

The background features a dark blue-to-purple gradient with faint, light-colored concentric circles and degree markings (40, 150, 160, 170, 180, 190, 200, 210, 220, 230, 240, 250, 260) on the left side, suggesting a technical or design theme.

AJAX 程式設計

AJAX PROGRAMMING

DESIGN

張家瑋 博士

助理教授

國立臺中科技大學資訊工程系



GOOGLE FIREBASE
GITHUB PAGES

G
O
O
G
L
E

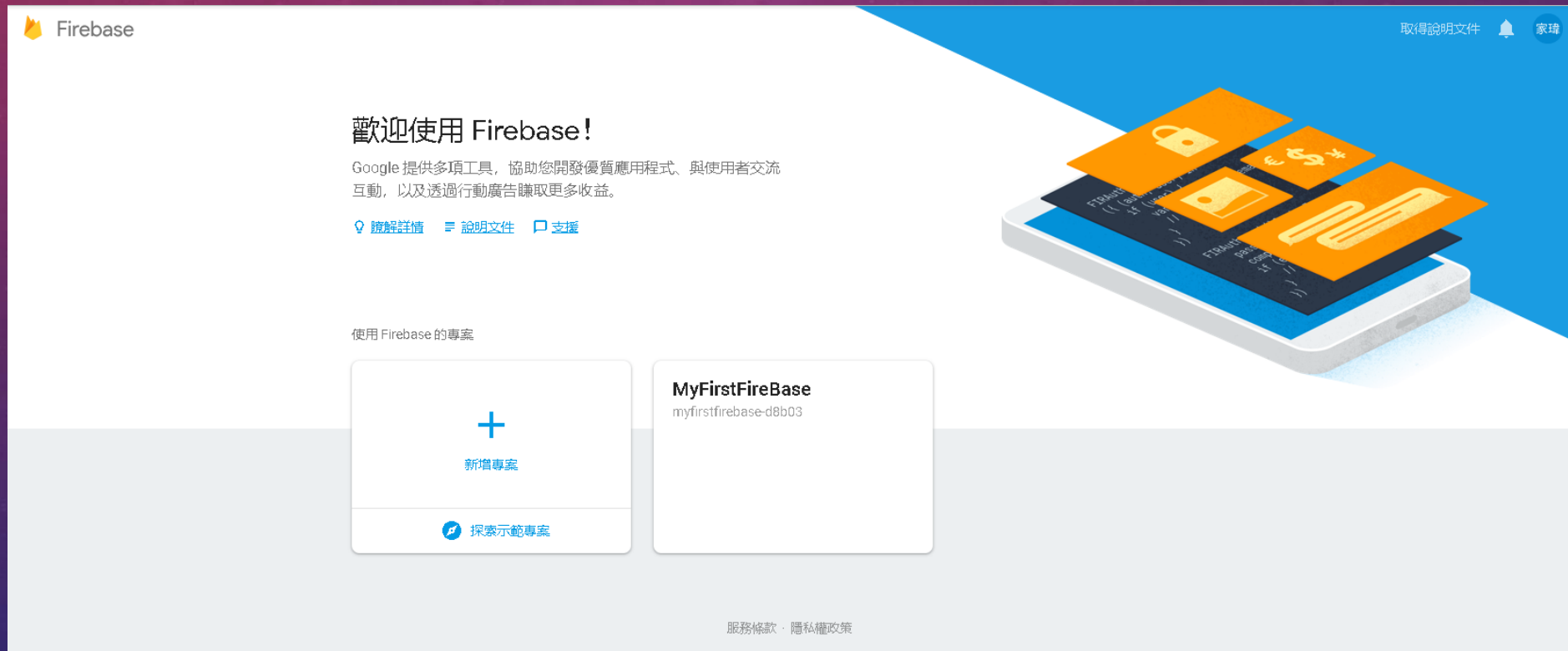
GITHUB

FIREBASE

- 帳號認證
- 資料庫
- 資料儲存
- 架站
- Functions

FIREBASE-DATABASE

- <https://console.firebase.google.com/>



建立專案

新增專案

專案名稱

test

 +  + 

提示：專案可連結不同平台的多個應用程式 [?](#)

專案 ID [?](#)

test-1e3a7 

數據分析位置 [?](#)

台灣

Cloud Firestore 位置 [?](#)

us-central

☒ 使用預設的 Google Analytics for Firebase 資料分享設定

☒ 提供 Analytics (分析) 資料給 Google，協助我們改善產品和服務

☒ 提供 Analytics (分析) 資料給 Google 以取得技術支援

☒ 提供 Analytics (分析) 資料給 Google 以啟用基準化功能

☒ 提供 Analytics (分析) 資料給 Google 帳戶專家

☒ 我接受 [控管者對控管者的相關條款](#)。如要提供數據分析資料以改善 Google 產品和服務，您必須勾選這個核取方塊。 [瞭解詳情](#)

☒ 我同意在我的應用程式中使用 Firebase 服務，且接受的適用 [條款](#)。

取消 建立專案

建立資料庫

test ▾ 取得說明文件 家瑋

開發

- Authentication
- Database**
- Storage
- Hosting
- Functions
- ML Kit

品質
Crashlytics, Performance, Test Lab

數據分析
Dashboard, Events, Conversions, A...

拓展
Predictions, A/B Testing, Cloud Mes...

Spark
免費 每月 \$0 美元 升級

test ▾ 取得說明文件 家瑋

測試版

Cloud Firestore

新一代 Realtime Database 擁有更強大的查詢和自動調整資源配置功能

建立資料庫

瞭解詳情

- 瞭解「Cloud Firestore」是否適合您
比較資料庫
- 如何開始使用?
查看相關文件
- 使用「Cloud Firestore」需要多少費用?
查看計費方式
- Cloud Firestore 有哪些功能?

Introducing Cloud Firestore

稍後觀看 分享

建立資料庫

Cloud Firestore 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。

```
service cloud.firestore {  
  match /databases/{database}/documents {  
    match /{document=**} {  
      allow read, write;  
    }  
  }  
}
```

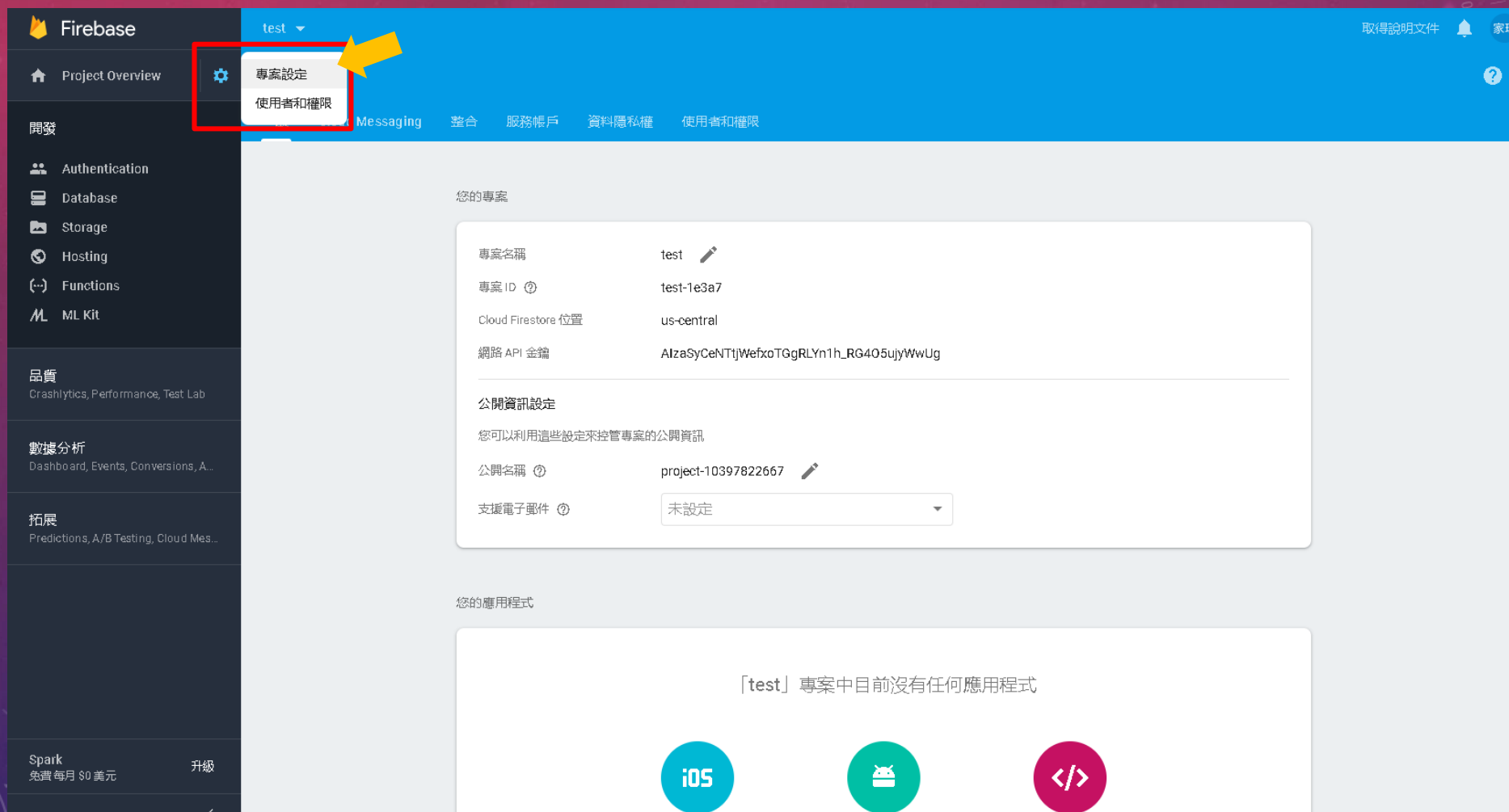
 擁有您資料庫參考資料的所有使用者都能讀取或寫入您的資料庫

啟用 Cloud Firestore 測試版後，您將無法在這個專案中使用 Cloud Datastore (特別是透過相關的 App Engine 應用程式)。

取消

啟用

取得WEB接口



The screenshot displays the Firebase console interface. On the left sidebar, the 'Project Overview' section is active, and the 'Settings' gear icon is highlighted with a red box. A yellow arrow points to the 'Project settings' dropdown menu. The main content area shows the 'Project Overview' page for the project named 'test'. The page includes a table of project details and a section for public information settings.

Project Name	test
Project ID	test-1e3a7
Cloud Firestore location	us-central
Network API key	AIzaSyCeNT1jWefxoTGgRLyn1h_RG4O5ujyWwUg

Public Information Settings

You can use these settings to manage the public information of your project.

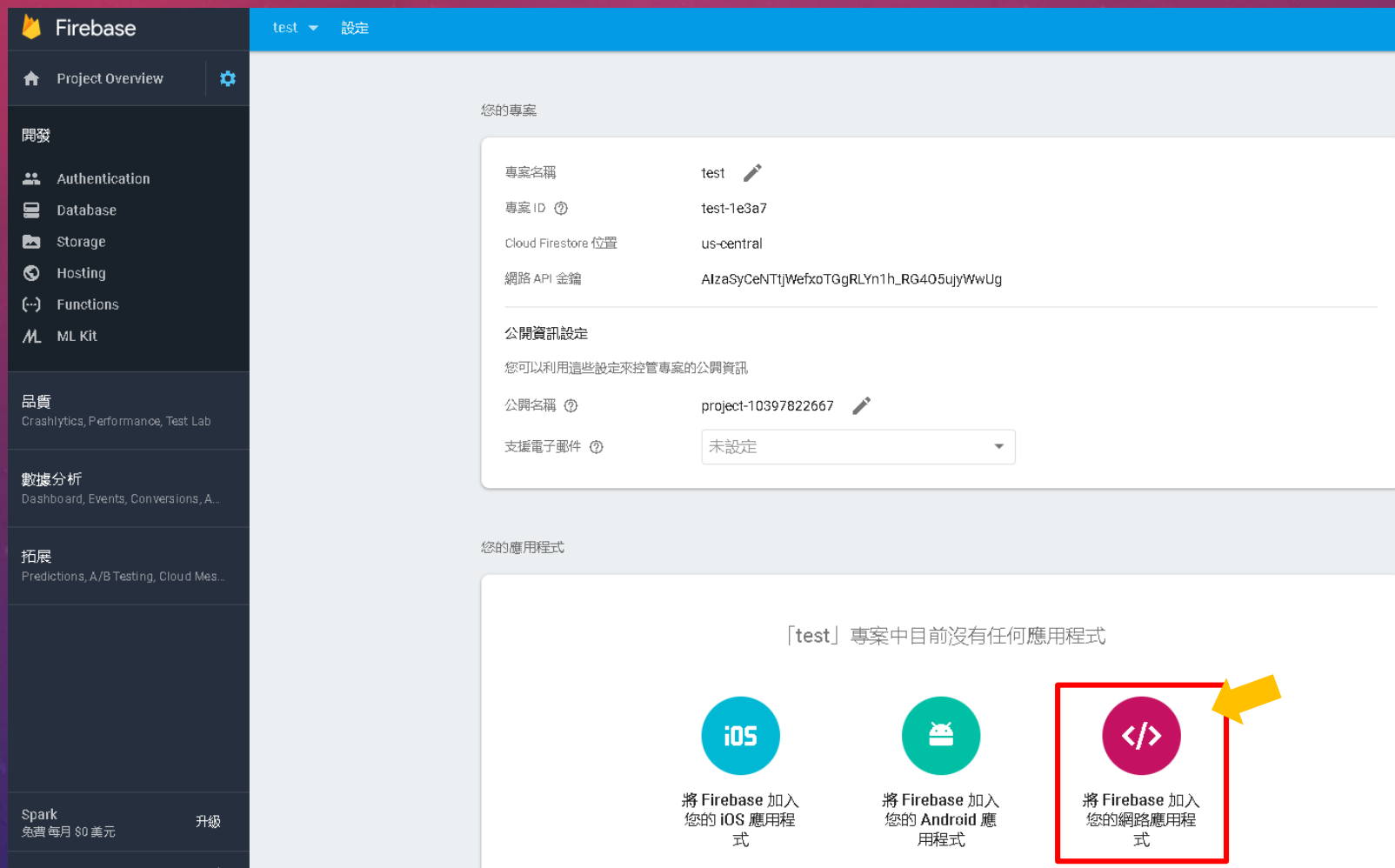
Public Name	project-10397822667
Support Email	未設定

Your Applications

「test」專案中目前沒有任何應用程式

iOS Android Web

取得WEB接口



The screenshot displays the Firebase console interface. On the left is a dark sidebar with the 'Firebase' logo and a menu including 'Project Overview', '開發' (Development) with sub-items like Authentication, Database, Storage, Hosting, Functions, and ML Kit, '品質' (Quality), '數據分析' (Data Analytics), '拓展' (Extensions), and 'Spark'. The main content area has a blue header with 'test' and '設定' (Settings). Below this, the '您的專案' (Your Project) section shows details for the 'test' project, including its ID, location, and API key. The '公開資訊設定' (Public Information Settings) section allows for setting the public name and support email. The '您的應用程式' (Your Applications) section shows a message that no applications are currently registered for the 'test' project. At the bottom, there are three buttons: 'iOS', 'Android', and a web icon (represented by a code symbol </>). The web icon button is highlighted with a red rectangular border and a yellow arrow pointing to it from the right.

Firebase

test ▼ 設定

您的專案

專案名稱 test

專案 ID test-1e3a7

Cloud Firestore 位置 us-central

網路 API 金鑰 AIzaSyCeNTIjWefxoTGgRLYn1h_RG4O5ujyWwUg

公開資訊設定

您可以利用這些設定來控管專案的公開資訊

公開名稱 project-10397822667

支援電子郵件 未設定

您的應用程式

[test] 專案中目前沒有任何應用程式

iOS

將 Firebase 加入您的 iOS 應用程式

Android

將 Firebase 加入您的 Android 應用程式

</>

將 Firebase 加入您的網路應用程式

取得WEB接口

將 Firebase 加入您的網路應用程式



複製下方的程式碼片段，然後貼到 HTML 底端 (其他 script 標記前)。

```
<script src="https://www.gstatic.com/firebasejs/5.7.0/firebase.js"></script>
<script>
  // Initialize Firebase
  var config = {
    apiKey: "[REDACTED]",
    authDomain: "[REDACTED].app.com",
    databaseURL: "https://[REDACTED].firebaseio.com",
    projectId: "[REDACTED]",
    storageBucket: "[REDACTED].spot.com",
    messagingSenderId: "[REDACTED]"
  };
  firebase.initializeApp(config);
</script>
```

複製

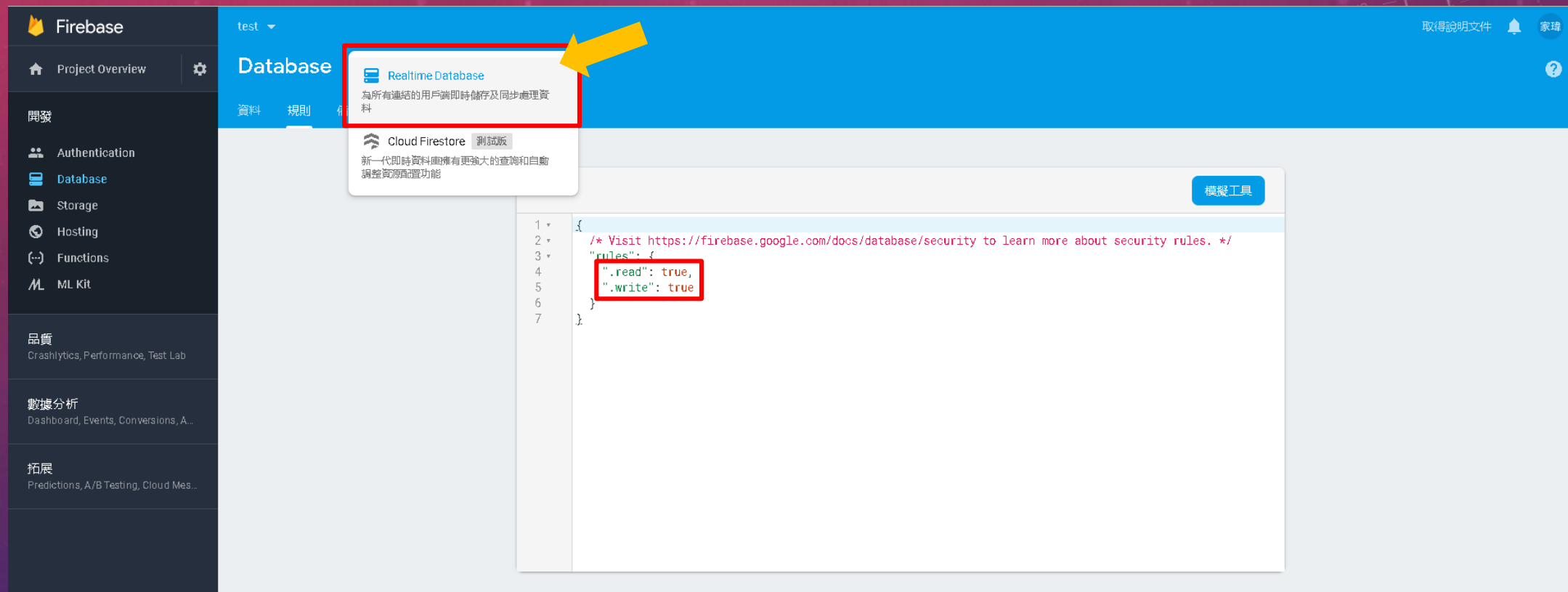
如要進一步瞭解適用於網路應用程式的 Firebase，請查看下列資源：

[Get Started with Firebase for Web Apps](#)

[Firebase Web SDK API Reference](#)

[Firebase Web Samples](#)

變更 REALTIME DATABASE 權限



The screenshot shows the Firebase console interface. On the left sidebar, the 'Database' option is highlighted with a red box. A yellow arrow points to this option. The main content area shows the 'Database' page with a red box around the 'Realtime Database' option. Below it, the 'Cloud Firestore' option is visible. The 'rules' section is expanded, showing the following code:

```
1 /* Visit https://firebase.google.com/docs/database/security to learn more about security rules. */
2
3 "rules": {
4   ".read": true,
5   ".write": true
6 }
7
```

The 'rules' section is highlighted with a red box, and the 'read' and 'write' permissions are also highlighted with a red box.

EXAMPLE-SET (1)

Firestore Realtime Database

建立使用者帳號

<input type="text" value="ID"/>	<input type="text" value="Email"/>	<input type="text" value="名稱"/>	<input type="button" value="建立"/>
---------------------------------	------------------------------------	---------------------------------	-----------------------------------

Mandy, mandy@gmail.com
Jason, jason@gmail.com
Gary, gary@gmail.com
張家瑋, jwchang@nutc.edu.tw

```
<body>
  <h1>Firestore Realtime Database</h1>
  <h3>建立使用者帳號</h3>
  <input type="text" id="create_id"           placeholder="ID"/>
  <input type="text" id="create_email"       placeholder="Email"/>
  <input type="text" id="create_name"        placeholder="名稱"/>
  <button id="create_user">建立</button><br><br>

  <label id="data"></label>
</body>
```

EXAMPLE-SET (2)

```
<head>
  <meta charset="UTF-8">
  <title>Firebase Realtime Database</title>
  <script src="https://code.jquery.com/jquery-3.1.0.min.js"></script>
  <script src="https://www.gstatic.com/firebasejs/5.7.1/firebase.js"></script>
  <script>
    // Initialize Firebase
    var config = {
      apiKey: "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX",
      authDomain: "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX",
      databaseURL: "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX",
      projectId: "XXXXXXXX",
      storageBucket: "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX",
      messagingSenderId: "XXXXXXXX"
    };
    firebase.initializeApp(config);
    var db = firebase.database();
  </script>
</head>
```


EXAMPLE-SET (3)

```
<script>
function writeUserData(userId, name, email) {
  db.ref('users/' + userId).set({
    username: name,
    email: email
  }).then(function () {alert("成功創建使用者!");})
  .catch(function () {alert("伺服器發生錯誤，請稍後再試。");});
}
```

EXAMPLE-SET (4)

```
$(document).ready(function(){
    /* 建立使用者 */
    $("#create_user").click(function(){
        var id      = $('#create_id').val();
        var email    = $('#create_email').val();
        var name     = $('#create_name').val();
        writeUserData(id,name,email);
    });

    function display_users(data_val){
        keys = Object.keys(data_val);
        users = '';
        //alert(data_val[keys[0]]['username']);

        for (var i = keys.length - 1; i >= 0; i--) {
            users += data_val[keys[i]]['username']+', '+data_val[keys[i]]['email']+'<br>';
        }
        $('#data').html(users);
    }

    db.ref('users/').on('value', function(snapshot) {
        //alert(Object.keys(snapshot.val()))
        display_users(snapshot.val());
    });
});
</script>
```

EXAMPLE-PUSH (1)

Firestore Realtime Database

即時通訊示範

Gary: 嗯嗯

Jason: 66666

Jason: ZZZZ

Jason: 安安

Gary: Hi

```
<body>
```

```
  <h1>Firestore Realtime Database</h1>
```

```
  <input type="text" id="push_name"           placeholder="暱稱"/>
```

```
  <input type="text" id="push_msg"           placeholder="訊息"/>
```

```
  <button id="push_test">Push</button><br><br>
```

```
  <label>即時通訊示範</label><br>
```

```
  <label id="data"></label><br>
```

```
</body>
```

EXAMPLE-PUSH (2)

```
<script>
  function push(name, msg) {

    var key = db.ref('message/').push({
      username: name,
      message: msg
    }).key;

    //alert("Push key = "+key);

  }
```


EXAMPLE-PUSH (3)

```
$(document).ready(function(){  
  
    $("#push_test").click(function(){  
        var name      = $('#push_name').val();  
        var msg        = $('#push_msg').val();  
        push(name, msg);  
    });  
  
    function display_msgs(data_val){  
        keys = Object.keys(data_val);  
        msgs = '';  
  
        for (var i = keys.length - 1; i >= 0; i--) {  
            msgs += data_val[keys[i]]['username'] +': ' + data_val[keys[i]]['message']+'<br>';  
        }  
        $('#data').html(msgs);  
    }  
  
    db.ref('message/').on('value', function(snapshot) {  
        display_msgs(snapshot.val());  
    });  
  
});  
</script>
```


EXAMPLE-UPDATE (1)

Firestore Realtime Database

-----[所有的貼文]-----

UserID: 4
Author: Mandy
Title: test3
Body: test3 test3 test3 test3
Star: 0

UserID: 3
Author: Jason
Title: test2
Body: test2 test2 test2
Star: 0

UserID: 2
Author: Gary
Title: test1
Body: test1 test1
Star: 0

UserID: 1
Author: 張家瑋
Title: test
Body: test test
Star: 0

```
<body>
  <h1>Firestore Realtime Database</h1>

  <input type="text" id="update_uid"           placeholder="UID"/>
  <input type="text" id="update_username"      placeholder="名稱"/>
  <input type="text" id="update_title"         placeholder="標題"/>
  <input type="text" id="update_body"          placeholder="內文"/>
  <button id="update_test">發文</button><br><br>

  <label>-----[所有的貼文]-----</label><br>
  <label id="data"></label><br>
</body>
```

EXAMPLE-UPDATE (2)

```
<script>

function writeNewPost(uid, username, title, body) {
  // A post entry.
  var postData = {
    author: username,
    uid: uid,
    body: body,
    title: title,
    starCount: 0
  };
  // Get a key for a new Post.
  var newPostKey = db.ref().child('posts').push().key;
  // Write the new post's data simultaneously in the posts list and the user's post list.
  var updates = {};
  updates['/posts/' + newPostKey] = postData;
  //updates['/user-posts/' + uid + '/' + newPostKey] = postData;
  return db.ref().update(updates);
}
```

EXAMPLE-UPDATE (3)

```
$(document).ready(function(){

    $("#update_test").click(function(){
        var uid      = $('#update_uid').val();
        var username  = $('#update_username').val();
        var title     = $('#update_title').val();
        var body      = $('#update_body').val();
        writeNewPost(uid, username, title, body);
    });

    function display_posts(data_val){
        keys = Object.keys(data_val);
        posts = '';

        for (var i = keys.length - 1; i >= 0; i--) {
            posts += 'UserID: '+data_val[keys[i]]['uid']+'<br>';
            posts += 'Author:  '+data_val[keys[i]]['author']+'<br>';
            posts += 'Title:   '+data_val[keys[i]]['title']+'<br>';
            posts += 'Body:    '+data_val[keys[i]]['body']+'<br>';
            posts += 'Star:    '+data_val[keys[i]]['starCount']+'<br>';
            posts += '-----<br>';
            posts += '-----<br>';
        }

        $('#data').html(posts);
    }

    db.ref('posts/').on('value', function(snapshot) {
        display_posts(snapshot.val());
    });

});
</script>
```

EXAMPLE-REMOVE (1)

Firestore Realtime Database

建立使用者帳號

<input type="text" value="ID"/>	<input type="text" value="Email"/>	<input type="text" value="名稱"/>	<input type="button" value="建立"/>
---------------------------------	------------------------------------	---------------------------------	-----------------------------------

刪除使用者帳號

<input type="text" value="使用者帳號"/>	<input type="button" value="刪除"/>
------------------------------------	-----------------------------------

-----[所有的用戶]-----

4. Mandy, mandy@gmail.com
3. Jason, jason@gmail.com
2. Gary, gary@gmail.com
1. 張家瑋, jwchang@nutc.edu.tw

```
<body>
  <h1>Firestore Realtime Database</h1>
  <h3>建立使用者帳號</h3>
  <input type="text" id="create_id"           placeholder="ID"/>
  <input type="text" id="create_email"        placeholder="Email"/>
  <input type="text" id="create_name"         placeholder="名稱"/>
  <button id="create_user">建立</button><br><br>

  <h3>刪除使用者帳號</h3>
  <input type="text" id="delete_user"         placeholder="使用者帳號"/>
  <button id="delete_test">刪除</button><br><br>

  <label>-----[所有的用戶]-----</label><br>
  <label id="data"></label><br>
</body>
```


EXAMPLE-REMOVE (2)

```
<script>
  function writeUserData(userId, name, email) {
    db.ref('users/' + userId).set({
      username: name,
      email: email
    }).then(function () {alert("成功創建使用者!");})
    .catch(function () {alert("伺服器發生錯誤，請稍後再試。");});
  }

  function removeUser(userId) {
    // 法一
    db.ref('users/' + userId).remove();
    // 法二
    /*
    firebase.database().ref('users/' + userId).set({
      username: null,
    });
    */
  }
}
```


EXAMPLE-REMOVE (3)

```
$(document).ready(function(){
    /* 建立使用者 */
    $("#create_user").click(function(){
        var id          = $('#create_id').val();
        var email       = $('#create_email').val();
        var name        = $('#create_name').val();
        writeUserData(id,name,email);
    });

    $("#delete_test").click(function(){
        var userId      = $('#delete_user').val();
        removeUser(userId);
    });

    function display_users(data_val){
        keys = Object.keys(data_val);
        users = '';
        //alert(data_val[keys[0]]['username']);

        for (var i = keys.length - 1; i >= 0; i--) {
            users += keys[i] + ". " + data_val[keys[i]]['username'] + ', ' + data_val[keys[i]]['email'] + '<br>';
        }
        $('#data').html(users);
    }

    db.ref('users/').on('value', function(snapshot) {
        //alert(Object.keys(snapshot.val()))
        display_users(snapshot.val());
    });

});
</script>
```

EXAMPLE-ON & ONCE (1)

Firestore Realtime Database

<input type="text" value="UID"/>	<input type="text" value="名稱"/>	<input type="text" value="標題"/>	<input type="text" value="內文"/>	<input type="button" value="發文"/>
-----[所有的貼文]-----				<input type="button" value="查詢發文"/>

```
<body>
  <h1>Firestore Realtime Database</h1>

  <input type="text" id="update_uid"           placeholder="UID"/>
  <input type="text" id="update_username"      placeholder="名稱"/>
  <input type="text" id="update_title"         placeholder="標題"/>
  <input type="text" id="update_body"          placeholder="內文"/>
  <button id="update_test">發文</button><br><br>

  <label>-----[所有的貼文]-----</label><button id="query_test">查詢發文</button><br>
  <label id="data"></label><br>
</body>
```

EXAMPLE-ON & ONCE (2)

```
<script>
function writeNewPost(uid, username, title, body) {
  // A post entry.
  var postData = {
    author: username,
    uid: uid,
    body: body,
    title: title,
    starCount: 0
  };
  // Get a key for a new Post.
  var newPostKey = db.ref().child('posts').push().key;
  // Write the new post's data simultaneously in the posts list and the user's post list.
  var updates = {};
  updates['/posts/' + newPostKey] = postData;
  //updates['/user-posts/' + uid + '/' + newPostKey] = postData;
  return db.ref().update(updates);
}

function display_posts(data_val){
  keys = Object.keys(data_val);
  posts = '';

  for (var i = keys.length - 1; i >= 0; i--) {
    posts += 'UserID: ' + data_val[keys[i]]['uid'] + '<br>';
    posts += 'Author: ' + data_val[keys[i]]['author'] + '<br>';
    posts += 'Title: ' + data_val[keys[i]]['title'] + '<br>';
    posts += 'Body: ' + data_val[keys[i]]['body'] + '<br>';
    posts += 'Star: ' + data_val[keys[i]]['starCount'] + '<br>';
    posts += '-----<br>';
    posts += '-----<br>';
  }

  $('#data').html(posts);
}

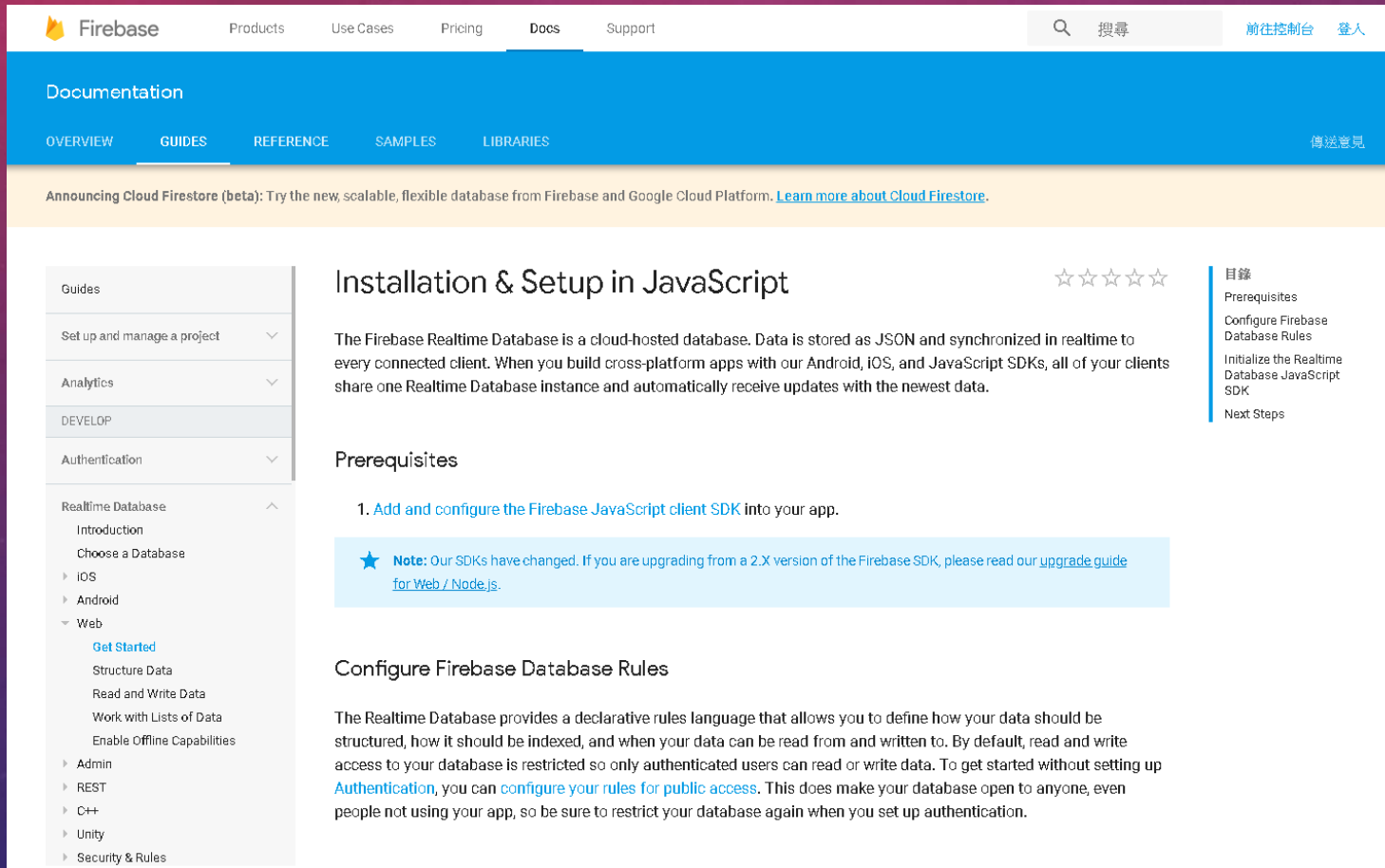
function query() {
  db.ref('posts/').once('value', function(snapshot) {
    display_posts(snapshot.val());
  });
}
```

EXAMPLE-ON & ONCE (3)

```
$(document).ready(function(){
    $("#update_test").click(function(){
        var uid      = $('#update_uid').val();
        var username  = $('#update_username').val();
        var title     = $('#update_title').val();
        var body      = $('#update_body').val();
        writeNewPost(uid, username, title, body);
    });
    $("#query_test").click(function(){
        query();
    });
});
</script>
```


延伸閱讀

<https://firebase.google.com/docs/database/web/start>



The screenshot shows the Firebase documentation website. The top navigation bar includes links for Products, Use Cases, Pricing, Docs (selected), and Support. A search bar and links to '前往控制台' (Go to Console) and '登入' (Login) are also present. Below the navigation bar is a blue header with 'Documentation' and sub-tabs for OVERVIEW, GUIDES (selected), REFERENCE, SAMPLES, and LIBRARIES. A yellow banner announces 'Cloud Firestore (beta)'. The main content area is titled 'Installation & Setup in JavaScript' with a five-star rating. It includes a left sidebar with a table of contents, a main text area with a note about SDK updates, and a right sidebar with a table of contents. The left sidebar lists guides like 'Set up and manage a project', 'Analytics', 'DEVELOP', 'Authentication', 'Realtime Database', and 'Admin'. The right sidebar lists prerequisites like 'Prerequisites', 'Configure Firebase Database Rules', 'Initialize the Realtime Database JavaScript SDK', and 'Next Steps'.

Firestore

Products Use Cases Pricing Docs Support

搜尋 前往控制台 登入

Documentation

OVERVIEW GUIDES REFERENCE SAMPLES LIBRARIES 傳送意見

Announcing Cloud Firestore (beta): Try the new, scalable, flexible database from Firebase and Google Cloud Platform. [Learn more about Cloud Firestore.](#)

Guides

Set up and manage a project

Analytics

DEVELOP

Authentication

Realtime Database

Introduction

Choose a Database

iOS

Android

Web

Get Started

Structure Data

Read and Write Data

Work with Lists of Data

Enable Offline Capabilities

Admin

REST

C++

Unity

Security & Rules

Installation & Setup in JavaScript

☆☆☆☆☆

The Firebase Realtime Database is a cloud-hosted database. Data is stored as JSON and synchronized in realtime to every connected client. When you build cross-platform apps with our Android, iOS, and JavaScript SDKs, all of your clients share one Realtime Database instance and automatically receive updates with the newest data.

Prerequisites

1. [Add and configure the Firebase JavaScript client SDK](#) into your app.

★ **Note:** Our SDKs have changed. If you are upgrading from a 2.X version of the Firebase SDK, please read our [upgrade guide for Web / Node.js](#).

Configure Firebase Database Rules

The Realtime Database provides a declarative rules language that allows you to define how your data should be structured, how it should be indexed, and when your data can be read from and written to. By default, read and write access to your database is restricted so only authenticated users can read or write data. To get started without setting up [Authentication](#), you can [configure your rules for public access](#). This does make your database open to anyone, even people not using your app, so be sure to restrict your database again when you set up authentication.

目錄

Prerequisites

Configure Firebase Database Rules

Initialize the Realtime Database JavaScript SDK

Next Steps



THANK YOU