Google Analytics
- Continue

After several seconds your cloud backend is ready.

Step 2: Add Your Application

Select

- Android
- Web
- iOS

For Android:

Package Name

com.company.cybersecurity

Download

google-services.json

Place it inside

app/

Step 3: Enable Firebase Authentication

Authentication protects your security application.

Navigate to

Authentication

Get Started

Enable

- Email/Password
- Google
- Microsoft
- Anonymous (optional)

Example login screen

Email

Password

Login

Example Java Authentication Code

```
FirebaseAuth auth = FirebaseAuth.getInstance();

auth.signInWithEmailAndPassword(email,password)
.addOnCompleteListener(task -> {

    if(task.isSuccessful()){

        Log.d("LOGIN","Success");

    }

});
```

Why Authentication Matters

Cybersecurity software should never allow anonymous access to:

- Incident reports
- Malware databases
- Security alerts
- User logs
- Threat intelligence

Step 4: Create Firestore Database

Firestore stores security information.

Examples:

Threat Reports

User Activity

Login History

Malware Samples

Security Alerts

Audit Logs

Example collection

Threats

Document

ThreatID001

{

attack:"Ransomware",

severity:"Critical",

date:"2026-08-03"

}

Java Example

FirebaseFirestore db = FirebaseFirestore.getInstance();

Map<String,Object> report = new HashMap<>();

report.put("Attack","Brute Force");

```
report.put("Severity","High");

db.collection("Threats")
.add(report);
```

Step 5: Secure Firestore with Rules

One of the biggest cybersecurity mistakes is leaving databases public.

Bad Rule

```
allow read, write: if true;
```

Good Rule

```
service cloud.firestore {

match /databases/{database}/documents {

match /Threats/{doc} {

allow read, write:

if request.auth != null;

}

}

}
```

Now only authenticated users may access data.

Step 6: Enable App Check

App Check prevents unauthorized applications from accessing Firebase services.

Navigate

Firebase

App Check

Enable

- Play Integrity API
- Device Check
- reCAPTCHA

Benefits

✔ Stops fake clients

✔ Prevents bot attacks

✔ Protects APIs

Step 7: Build an Incident Reporting System

Example fields

Attack Type

Source IP

Time

Country

Device

Severity

Status

Firestore Example

Incident001

Attack

SQL Injection

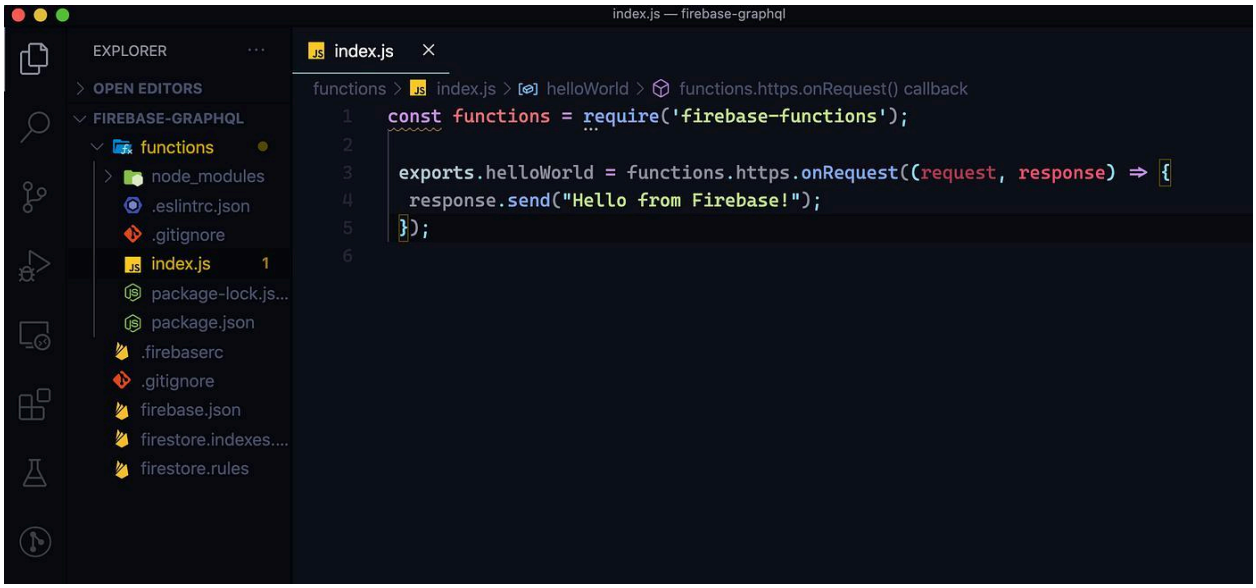
Severity

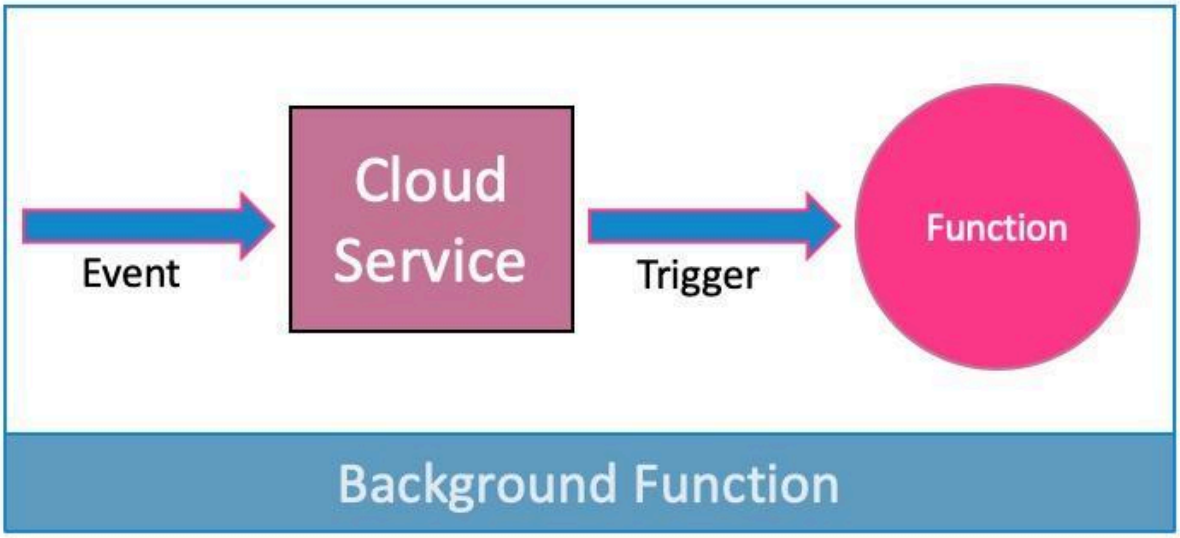
Critical

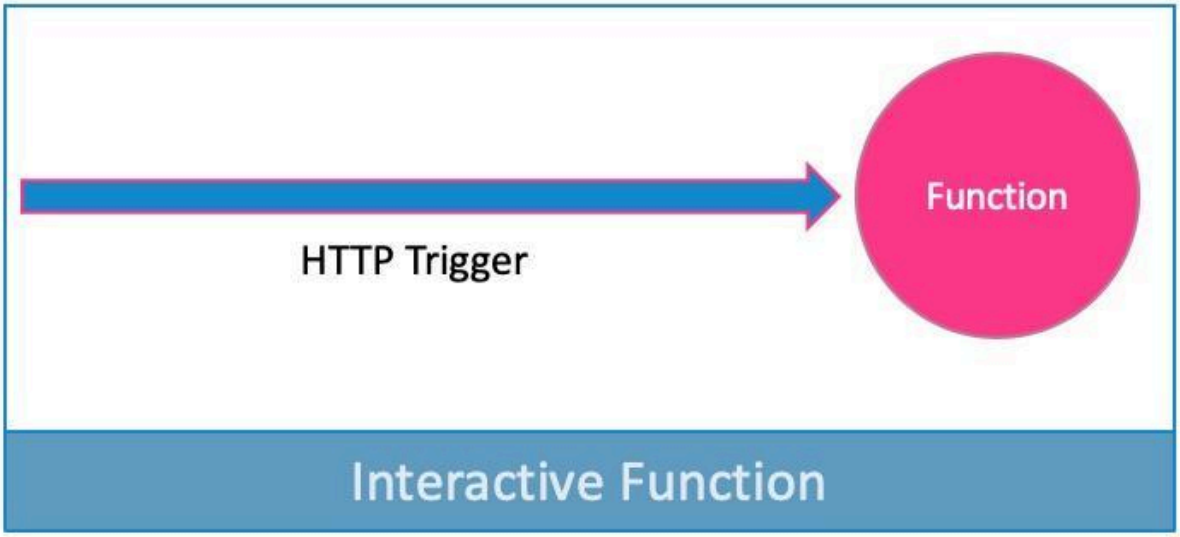
Status

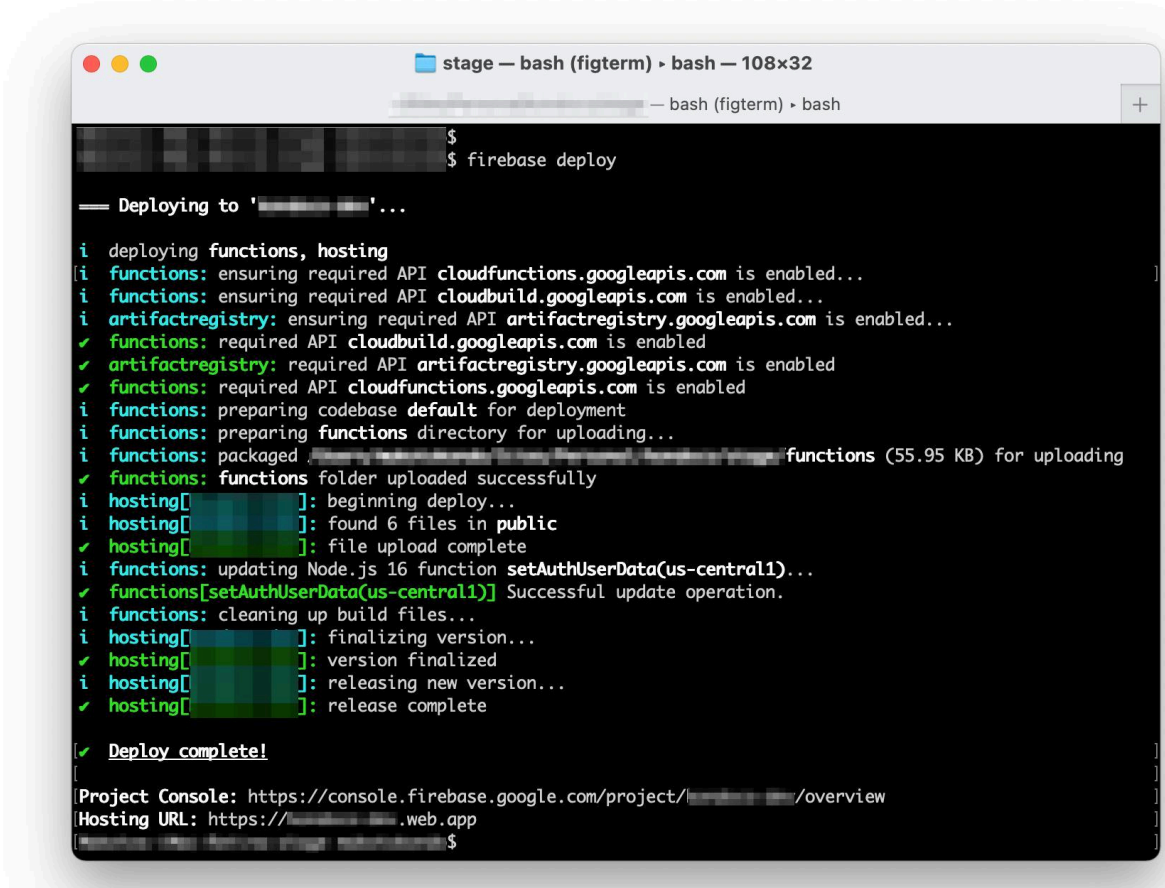
Open

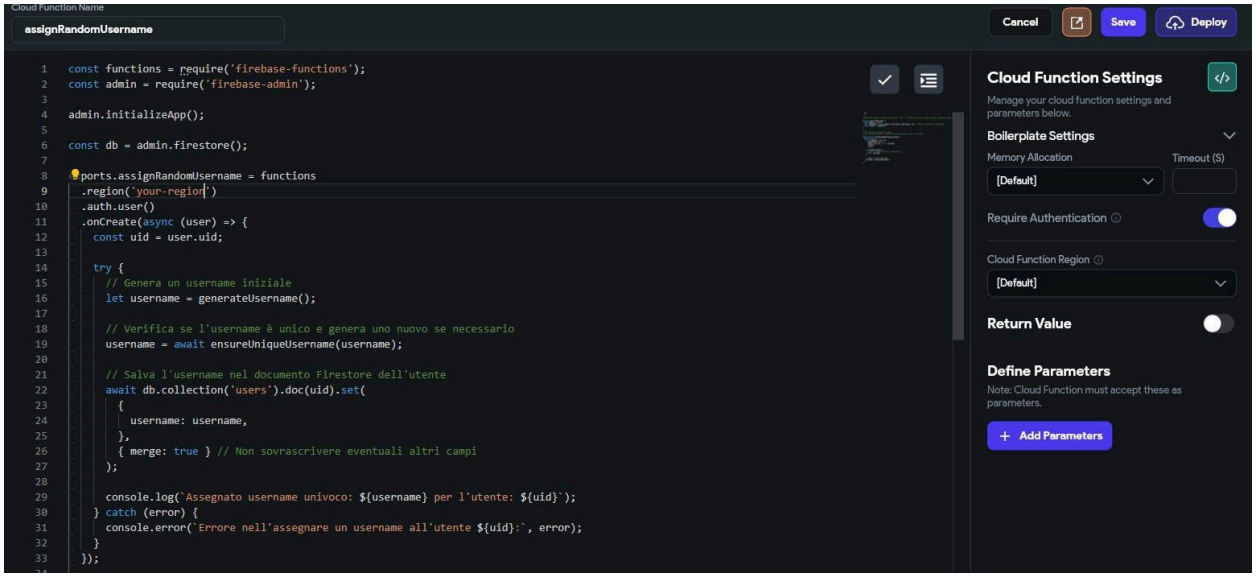
Step 8: Use Cloud Functions




Cloud Service
Event
Trigger
Function
Background Function


HTTP Trigger
Function
Interactive Function





Cloud Functions automatically respond to events.

Example

When a malware report is uploaded

↓

Send email

↓

Notify administrator

↓

Create audit log

↓

Update dashboard

Node.js Example

```
exports.alert =
functions.firestore

.document('Threats/{id}')

.onCreate((snap,context)=>{

console.log("Threat Detected");

});
```

Step 9: Store Malware Samples

Firebase Storage can securely store

- Suspicious PDFs
- Log Files
- Screenshots
- USB Images
- Memory Dumps

Never make storage public.

Example Storage Rule

allow read, write:

if request.auth != null;

Step 10: Send Real-Time Alerts

Cloud Messaging (FCM)

Example alerts

Unauthorized Login

Malware Uploaded

Firewall Disabled

USB Device Connected

Critical Alert

Users instantly receive notifications.

Step 11: Monitor User Activity

Store

Login Time

Logout

Failed Attempts

Location

IP Address

Device

This creates an audit trail.

Example

User

Steven

Login

08:13

Device

Android

Status

Success

Step 12: Build a Security Dashboard

Dashboard widgets

- Threat Count
- Active Users
- Failed Logins
- Malware Uploads
- Open Incidents
- Critical Alerts
- USB Events

Firestore updates the dashboard in real time.

Step 13: Detect Suspicious Behavior

Example algorithm

If

Failed Logins >5

within

2 minutes

↓

Lock Account

↓

Notify Administrator

↓

Log Incident

Cloud Functions can automate this process.

Step 14: Security Best Practices

Always

- Enable Authentication
- Use Firestore Rules
- Encrypt communications with HTTPS
- Enable App Check
- Use Multi-Factor Authentication
- Log every security event
- Validate user input
- Monitor failed logins
- Restrict administrator privileges

- Rotate API keys regularly
- Review Firebase security rules frequently
- Keep Firebase SDKs updated

Never

- Store passwords in Firestore
- Leave databases public
- Disable security rules
- Hard-code API secrets
- Trust client-side validation alone

Example Cybersecurity Project

AI USB Malware Detection System

Components

Android App

↓

Firebase Authentication

↓

Firestore

↓

Cloud Functions

↓

AI Detection Model

↓

Threat Score

↓

Security Dashboard

↓

Administrator Alert

Possible workflow

1. User inserts a USB device.
2. The app collects metadata (not the file contents unless authorized).
3. Metadata is securely sent to Firestore.
4. A Cloud Function invokes an AI model to assess risk.
5. If the risk score exceeds a threshold:
    - The event is logged.
    - Administrators receive a push notification.
    - The USB device can be quarantined by the application if supported by the operating environment.

Advantages of Firebase for Cybersecurity

Feature	Benefit
Authentication	Secure user access
Firestore	Real-time event logging
Storage	Secure evidence storage
Cloud Functions	Automated responses
App Check	Blocks unauthorized clients
Cloud Messaging	Instant security alerts
Analytics	User behavior insights
Scalability	Handles growing workloads

Limitations

- Firebase is not a replacement for a Security Information and Event Management (SIEM) platform.
- It should not be used as the sole malware detection engine.
- Security rules require careful configuration and testing to avoid accidental data exposure.
- Sensitive data may require additional encryption and compliance controls depending on regulations (e.g., HIPAA, GDPR, CJIS).

Conclusion

Firebase provides a powerful cloud backend for cybersecurity applications by combining secure authentication, real-time databases, automated cloud functions, protected file storage, and instant notifications. When configured with strong security rules, App Check, and least-privilege access controls, it can serve as the foundation for applications such as intrusion detection dashboards, incident response systems, digital forensics platforms, and AI-assisted threat monitoring—including projects like USB-based malware detection. Developers should treat Firebase as one component of a defense-in-depth strategy, complementing it with secure coding practices, encryption, continuous monitoring, and regular security assessments.