

Software Studio

軟體設計與實驗

Server & Firebase

Hung-Kuo Chu

Department of Computer Science
National Tsing Hua University

CS2410



Codeblock Conventions

HTML5 Program

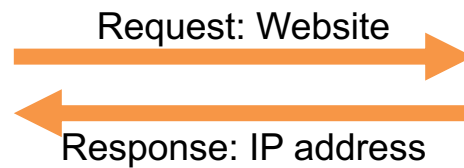
JavaScript Program

JSON or Rule

Basic Web Workflow



Client



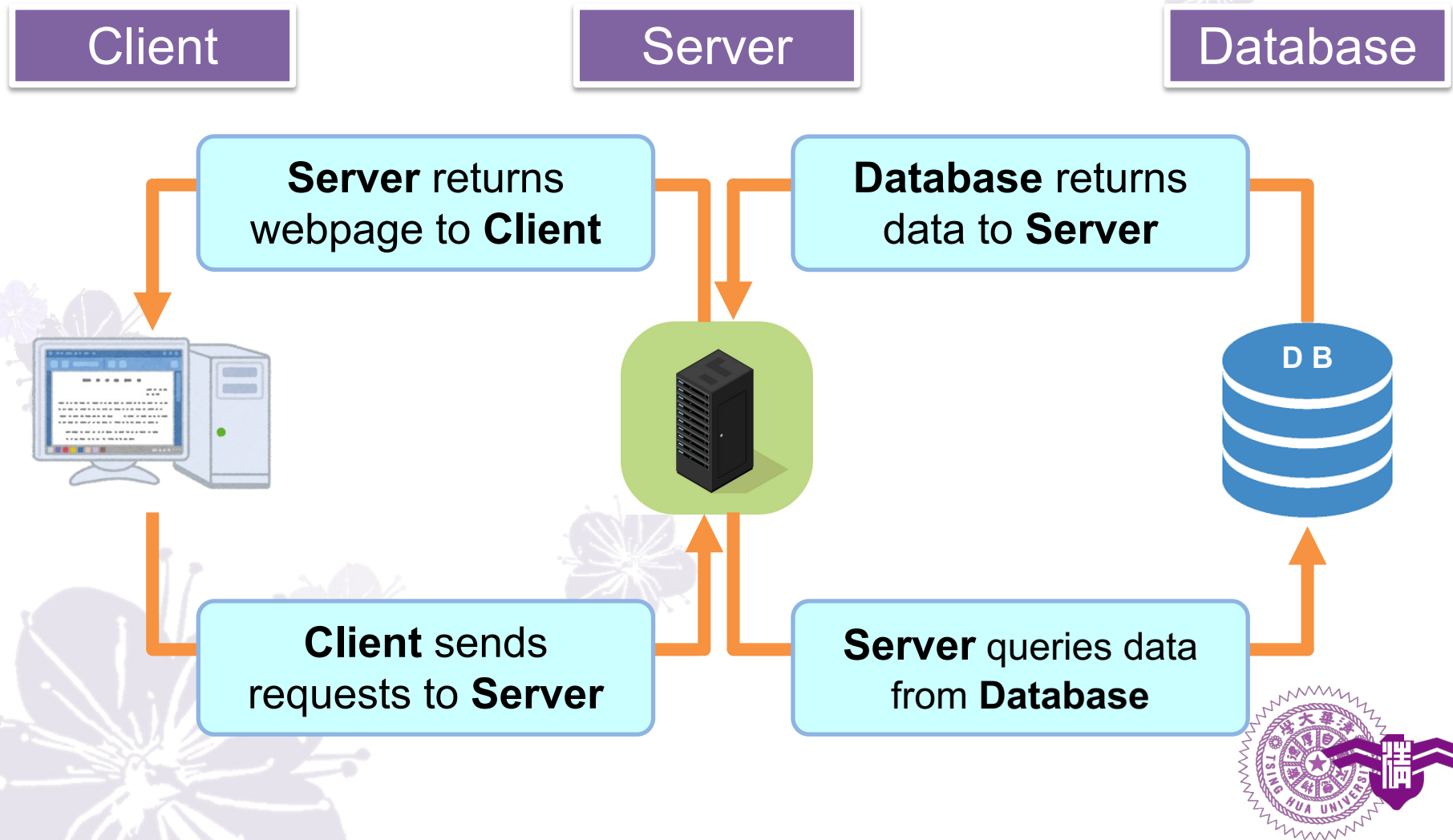
DNS Server



Main Server



Web Architecture



Client

- **Client (Frontend)**

- Send requests to Server.
- Process/Display data received from Server via Web Browser.
- HTML + CSS + Javascript can build a good interactive website
- We cannot keep any data
- We cannot collect data



Server

- **Server (backend)**
 - save data in server
 - do some private function
 - collect data and display to client
- **Database**
 - save some txt data
- **Storage**
 - save some multi media data



What is Firestore?

- Firestore is an application cloud development platform built by Google.
- Help developers **quickly set up** backend services in the cloud which effectively shorten the application development time.



Firestore



Why Use Firebase?

- A powerful **real-time database** which makes it an excellent candidate to drive **multi-player games**.
- Full document storage, analytics, hosting, etc.
- Using **JavaScript** API and provide user side high security.



Who are Using Firebase?

The New York Times



venmo

The Economist

trivago



wattpad





What Will You Learn?

- Firebase has a lot of features and we will teach you





Hosting
</>





Realtime Database
iOS  </> C++ 



Cloud Storage
iOS  </> C++ 



Authentication
iOS  </> C++ 



Cloud Functions
iOS  </> C++ 



Step by step

1. Create a new Firebase Project
2. Create a new Web App on Firebase Console
3. Activate Firebase Features
(Authentication, Database, Hosting...)
4. Install the Firebase CLI
5. Import Firebase SDK
6. Run Your Webpage



1. Create a new Firebase Project

- Go to [Firebase Console](#) (need a Google account) and add a project

× 建立專案(步驟 3 之 1)

請先輸入您的 專案名稱[®]

專案名稱

Software Studio

software-studio-a2c21

☒ 我接受 [Firebase 條款](#)

繼續

This id will be used in database URL and webpage domain.
Please record it!



1. Create a new Firebase Project

- Then finish remaining steps.

× 建立專案(步驟 3 之 2)

適用於 Firebase 專案的 Google Analytics (分析)

Google Analytics (分析) 是一項沒有用量限制的免費資料分析解決方案，能讓您在 Firebase Crashlytics、雲端通訊、應用程式內通訊、遠端設定、A/B 測試、預測和 Cloud Functions 等功能時，執行指定目標和查看報表等動作。

Google Analytics (分析) 提供以下功能：

- A/B 測試 ①
- 所有 Firebase 產品中的使用者區隔和指定目標功能 ②
- 預測使用者行為 ③
- 未受當機情況影響的使用者 ④
- 以事件為基礎的 Cloud Functions 觸發條件 ⑤
- 沒有用量限制的免費報表功能 ⑥

☒ 啟用這項專案的 Google Analytics (分析) 功能建議選項

上一步 繼續

× 建立專案(步驟 3 之 3)

設定 Google Analytics (分析)

數據分析位置 ①

美國

資料共用設定和 Google Analytics (分析) 條款

- ☒ 使用預設的 Google Analytics (分析) 資料共用設定。 [Learn more](#)
 - ✓ 將 Analytics (分析) 資料提供給 Google，協助我們改善產品和服務
 - ✓ 將 Analytics (分析) 資料提供給 Google，以便啟用基礎化功能
 - ✓ 提供 Analytics (分析) 資料給 Google 以取得技術支援
 - ✓ 將 Analytics (分析) 資料提供給 Google 帳戶專家
- ☒ 我接受《[評估服務控管者與控管者的資料保護條款](#)》，並確認本人必須遵守《[歐盟地區使用者同意授權政策](#)》。如想提供 Google Analytics (分析) 資料來改善 Google 產品和服務，您必須勾選這個核取方塊。 [瞭解詳情](#)
- ☒ 我接受 [Google Analytics \(分析\) 的條款](#)

建立專案時，系統會建立新的 Google Analytics (分析) 資源，並將該資源連結至您的 Firebase 專案。連結完成後，資料就能在產品之間流通，從 Google Analytics (分析) 資源匯出到 Firebase 的資料需符合 Firebase《[服務條款](#)》的規定；而匯入 Google Analytics (分析) 的 Firebase 資料則需符合 Google Analytics (分析)《[服務條款](#)》的規定。 [瞭解詳情](#)。

上一步 建立專案


2. Create a new Web App on Firebase Console



Initialization Configuration

× 將 Firebase 新增至您的網路應用程式

1 註冊應用程式

應用程式暱稱 

SoftwareStudioExample-1

— Pick a name for your application

☐ 一併為這個應用程式設定 **Firebase 代管功能**。瞭解詳情 

您也可以之後再設定代管功能。您隨時可以開始免費使用這項服務。

註冊應用程式

2 新增 Firebase SDK



Initialization Configuration

2 新增 Firebase SDK

☒ 使用 npm  ☐ 使用 <script> 標記 

如果你已使用 [npm](#) 和 [webpack](#) 或 [Rollup](#) 等模組整合工具，則可執行下列指令來安裝最新版 SDK：

```
$ npm install firebase
```



請初始化 Firebase，接著即可開始將 SDK 套用至要使用的產品。

```
// Import the functions you need from the SDKs you need
import { initializeApp } from "firebase/app";
// TODO: Add SDKs for Firebase products that you want to use
// https://firebase.google.com/docs/web/setup#available-libraries

// Your web app's Firebase configuration
const firebaseConfig = {
```

注意事項：這個選項會使用[模組 JavaScript SDK](#)，因此 SDK 的大小得以縮減。

如要進一步瞭解適用於網頁應用程式的 Firebase，請查看下列資源：[開始使用](#)、[Web SDK API 參考資料](#)、[使用範例](#)

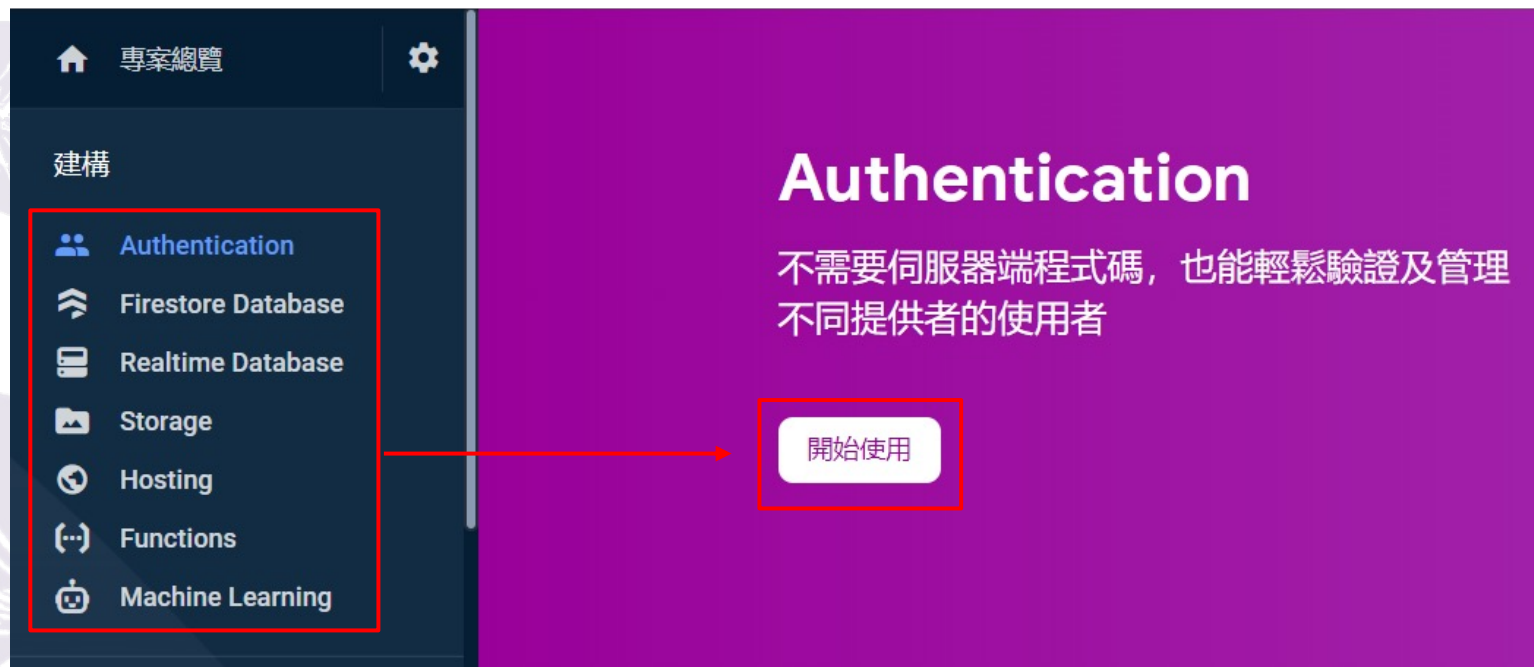
前往控制台

We will import Firebase SDK later



3. Activate Firebase Features

- Activate the firebase features you want to use one by one. (Authentication, Database, Hosting...)



3. Activate Firebase Features

Realtime Database

即時儲存及同步處理資料

建立資料庫

即時資料庫安全性規則

定義資料結構之後，您必須撰寫規則來保護資料。

[瞭解詳情](#)

- ☐ 以鎖定模式啟動
拒絕所有讀寫作業，保護資料庫的私密性
- ☒ 以測試模式啟動
允許對資料庫執行所有讀寫作業，加速完成設定程序。

```
{  
  "rules": {  
    ".read": true,  
    ".write": true  
  }  
}
```

！ 擁有您資料庫參考資料的所有使用者都能讀取或寫入您的資料庫

取消

啟用

3. Activate Firebase Features

Hosting

利用安全的全球內容傳遞聯播網，在彈指之間
部署網路版和行動版網頁應用程式

開始使用

1 安裝 Firebase CLI

如要透過 Firebase 託管功能來代管您的網站，則必須使用 Firebase CLI 這項指令列工具。

執行下列 [npm](#) 指令，藉此安裝 CLI 或更新至最新版 CLI。

```
$ npm install -g firebase-tools
```



無法順利執行操作嗎？您不妨查看 [Firebase CLI 參考資源](#) 或變更您的 [npm 權限](#)

- ☐ 一併顯示將 Firebase JavaScript SDK 新增至網頁應用程式的步驟
這項 SDK 含有 Cloud Firestore、Authentication、Performance Monitoring 等功能。您可以立即新增這項 SDK 或之後再新增。

下一步



4. Install the Firebase CLI

- Download [node.js](https://nodejs.org/en/download/), 12.16.2 LTS
- Run the command at your project directory and complete all settings
 1. `npm install -g firebase-tools`
 2. `firebase login`
 3. `firebase init`



Firebase Initialization

- Press <space> to select Firebase features.

```
? Are you ready to proceed? Yes
? Which Firebase features do you want to set up for this directory? Press Space to s
elect features, then Enter to confirm your choices. (Press <space> to select, <a> to
toggle all, <i> to invert selection, and <enter> to proceed)
(*) Realtime Database: Configure a security rules file for Realtime Database and (o
ptionally) provision default instance
  ( ) Firestore: Configure security rules and indexes files for Firestore
  ( ) Functions: Configure a Cloud Functions directory and its files
>(*) Hosting: Configure files for Firebase Hosting and (optionally) set up GitHub Ac
tion deploys
  ( ) Hosting: Set up GitHub Action deploys
  ( ) Storage: Configure a security rules file for Cloud Storage
  ( ) Emulators: Set up local emulators for Firebase products
(Move up and down to reveal more choices)
```



Firebase Initialization

```
? Please select an option:  
> Use an existing project  
Create a new project  
Add Firebase to an existing Google Cloud Platform project  
Don't set up a default project
```

- On default Firebase project selection, select your project
 - If you could not see the project you just create on firebase's console, you can directly add after initialization, so just select [don't setup a default project]
 - Run “**firebase projects:list**” and select one **ID**
 - Use “**firebase use [ID]**” to link your project



Firestore Initialization

- Use default setting in other selections
- Notice that if your application wants to use any firestore features, it need to be activated first. Otherwise, you can't set it in ***firebase init***.



Firebase Initialization

```
Which Firebase CLI features do you want to setup for this folder? Press Space to select features, then Enter to confirm your choices. Database: Deploy Firebase Realtime Database Rules, Functions: Configure and deploy Cloud Functions,
Hosting: Configure and deploy Firebase Hosting sites, Storage: Deploy Cloud Storage security rules

=== Project Setup

First, let's associate this project directory with a Firebase project.
You can create multiple project aliases by running firebase use --add,
but for now we'll just set up a default project.

? Select a default Firebase project for this directory: [don't setup a default project]

=== Database Setup

Firebase Realtime Database Rules allow you to define how your data should be
structured and when your data can be read from and written to.

? What file should be used for Database Rules? database.rules.json
+ Database Rules for have been downloaded to database.rules.json.
Future modifications to database.rules.json will update Database Rules when you run
firebase deploy.

=== Functions Setup

A functions directory will be created in your project with a Node.js
package pre-configured. Functions can be deployed with firebase deploy.

? What language would you like to use to write Cloud Functions? JavaScript
? Do you want to use ESLint to catch probable bugs and enforce style? No
+ Wrote functions/package.json
+ Wrote functions/index.js
+ Wrote functions/.gitignore
? Do you want to install dependencies with npm now? Yes

> grpc@1.19.0 install C:\Users\Eric\Desktop\Firebase Testing\functions\node_modules\grpc
> node-pre-gyp install --fallback-to-build --library=static_library

node-pre-gyp WARN Using needle for node-pre-gyp https download
[grpc] Success: "C:\Users\Eric\Desktop\Firebase Testing\functions\node_modules\grpc\src\node\extension_binary\node-v64-win32-x64-unknown\grpc_node.node" is installed via remote

> protobufjs@6.8.8 postinstall C:\Users\Eric\Desktop\Firebase Testing\functions\node_modules\protobufjs
> node scripts/postinstall

> firebase-functions@2.2.1 postinstall C:\Users\Eric\Desktop\Firebase Testing\functions\node_modules\firebase-functions
> node ./upgrade-warning

===== WARNING! =====

This upgrade of firebase-functions contains breaking changes if you are upgrading from a version below v1.0.0.
To see a complete list of these breaking changes, please go to:

https://firebase.google.com/docs/functions/beta-v1-diff

npm notice created a lockfile as package-lock.json. You should commit this file.
added 310 packages from 222 contributors and audited 1031 packages in 7.971s
found 0 vulnerabilities

=== Hosting Setup

Your public directory is the folder (relative to your project directory) that
will contain Hosting assets to be uploaded with firebase deploy. If you
have a build process for your assets, use your build's output directory.

? What do you want to use as your public directory? public
? Configure as a single-page app (rewrite all urls to /index.html)? No
+ Wrote public/404.html
+ Wrote public/index.html

=== Storage Setup

Firebase Storage Security Rules allow you to define how and when to allow
uploads and downloads. You can keep these rules in your project directory
and publish them with firebase deploy.

? What file should be used for Storage Rules? storage.rules
+ Writing configuration info to firebase.json...
+ Writing project information to .firebaserc...
+ Writing gitignore file to .gitignore...

+ Firebase initialization complete!
```


5. Import Firebase SDK

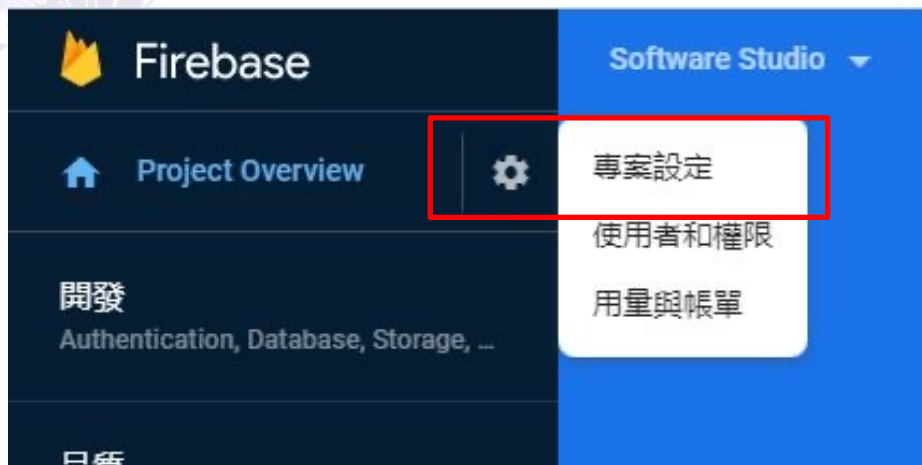
- Add Firebase SDK to your .html
 - firebase: Firebase core
 - firebase-app: The core Firebase client
 - firebase-auth: Firebase Authentication
 - firebase-database: Firebase Realtime Database
 - firebase-storage: Cloud Storage

```
<!-- Firebase App is always required and must be first -->  
<script src="https://www.gstatic.com/firebasejs/7.8.2/firebase-app.js"></script>  
<script src="https://www.gstatic.com/firebasejs/7.8.2/firebase-auth.js"></script>  
<script src="https://www.gstatic.com/firebasejs/7.8.2/firebase-database.js"></script>  
<script src="https://www.gstatic.com/firebasejs/7.8.2/firebase-firestore.js"></script>  
<script src="https://www.gstatic.com/firebasejs/7.8.2/firebase-functions.js"></script>
```



5. Import Firebase SDK

- Go to Settings to get the firebaseConfig.



SDK 設定和配置

☒ npm [?] ☐ CDN [?] ☐ 設定 [?]

如果你已使用 [npm](#) 和 [webpack](#) 或 [Rollup](#) 等模組整合工具，則可執行下列指令來安裝最新版 SDK：

```
$ npm install firebase
```

請初始化 Firebase，接著即可開始將 SDK 套用至要使用的產品。

```
// Import the functions you need from the SDKs you need
import { initializeApp } from "firebase/app";
// TODO: Add SDKs for Firebase products that you want to use
// https://firebase.google.com/docs/web/setup#available-libraries

// Your web app's Firebase configuration
const firebaseConfig = {
```



5. Import Firebase SDK

- Add Firebase Config to your config.js
- **Make sure there is databaseURL.**

```
<script src="config.js"></script>
```

```
// Initialize Firebase
// TODO: Replace with your project's customized code snippet
const firebaseConfig = {
  apiKey: "<API_KEY>",
  authDomain: "<PROJECT_ID>.firebaseapp.com",
  databaseURL: "https://<DATABASE_NAME>.firebaseio.com",
  projectId: "<PROJECT_ID>",
  storageBucket: "< PROJECT_ID >.appspot.com",
  messagingSenderId: "<SENDER_ID>",
  appId: "<APPLICATION_ID>",
  measurementId: "<AnalyticsSDK>"
};
const app = initializeApp(firebaseConfig);
```

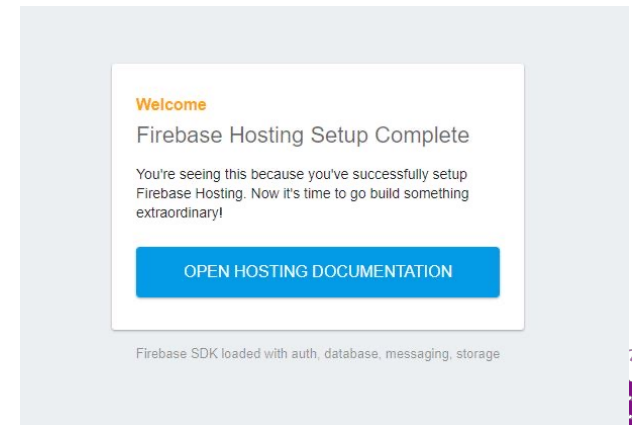
5. Import Firebase SDK

- If databaseURL is missing. You can find it in firebase console.



6. Run Your Webpage

- You can run a local server for testing
- Put all your web files(.html, .js, images...etc.) under the public folder
- Run the command at your project directory
— **firebase serve**
- Your webpage will run on <http://localhost:5000>



**Take a
Break!**



Hosting

- Firebase Hosting has some key capabilities
 - Served over a secure connection
 - Fast content delivery
 - Rapid deployment
 - One-click rollbacks



Hosting

- Run the command at your project directory
 - `firebase deploy`
- Your app will be deployed to the domain <YOUR-FIREBASE-APP>.firebaseapp.com



Authentication

- You can use Firebase Authentication to allow users to sign in your app using one or more sign-in methods

新增第一種登入方式，即可開始使用 Firebase 驗證


原生供應商

✉ 電子郵件/密碼

☎ 電信業者

👤 匿名

其他供應商


 Google


 Game Center

 Microsoft

 Facebook

 Apple

 Twitter

 Play 遊戲

 GitHub

 Yahoo

Authentication

1. Add your web page domain to Authorized Domains. (You can skip this step if you are using firebase hosting.)
2. Enable the sign-in method
3. Implement authentication with `firebase.auth()`



Authorized Domains

- Before adding Firebase Authorized to your web page, you need to add your web page domain in firebase console.
- For example, your gitlab website url.

Authentication

Users

Sign-in method

Templates

Usage

授權網域 ⓘ

新增網域

已授權網域

類型

localhost

Default

softwarestudio-32b32.firebaseio.com

Default

softwarestudio-32b32.web.app

Default

Email/Password Sign-In

- Enable the Email/password sign-in method

The screenshot shows the 'Authentication' settings page. The 'Sign-in method' tab is selected and highlighted with a red box. A red arrow points from this tab to the '新增供應商' (Add Provider) button, which is also highlighted with a red box. Another red arrow points from the '新增供應商' button to the '電子郵件/密碼' (Email/Password) sign-in method, which is highlighted with a red box. The '電子郵件/密碼' method is shown with a toggle switch set to '啟用' (Enabled). Below this, there is a description of the method and a link to '瞭解詳情' (Learn more). At the bottom, there is a section for '電子郵件連結 (不需要密碼即可登入)' (Email link (no password required for login)) with a toggle switch set to '啟用' (Enabled).

Authentication

Users **Sign-in method** Templates Usage

登入供應商

新增供應商

識別資訊提供者

- 電子郵件/密碼
- Google

電子郵件/密碼

可讓使用者以自己的電子郵件地址和密碼註冊。我們的 SDK 也提供電子郵件地址驗證、密碼救援和電子郵件地址變更等基本功能。[瞭解詳情](#)

電子郵件連結 (不需要密碼即可登入)

Email/Password Sign-In

- Sign up new users

```
firebase.auth().createUserWithEmailAndPassword(email, password)
  .then((userCredential) => {
    // Signed in
    var user = userCredential.user;
    // ...
  })
  .catch((error) => {
    var errorCode = error.code;
    var errorMessage = error.message;
    // ...
  });
```



Email/Password Sign-In

- Sign in existing users

```
firebase.auth().signInWithEmailAndPassword(email, password)
  .then((userCredential) => {
    // Signed in
    var user = userCredential.user;
    // ...
  })
  .catch((error) => {
    var errorCode = error.code;
    var errorMessage = error.message;
    // ...
  });
```



Google Sign-In

- Enable the Google sign-in method



The screenshot shows the 'Authentication' settings page with the 'Sign-in method' tab selected. A red box highlights the 'Sign-in method' tab, and a red arrow points from it to the '新增供應商' (Add Provider) button. Another red box highlights the '新增供應商' button, and a red arrow points from it to the '啟用' (Enable) toggle switch for the Google sign-in method.

Authentication

Users **Sign-in method** Templates Usage

登入供應商

識別資訊提供者	狀態
✉ 電子郵件/密碼	✓ 已啟用
 Google	<input checked="" type="checkbox"/> 啟用

系統會自動為已連結的 Apple 和網頁應用程式設定 Google 登入機制。如要為 Android 應用程式設定 Google 登入機制，你必須在[專案設定](#)中為每個應用程式新增 [SHA1 指紋](#)。

Google Sign-In

- Create an instance of the Google provider object

```
var provider = new firebase.auth.GoogleAuthProvider();
```



Google Sign-In

- Sign in with a pop-up window

```
firebase.auth().signInWithPopup(provider).then(function(result) {  
  // This gives you a Google Access Token. You can use it to access the Google API.  
  var token = result.credential.accessToken;  
  // The signed-in user info.  
  var user = result.user;  
  // ...  
}).catch(function(error) {  
  // Handle Errors here.  
  var errorCode = error.code;  
  var errorMessage = error.message;  
  // The email of the user's account used.  
  var email = error.email;  
  // The firebase.auth.AuthCredential type that was used.  
  var credential = error.credential;  
});
```



Google Sign-In

- Sign in by redirecting to the sign-in page

```
firebase.auth().signInWithRedirect(provider);
firebase.auth().getRedirectResult().then(function(result) {
  if (result.credential) {
    // This gives you a Google Access Token. You can use it to access the Google API.
    var token = result.credential.accessToken;
    // ...
  }
  // The signed-in user info.
  var user = result.user;
}).catch(function(error) {
  // Handle Errors here.
  var errorCode = error.code;
  var errorMessage = error.message;
  // The email of the user's account used.
  var email = error.email;
  // The firebase.auth.AuthCredential type that was used.
  var credential = error.credential;
});
```

Facebook Sign-In

- On the [Facebook for Developers](#) site, create a new App



The screenshot displays the Facebook for Developers website. On the left, there is a photograph of a group of people in a meeting. The main content area features the 'Facebook for Startups' section with the text '參加 Facebook 新創公司計畫 助您打造優質的產品並拓展您 創公司規模。' and a '深入瞭解' button. On the right, a navigation menu is open, showing options like '要求', '開發人員設定', '公司設定', and '登出 Facebook'. The '建立應用程式' (Create New App) button is highlighted with a red rectangle. At the bottom, there are three news items: 'Introducing Early Testing of Data Use Checkup', 'Winners of our first Facebook Online Hackathon announced', and 'Pausing Individual Verification', each with a '瞭解詳情' link.

facebook for developers

產品 文件 更多 ▾ 我的應用程式 ▾

Facebook for Startups

參加 Facebook 新創公司計畫
助您打造優質的產品並拓展您
創公司規模。

深入瞭解

建立應用程式

要求
開發人員設定
公司設定
登出 Facebook

2020 年 F8 大會重要更新：鑑於 2019 年新冠病毒 (COVID-19) 疫情持續擴大，我們決定取消今年 F8 大會的現場實體活動。更多內容

Introducing Early Testing of Data Use Checkup
瞭解詳情

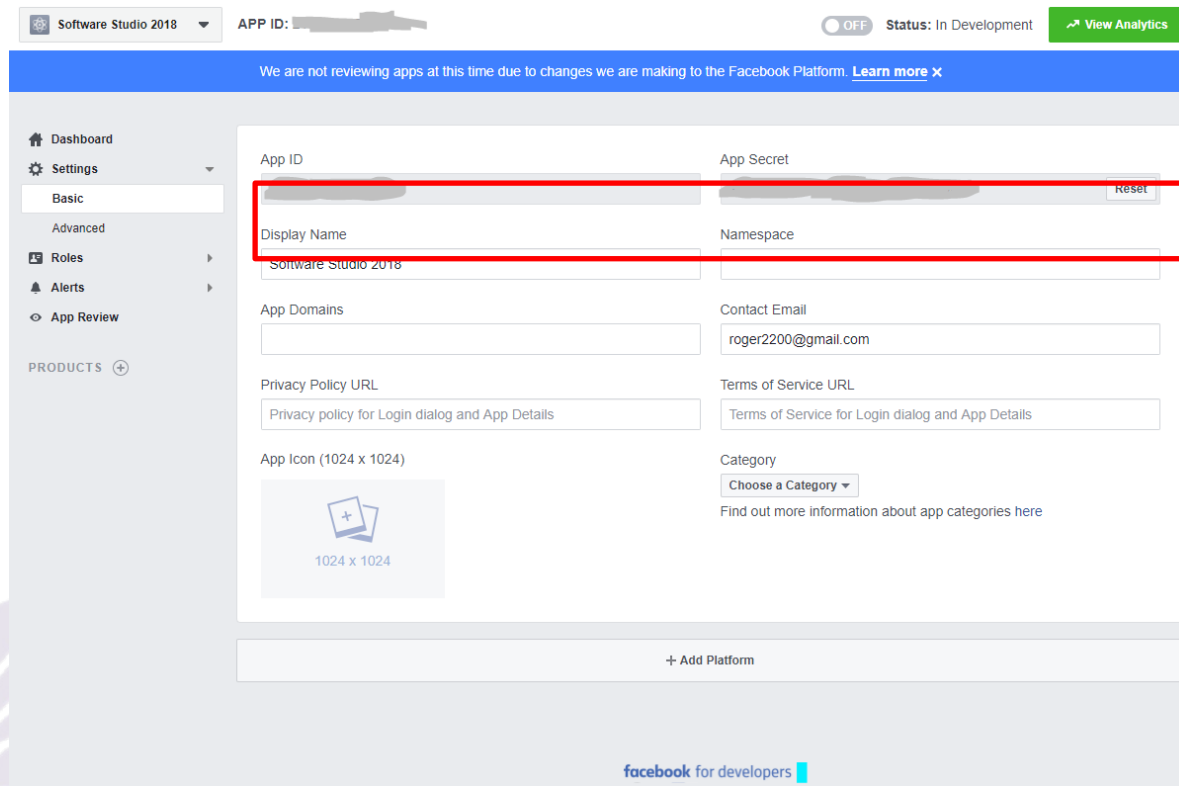
Winners of our first Facebook Online Hackathon announced
瞭解詳情

Pausing Individual Verification
瞭解詳情



Facebook Sign-In

- Get the **App ID** and an **App Secret** for your app



The screenshot displays the Facebook Developer console interface. At the top, a blue banner states: "We are not reviewing apps at this time due to changes we are making to the Facebook Platform. [Learn more](#) ×". Below this, the left sidebar contains navigation links: Dashboard, Settings (selected), Basic, Advanced, Roles, Alerts, and App Review. The main content area shows the settings for the app "Software Studio 2018". A red rectangular box highlights the "App ID" and "App Secret" fields, which are both masked with grey bars. Other visible fields include "Display Name" (Software Studio 2018), "Namespace", "App Domains", "Contact Email" (roger2200@gmail.com), "Privacy Policy URL", "Terms of Service URL", "App Icon (1024 x 1024)", and "Category" (Choose a Category). At the bottom of the console, there is a "+ Add Platform" button and the "facebook for developers" logo.

(in Facebook for Developer)



Facebook Sign-In

- Enable the Facebook sign-in method

Authentication

USERS SIGN-IN METHOD TEMPLATES USAGE

Sign-in providers

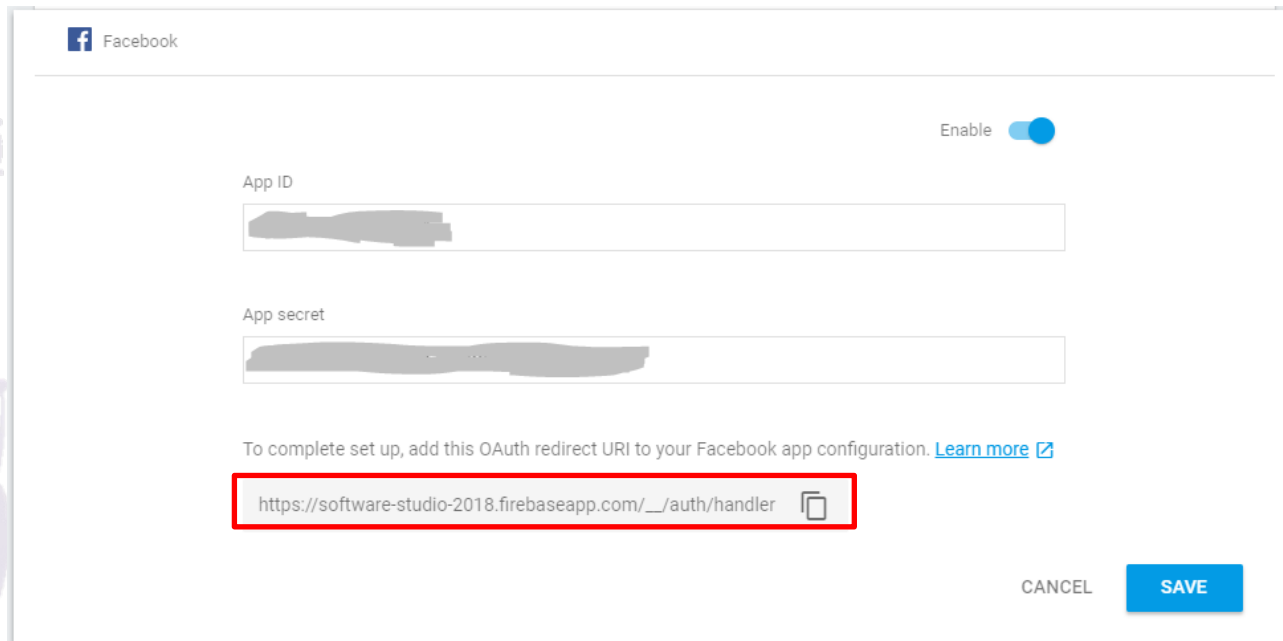
Provider	Status
Email/Password	Enabled
Phone	Disabled
Google	Enabled
Play Games	Disabled
Facebook	Enabled
Twitter	Disabled
GitHub	Disabled
Anonymous	Disabled

(in firebase page)



Facebook Sign-In

- Add OAuth redirect URL (find in firebase console) in **Product Settings > Facebook Login** on the [Facebook for Developers](#)



Facebook

Enable ☒

App ID

App secret

To complete set up, add this OAuth redirect URI to your Facebook app configuration. [Learn more](#)

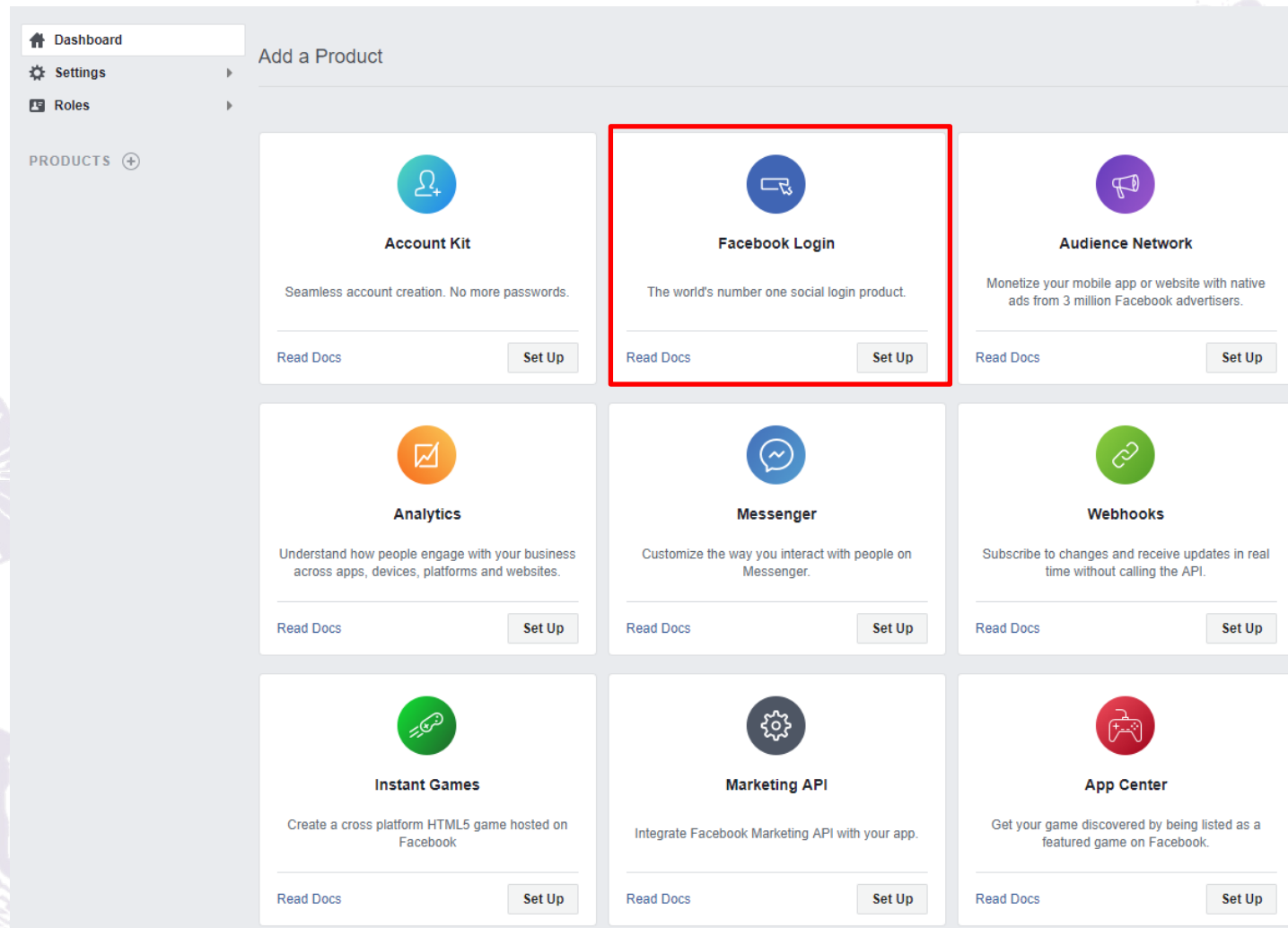
`https://software-studio-2018.firebaseio.com/_/auth/handler`

CANCEL SAVE

(in firebase page)



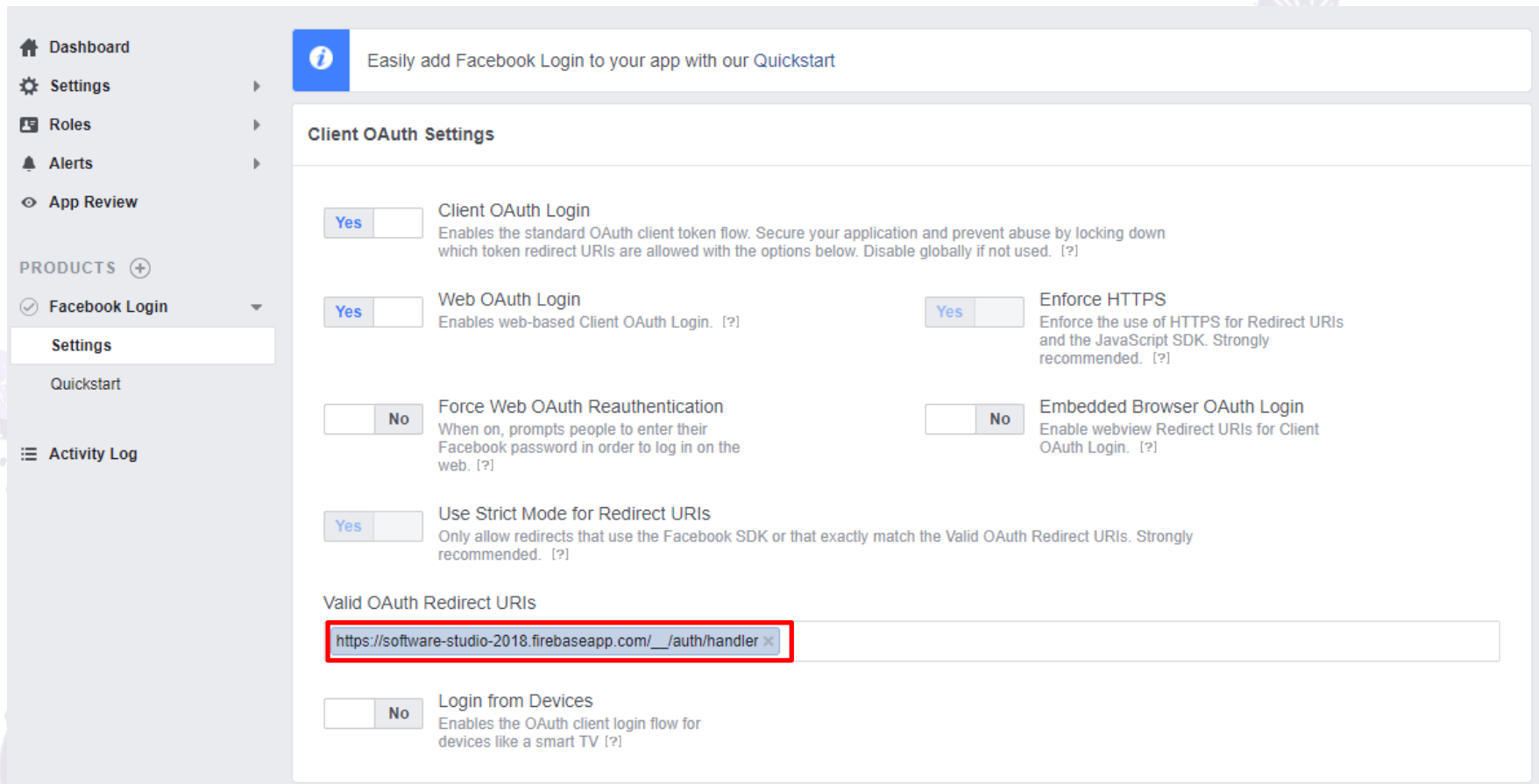
Facebook Sign-In



(in Facebook for Developer)



Facebook Sign-In



Dashboard

Settings

Roles

Alerts

App Review

PRODUCTS (+)

Facebook Login

Settings

Quickstart

Activity Log

Client OAuth Settings

Client OAuth Login
☒ Yes
Enables the standard OAuth client token flow. Secure your application and prevent abuse by locking down which token redirect URIs are allowed with the options below. Disable globally if not used. [?]

Web OAuth Login
☒ Yes
Enables web-based Client OAuth Login. [?]

Enforce HTTPS
☒ Yes
Enforce the use of HTTPS for Redirect URIs and the JavaScript SDK. Strongly recommended. [?]

Force Web OAuth Reauthentication
☐ No
When on, prompts people to enter their Facebook password in order to log in on the web. [?]

Embedded Browser OAuth Login
☐ No
Enable webview Redirect URIs for Client OAuth Login. [?]

Use Strict Mode for Redirect URIs
☒ Yes
Only allow redirects that use the Facebook SDK or that exactly match the Valid OAuth Redirect URIs. Strongly recommended. [?]

Valid OAuth Redirect URIs

Login from Devices
☐ No
Enables the OAuth client login flow for devices like a smart TV [?]

(in Facebook for Developer)



Facebook Sign-In

- Create an instance of the Facebook provider object

```
var provider = new firebase.auth.FacebookAuthProvider();
```



Facebook Sign-In

- Sign in with popup

```
firebase.auth().signInWithPopup(provider).then(function(result) {  
    // This gives you a Facebook Access Token. You can use it to access the Facebook API.  
    var token = result.credential.accessToken;  
    // The signed-in user info.  
    var user = result.user;  
}).catch(function(error) {  
    // Handle Errors here.  
    var errorCode = error.code;  
    var errorMessage = error.message;  
    // The email of the user's account used.  
    var email = error.email;  
    // The firebase.auth.AuthCredential type that was used.  
    var credential = error.credential;  
});
```



Facebook Sign-In

- Sign in with redirect

```
firebase.auth().signInWithRedirect(provider);
```

- The **getRedirectResult()** function is shared with google



Manage Users

- You can get the current user is by setting an observer on the Auth object

```
firebase.auth().onAuthStateChanged(function(user) {  
  if (user) {  
    // User is signed in.  
    var uid = user.uid;  
  } else {  
    // No user is signed in.  
  }  
});
```



Manage Users

- You can also get the currently signed-in user by using the currentUser property

```
var user = firebase.auth().currentUser;  
  
if (user) {  
    // User is signed in.  
} else {  
    // No user is signed in.  
}
```



Sign-Out

- To sign-out a user, call `signOut()`

```
firebase.auth().signOut().then(() => {  
  // Sign-out successful.  
}).catch((error) => {  
  // An error happened.  
});
```



Reference

- <https://firebase.google.com/docs/auth/web/start>
- <https://firebase.google.com/docs/reference/js/v8/firebase.auth.Auth>



Realtime Database

- All Firebase Realtime Database data is stored as JSON objects
- Unlike a SQL database, there are no tables or records

```
{  
  "users": {  
    "alovelace": {  
      "name": "Ada Lovelace",  
      "contacts": { "ghopper": true },  
    },  
    "ghopper": { ... },  
    "eclarke": { ... }  
  }  
}
```

database

Read and Write Data

- Get a database reference

```
// Get a reference to the database service  
var database = firebase.database();
```



Write Data

- There are 4 ways to write data to the database
 - **set()**: Write or overwrite specify reference data
 - **push()**: Adding list data, every call to push() will generate a unique ID
 - **update()**: Give a key, and update the data of this key will not cover the entire data
 - **transaction()**: Updating complex data while updating will cause errors



Write Data

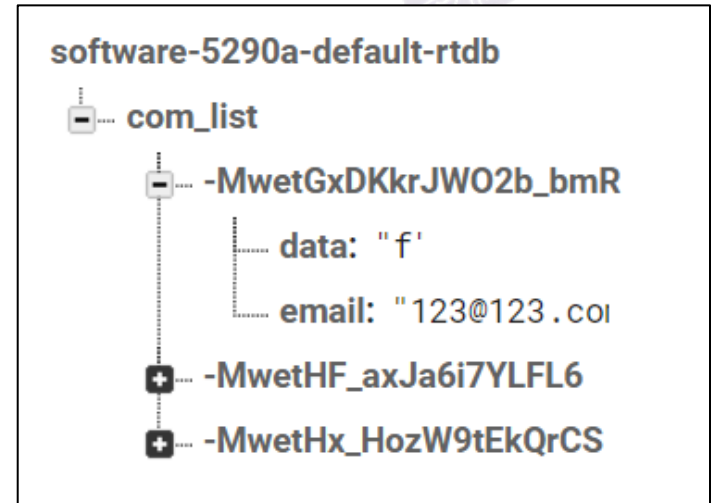
- Use **ref()** to reference the specific path.
- Use **set()** to write data and replace any existing data at that path.

```
function writeUserData(userId, name, email, imageUrl) {  
  firebase.database().ref('users/' + userId).set({  
    username: name,  
    email: email,  
    profile_picture : imageUrl  
  });  
}
```



Write Data

- Use **push()** to push data.
- Every **push()** generate a unique key.
- Use **.key** to access it.



```
function writeUserData(userId, name, email, imageUrl) {  
  var key = firebase.database().ref('com_list')  
    .push({data: data, email: email})  
    .key;  
}
```

Read Data

- To read data at a path and listen for changes, use the **on()** or **once()** methods of reference to observe events or use query
 - On: listen for change
 - Once: for retrieve one time, not listen



Read Data

- The first parameter of **on()** and **once()** is [EventType](#). There are 5 EventTypes.
- Use '**value**' to read the value of that path.
- Use '**child_added**' to read every added child.

```
var starCountRef = firebase.database().ref('posts/' + postId + '/starCount');
starCountRef.on('value', function(snapshot) {
  updateStarCount(postElement, snapshot.val());
});
```

```
var userId = firebase.auth().currentUser.uid;
firebase.database().ref('/users/' + userId).once('value').then(function(snapshot){
  var username = (snapshot.val() && snapshot.val().username) || 'Anonymous';
});
```

Read Data

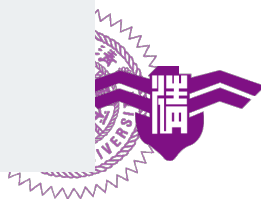
- **on()** and **once()** receive data as a DataSnapshot object.
- Use **val()** to get the content of the snapshot after you received data.

```
var starCountRef = firebase.database().ref('posts/' + postId + '/starCount');
starCountRef.on('value', function(snapshot) {
  updateStarCount(postElement, snapshot.val());
});
```

```
var userId = firebase.auth().currentUser.uid;
firebase.database().ref('/users/' + userId).once('value').then(function(snapshot){
  var username = (snapshot.val() && snapshot.val().username) || 'Anonymous';
});
```

Database Rules

- Firebase Realtime Database Rules determine who has read and write access to your database, how your data is structured, and what indexes exist



Database Rules

- Rule Types
 - **.read**
 - **.write**
 - **.validate**: Defines what a correctly formatted value will look like, whether it has child attributes, and the data type
 - **.indexOn**: Specifies a child to index to support ordering and querying



Database Rules

- For example, here's a set of security rules that allows anyone to read the path /foo/, but no one to write to it

```
{  
  "rules": {  
    "foo": {  
      ".read": true,  
      ".write": false  
    }  
  }  
}
```



Database Rules

- The Firebase Database Rules include [built-in variables](#) and functions

```
{  
  "rules": {  
    "users": {  
      "$uid": {  
        ".write": "$uid === auth.uid"  
      }  
    }  
  }  
}
```

rule

```
{  
  "users": {  
    "James": { // id 1  
      "name": "",  
      ""  
    },  
    "John": { // id 2  
      "name": "",  
      ""  
    }  
  }  
}
```

Database data

Database Rules

- The rules language includes a **.validate** rule which allows you to apply validation logic using the same expressions

```
{  
  "rules": {  
    "foo": {  
      ".validate": "newData.isString() && newData.val().length < 100"  
    }  
  }  
}
```



Database Rules

- When you changed database.rules.json locally, you must use **firebase deploy** to update the rule. Otherwise, your changes won't apply.
- Or you can change the rules directly on firebase console.



Database Rules

- More information and usage about database:
- <https://firebase.google.com/docs/reference/security/database>
- <https://firebase.google.com/docs/database/web/start#web-version-8>
- <https://firebase.google.com/docs/database/web/read-and-write>



Cloud Storage

- Cloud Storage for Firebase lets you upload and share user generated content, such as images and video, which allows you to build rich media content into your apps



Initialization

Storage

不需要伺服器端程式碼，也能儲存及擷取使用者產生的圖片、音訊與影片等檔案

開始使用

設定 Cloud Storage

1 Cloud Storage 安全性規則 ——— 2 設定 Cloud Storage 位置

根據預設，您的規則允許通過驗證的使用者執行所有讀寫作業。

定義資料結構之後，您需要撰寫規則來保護資料。[瞭解詳情](#)

```
service firebase.storage {
  match /b/{bucket}/o {
    match /{allPaths=**} {
      allow read, write: if request.auth != null;
    }
  }
}
```

取消

下一步

設定 Cloud Storage

✓ Cloud Storage 安全性規則 ——— 2 設定 Cloud Storage 位置

您在位置設定中指定的位置即為 Cloud Storage 預設值區的所在位置，值區資料都會儲存在該處。

⚠ 位置一經設定即無法變更。此位置設定會成為 Cloud Firestore 的預設位置。

[瞭解詳情](#)

Cloud Storage 位置

nam5 (us-central)

Blaze 方案客戶可為額外的值區選擇其他位置

取消

完成

Storage Reference

- Similar to Database, Cloud Storage need a reference to upload/download files

```
// Get a reference to the storage service, which is used to create references in  
your storage bucket
```

```
var storage = firebase.storage();
```

```
// Create a storage reference from our storage service
```

```
var storageRef = storage.ref();
```

```
// Create a child reference, imagesRef now points to 'images'
```

```
var imagesRef = storageRef.child('images');
```

```
// Child references can also take paths delimited by '/'
```

```
// spaceRef now points to "images/space.jpg"
```

```
var spaceRef = storageRef.child('images/space.jpg');
```

Upload File

- Once you've created an appropriate reference, you then up call the put() method to up;

```
// Create a root reference
var storageRef = firebase.storage().ref();

// Create a reference to 'mountains.jpg'
var mountainsRef = storageRef.child('mountains.jpg');

var file = ... // use the Blob or File API to get the file
mountainsRef.put(file).then(function(snapshot) {
  console.log('Uploaded a blob or file!');
});

//now the file's reference in database is '/mountain.jpg'
```

Upload File (Cont'd)

```
// Create a root reference
var storageRef = firebase.storage().ref();

// Create a reference to 'images/mountains.jpg'
var mountainImagesRef = storageRef.child('images/mountains.jpg');

var file = ... // use the Blob or File API to get the file
mountainImagesRef.put(file).then(function(snapshot) {
  console.log('Uploaded a blob or file!');
});
//now the file's reference is '/images/mountains.jpg'
```



Download File

- Use **getDownloadURL()** to get file URL, your can download file directly by this URL or inserted into a HTML element



Download File

- Directly download the file

```
storageRef.child('images/stars.jpg').getDownloadURL().then(function(url) {  
    // `url` is the download URL for 'images/stars.jpg'  
  
    // Download the file directly  
    var xhr = new XMLHttpRequest();  
    xhr.responseType = 'blob';  
    xhr.onload = function(event) {  
        var blob = xhr.response;  
    };  
    xhr.open('GET', url);  
    xhr.send();  
  
}).catch(function(error) {  
    // Handle any errors  
});
```

Download File (Cont'd)

- Insert file into html element

```
storageRef.child('images/stars.jpg').getDownloadURL().then(function(url) {  
    // `url` is the download URL for 'images/stars.jpg'  
  
    // inserted into an <img> element:  
    var img = document.getElementById('myimg');  
    img.src = url;  
}).catch(function(error) {  
    // Handle any errors  
});
```



Cloud Storage Rules

- Storage Security Rules manage the complexity for you by allowing you to specify path-based permissions

```
// Rules can optionally specify a condition  
allow write: if <condition>;
```

Storage Rules

```
service firebase.storage {  
  // The {bucket} wildcard indicates we match files in all Cloud Storage buckets  
  match /b/{bucket}/o {  
    // Match filename  
    match /filename {  
      allow read: if <condition>;  
      allow write: if <condition>;  
    }  
  }  
}
```

Storage Rules

Cloud Storage Rules

- You can also specify the file size

// Rules can specify conditions that consider the request

Storage Rules

```
service firebase.storage {  
  match /b/{bucket}/o {  
    match /images/{imageId} {  
      // Only allow uploads of any image file that's less than 5MB  
      allow write: if request.resource.size < 5 * 1024 * 1024  
        && request.resource.contentType.matches('image/.*');  
    }  
  }  
}
```



Cloud Storage Rules

- More information and usage about cloud storage rules:
- <https://firebase.google.com/docs/storage/security>

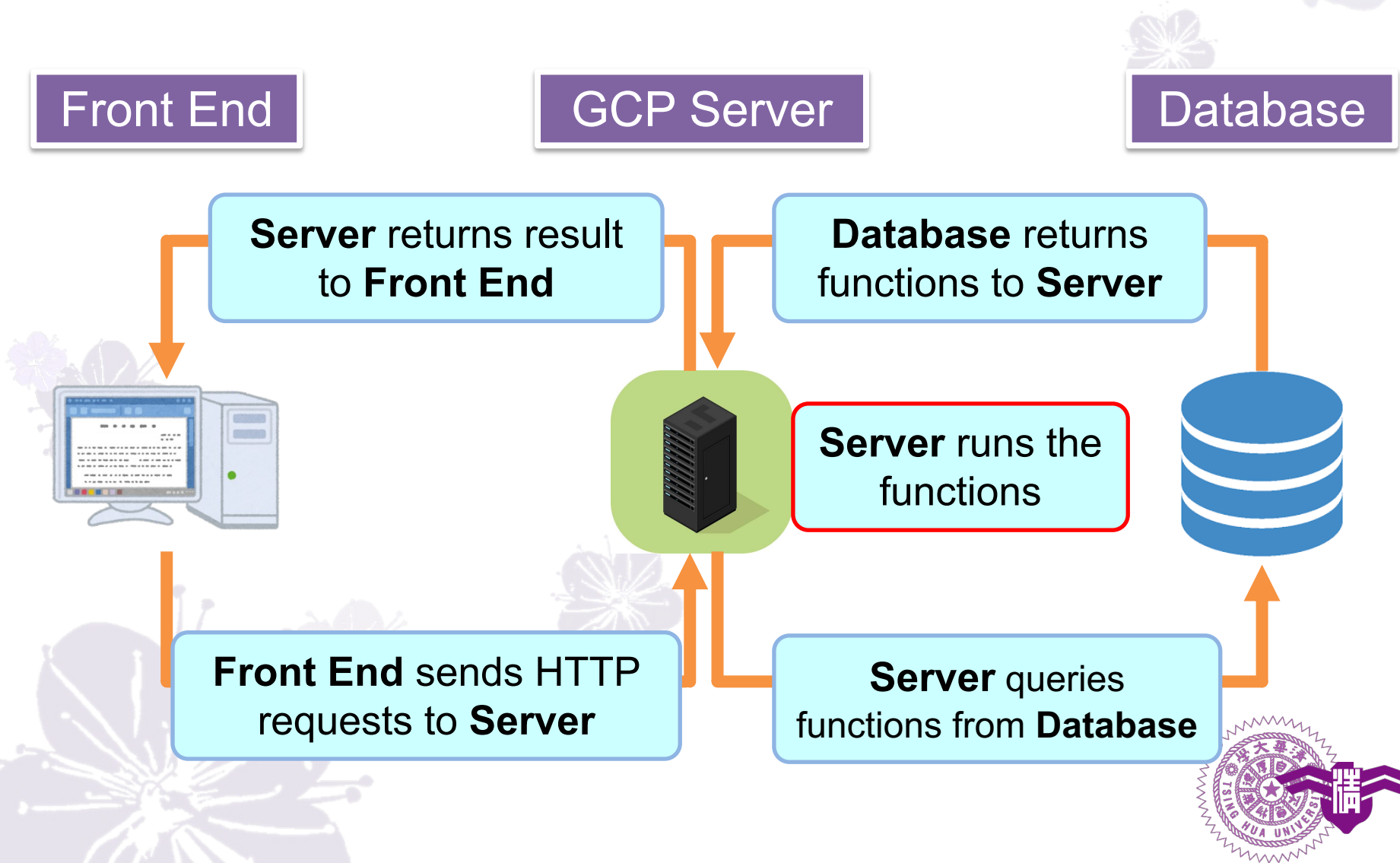


Cloud Function

- Cloud Functions let you automatically run backend code in response to events triggered by Firebase features and HTTPS requests.
- Your code is stored in Google's cloud and runs in a managed environment. There's no need to manage and scale your own servers.



Cloud Function



Cloud Function

- First, enable cloud functions to your project.

Command : `firebase init functions`

```
? Are you ready to proceed? Yes

=== Project Setup

First, let's associate this project directory with a Firebase project.
You can create multiple project aliases by running firebase use --add,
but for now we'll just set up a default project.

? Please select an option: Use an existing project
? Select a default Firebase project for this directory: software-studio-6698b (Software Studio)
i Using project software-studio-6698b (Software Studio)

=== Functions Setup

A functions directory will be created in your project with a Node.js
package pre-configured. Functions can be deployed with firebase deploy.

? What language would you like to use to write Cloud Functions? JavaScript
? Do you want to use ESLint to catch probable bugs and enforce style? No
+ Wrote functions/package.json
+ Wrote functions/index.js
+ Wrote functions/.gitignore
? Do you want to install dependencies with npm now? Yes

> protobufjs@6.8.9 postinstall D:\大三嗨起來\助教人生\軟體實驗\Lecture Program\Firebase\answer\functions\node_modules\protobufjs
> node scripts/postinstall

npm notice created a lockfile as package-lock.json. You should commit this file.
added 258 packages from 281 contributors and audited 930 packages in 73.817s

30 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities

i Writing configuration info to firebase.json...
i Writing project information to .firebaserc...
i Writing gitignore file to .gitignore...

+ Firebase initialization complete!
```



Cloud Function

- Then you will get the following folder in your project.
- Now, we are going to edit **index.js** to create our cloud functions.

```
└─ functions
   └─ node_modules
   └─ .gitignore
   └─ JS index.js
   └─ {} package-lock.json
   └─ {} package.json
```



Get Firebase admin

- Just like what we did **firebase.initializeApp(config)** in our .js files. We initialize firebase again in index.js.
- After that we can use the firebase functions, such as database or storage.

```
const admin = require('firebase-admin');  
admin.initializeApp();
```

index.js



Create a New Function

- This function is an HTTP endpoint. Any request to the endpoint results in ExpressJS-style Request and Response objects passed to the `onRequest()` callback. <https://expressjs.com/zh-tw/>

```
exports.addMessage = functions.https.onRequest(async(req, res) => {  
  // Grab the text parameter.  
  const original = req.query.text;  
  // Push the new message into the Realtime Database using the Firebase Admin SDK.  
  const snapshot = await admin.database().ref('/messages').push({ original: original });  
  // Redirect with 303 SEE OTHER to the URL of the pushed object in the Firebase console.  
  res.redirect(303, snapshot.ref.toString());  
});
```

Serve code

```
// This client-end function is equivalent to the previous cloud function  
function addMessageLocal(text){  
  const snapchat = await admin.database().ref('/messages').push(text);  
}
```

Client code

Setup Local Testing Server

- Once you finish editing your cloud functions, you can start up a local server to test the correctness of your functions.

Command : `firebase emulators:start --only functions`

```
$ firebase emulators:start --only functions
i  emulators: Starting emulators: functions
!  Your requested "node" version "8" doesn't match your global version "12"
+  functions: Emulator started at http://localhost:5001
i  functions: Watching "C:\Users\penguin\Desktop\firebase\functions" for Cloud Functions...
+  functions[addMessage]: http function initialized (http://localhost:5001/software-studio-6698b/us-central1/addMessage).
+  All emulators started, it is now safe to connect.
```



Use Cloud Functions (local)

- After initialize firebase in your js code, you can get the functions you created.
- Remember that if you are using **local server** to call cloud functions, you have to **USE** `firebase.functions().useFunctionsEmulator()` to redirect to localhost server.

```
// connect to localhost server
```

```
firebase.functions().useFunctionsEmulator('http://localhost:5001');
```

```
//comment this line if you use formal cloud functions
```

```
// call function : addMessage
```

```
var addMessage = firebase.functions().httpsCallable('addMessage');  
addMessage({text:uppercasemetoo}).then(()=>{});
```

Client code

Deploy Cloud Functions

- Deploy cloud function to server require upgrading firebase service to **Blaze**

8

將功能部署到生產環境

一旦您的功能在仿真器中可以正常工作，您就可以繼續在生產環境中部署，測試和運行它們。請記住，要部署到建議的 Node.js 12 運行時環境，您的項目必須處於 Blaze 即用即付 [計費計劃](#) 中。請參閱 [雲功能定價](#)。

Firebase 計費方案

Spark

免費 每月 \$0 美元

資料庫、Firestore、儲存、函式、電話驗證、代管和 Test Lab 有使用量配額

✕ 你的專案將能使用 Google Cloud 功能

✓ 所有方案皆包含數據分析、通知、當機報告和支援等服務

[查看完整方案詳情](#)

目前的方案

Blaze

用多少，付多少

包括免費用量，每天會計算一次。超出免費用量上限後，您只需針對專案的使用量付費。

你的專案將能使用 Google Cloud 功能

✓ 所有方案皆包含數據分析、通知、當機報告和支援等服務

[查看完整方案詳情](#)

選取方案



Deploy Cloud Functions

- Deploy function to server make sure we don't need to run server by ourselves

Command : `firebase deploy --only functions`

```
$ firebase deploy --only functions

=== Deploying to 'software-studio-6698b'...

i deploying functions
i functions: ensuring necessary APIs are enabled...
+ functions: all necessary APIs are enabled
i functions: preparing functions directory for uploading...
i functions: packaged functions (26.79 KB) for uploading
+ functions: functions folder uploaded successfully
i functions: creating Node.js 8 function addMessage(us-central1)...
+ functions[addMessage(us-central1)]: Successful create operation.
Function URL (addMessage): https://us-central1-software-studio-6698b.cloudfunctions.net/addMessage
+ Deploy complete!

Project Console: https://console.firebase.google.com/project/software-studio-6698b/overview
```



Use Cloud Functions

- After deploy firebase cloud functions, you can use the functions you created.

```
// call function : addMessage
```

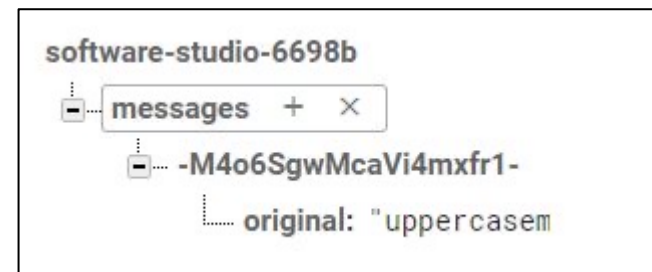
```
var addMessage = firebase.functions().httpsCallable('addMessage');  
addMessage({text:uppercasemetoo}).then(()=>{});
```

Client code



Demo

- Enter the following url in your browser.
- [https://us-central1-\[MY_PROJECT\].cloudfunctions.net/addMessage?text=uppercasemetoo](https://us-central1-[MY_PROJECT].cloudfunctions.net/addMessage?text=uppercasemetoo)
- Then go to firebase console and check your real-time database. If cloud functions setup successfully, you will have following structure.



Reference

- [Firestore Documentation](#)
- [Firestore Get Started – Web](#)
- [Firestore on the Web - Tutorials](#)



thank
you!

Question

