CHAPTER 1

認識 Firebase

歡迎來到 Firebase 學習的園地。Firebase 是 Google 子公司，主導一個為行動應用程式開發的 BaaS（Backend As A Service）雲端系統，其主要的目的是希望在行動應用開發中，可取代部份或全部的後台功能，讓開發者能有一個不需要考量效能需求與功能驗證的後台，也不需要特別配合後台的開發時程，或伺服器的採購行政流程等，如此可以更有效率的開發應用。

Firebase 的起源故事

行動應用開始發展大約是 2008 年之間的事，Apple APP Store 與 Google Play（當時稱為 Android Market），上市後為無數的程式開發者創造了一個能無限發展的機會，特別是對於中小型開發者或者個人開發者而言，是一個前所未有的市場，經過數年的發展，開發者發現了一些問題。

行動應用是一個特別的市場，除了一開始，還能有一些單機應用外，一旦程式成長到一定的程度，就會有不同的後台需求，如帳號管理、推播管理、資料同步、訊息通訊交換等，都需要後台支援，因此造就了一個特殊的生態。在 Web 時代，一個系統只需要寫確認最後展出的 HTML 就好，不管你用 JSP、ASP，還是 PHP，只需一種環境而前後台還是用同一種方法寫的。但一個行動應用的最小規格，就需要 Android、iOS、Web Frontend/Backend，用Java，Objective-C 以及一種 Web 平台才能組成最小的系統。所以這個時期，有無數的開發者在互相等待時間，討論與處理相互的 Bug 與規格等問題。這其實是很浪費時間的。

於是大約在 2010 至 2013 年間，有無數的 BaaS 建立起來，或多或少解決了 APP 開發者的問題，其中最出名的兩家公司就是 Parse 與 Firebase，這兩家公司後來分別被 Facebook 與 Google 收購。雖然各大雲端營運商如 Microsoft Azure、AWS、Oracle Cloud、Facebook 都有類似的服務，但大多數會和原本的雲端服務有所關連，相對上複雜，對於開發者而言仍算是不低的門檻。Facebook 後來還因為各種原因，決定停止營運 Parse。所以 APP 開發者的關心度漸漸幾乎都轉到 Firebase 身上。

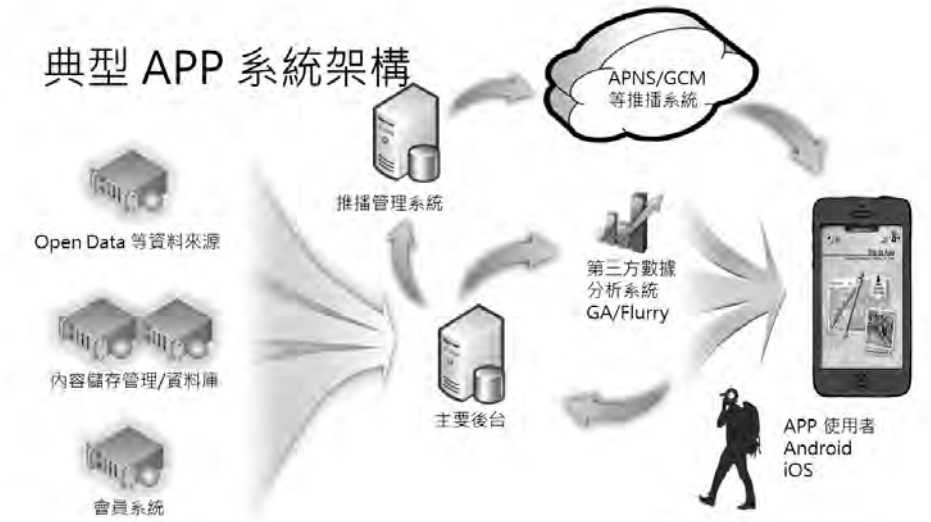
Firebase 是個完全獨立的系統，與母公司的 Google Cloud Computing 是完全分開的服務，也就是說它是特別針對行動應用而設計的。另一個很重要的事是母公司 Google 就是行動應用市場佔有率八成的 Android 開發商，對於自家產品更是第一時間支援，而且 SDK 效能特別的好，難得的是對於 iOS 的支援也不差。特別是在 2016 年的 Google I/O 大會中宣告，Firebase 大改版之後漸漸進入穩定期，之後就立下了在 APP BaaS 市場幾乎不敗的特殊地位。

Firebase 作為一個行動應用程式開發的 BaaS（Backend as a Service），希望取代部份或全部的後台功能，所以我們必須對它有所了解，才能充份發揮它的功能，本章就Firebase 的主要功能來做一個摘要的說明。

Firebase 有許多功能，若以名稱來看，不易了解實際提供的細節，如即時資料庫，並不只是一個 SQL 或 noSQL 而是讓開發者在沒有後台開發者的協助下，與資料庫同步資料，所以存取的方式與傳統的資料庫存取有很大的不同，它更像是一堆寫好的 API 讓我們去呼叫。下面就來討論有哪些功能是在開發前就必須了解的部份。

為什麼需要 Firebase

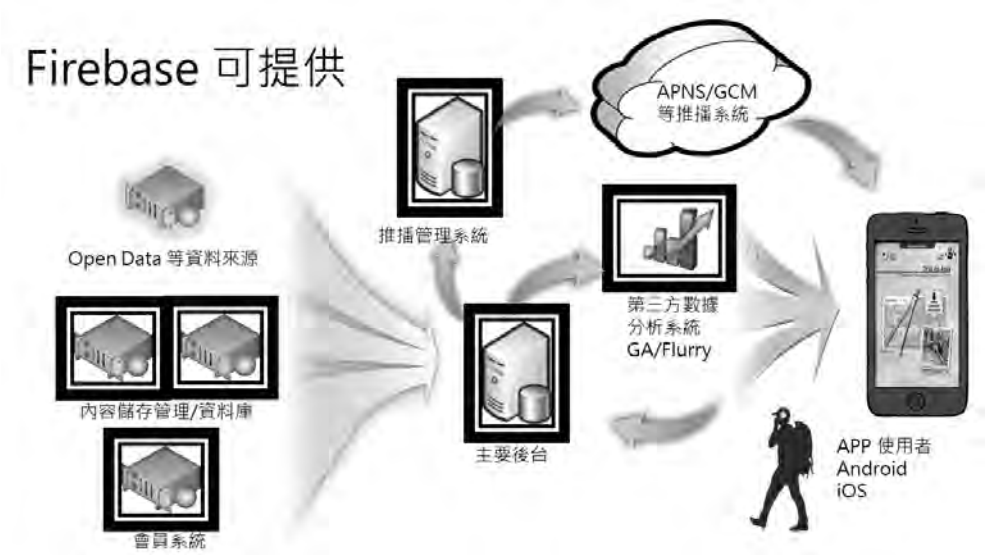
一個現代的 APP 多半和網路通訊有關，從 Market 下載 APP 開始，到更新內容、資料同步、推播，都離不開網路。典型的架構如下：



一個典型的應用，如「Instagram」之類的社交軟體或「旅行蛙」之類的遊戲，都有類似的需求。Android/iOS 兩個版本，均會透過後台去連結一個會員系統（也可能是 Facebook 或 Google 帳號），確認身分後把資料連到個人的資料庫或檔案庫中。有些APP也可能用到一些公開 API，如交通資訊等。認真經營的 APP 還會設定使用者行為分析工具，如Google Analytics 或 Flurry 作為改進的依據。當應用有事件發生時，如你的好友回覆文章時，還會使用推播系統來通知。

這個看起來簡單的後台，要做好卻相當不容易。光是內容管理來說，你若自己架設一個平台，有數十萬個活躍使用者，只要有一個大事件發生，數萬人同時打開 APP 更新訊息，你的 APP 就立刻破功。還記得數年前售票系統的卡票事件嗎？業者聲稱光是硬體伺服器就花了四百萬以上，完測過 22 萬人次的壓力測試，結果一開賣熱門歌手，數十萬人上線，系統馬上當機，狀況好一點時，也要卡個數分鐘。

即使不看使用效能問題，找個人來和 APP 開發者協同作業的成本也不算小。所以比較好的做法是，讓後台開發者只做關鍵性的功能，其他差異不是非常大的功能，就交由 BaaS 來執行是比較適合的，做出來的效果也比較會令使用者滿意。那 Firebase 到底可以做些什麼事情呢？我們來看下一張圖。



上圖中，黑框就是 Firebase 主要處理的部份。雖然不同於自行開發可以自訂 API，但 Firebase 提供了 iOS/Android 與 JavaScript 的存取 SDK 來和後台連絡，也就是 Firebase 寫了 APP 最難處理的資料同步的部份。

解決什麼問題

所以 Firebase 到底解決了什麼問題？以做一個類似 IG 的社群APP 為例，基本功能是登入，但是要以手機電話驗證、上傳照片、按讚、留言、通知等這樣的一個APP 需要的功能，來思考這個問題時，可能包括以下各項目。

帳號的問題

我們的設計是用電話號碼來驗證，比較麻煩的是要發送一個簡訊，除了系統設計外，還要有一個簡訊伺服器來提供這個服務，除了自己架設伺服器外，還要和電信公司等提供簡訊服務的公司談外接簡訊伺服器的申請與界接。使用 Firebase 就不用這麼麻煩，各種資源成本也整合在一起，Firebase 提供其他的帳號認證所需的一切資源，包括社群整合、簡訊、忘記密碼等。

儲存在哪兒？

登入完成後，面對使用者的第一件事情就是上傳照片、與好友討論等問題，第一個問題就是照片放在哪兒？使用者現在已經習慣了 IG、FB 等大公司的 APP，雖然我們的用量並不一定像他們那樣高，但也不能太差，至少要有簡單的存取方法，幾行程式就能存取、回傳狀況與有無限量的空間。

除了檔案的儲存外，資料的儲存也很重要，要有一個沒有容量限制、效能良好方便存取、自動備份、和前台可用的回傳機制。

Firebase 提供了儲存體與資料庫兩種使用資料的方法，分別用來處理檔案型式（照片、影片、文件等），和資料型式的內容。提供方便好用的 SDK，其用法和一般的 SQL 完全不同，大幅簡化了存取方式與安全設定。

即時資料傳遞與通知

在許多時候，APP 必須通知使用者有事情發生了，此時就要使用通知使用者功能。在實作上，通知有兩個層級的意義，一個是使用者正在線使用 APP 時，另一個是使用者已經關掉 APP，但有新的事件或訊息發生時。

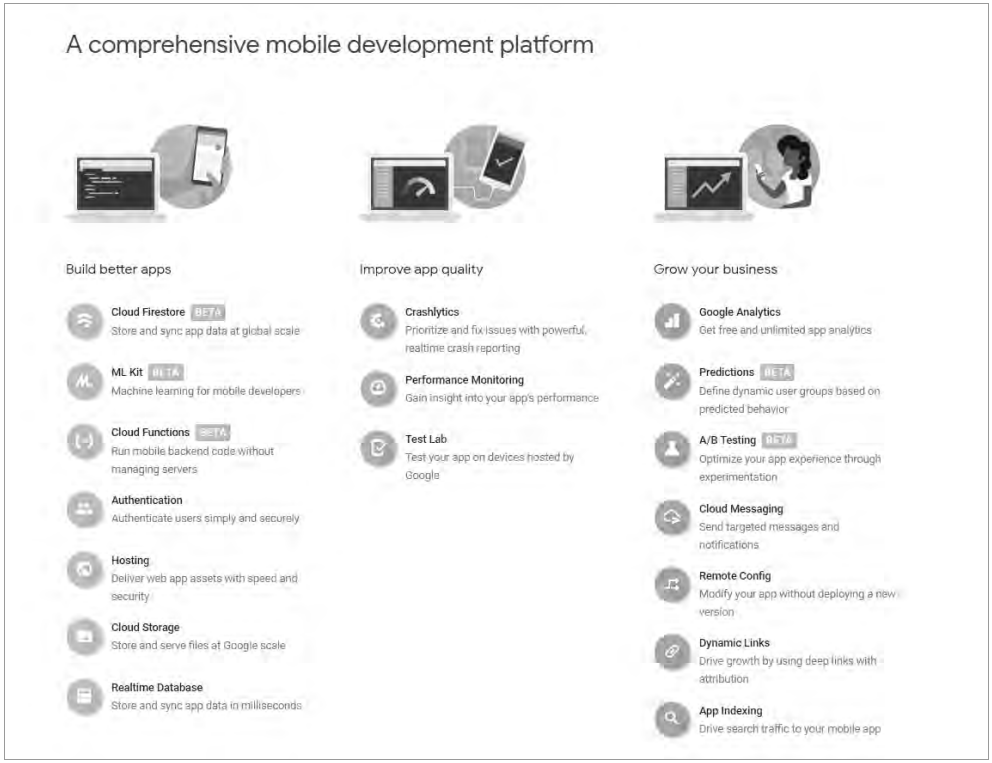
Firebase SDK 在很多方面實作了遠端通知的功能，如 Remote Control 即時通知更改設定，使用者在特定功能的預設值，由 Firebase 控制台設定後，APP 馬上就會收到更改設定的通知。Realtime Database 可以在有新資料時即時通知你，例如使用者正在討論區看一則討論時，如果有新的討論項目出現，就會立刻通知 APP 更新。這些都是 SDK 的內建功能，只要寫收到資料後要做的事就好，監聽的多執行緒程式已幫你寫好了，離線處理也幫你寫好了，只要有網路，條件符合就會自動更新。

如果使用者已經關閉了 APP，仍可以使用雲端訊息功能，也就是推播，用同一種方法整合 APNS、GCM，而且幫你做好使用者訂閱分類，寫好推送 API，最重要的是效能，雖然沒有保證送達，但記錄上 9 成的訊息在 0.25 秒發出去的效能可不是自己架後台能做出來的效能。

Firebase 主要功能

在用 Firebase 為後台之前，先要了解 Firebase 的特性與功能才能規劃出一個好的用法，本章會分析 Firebase 最重要的功能，這些功能在使用或規劃時應注意的事項與限制。本章還未討論到開發的細節，所以即使你完全沒有開發經驗，也可以看得懂。

首先，由功能清單開始。打開 Firebase 的首頁就會看到一個功能列表，列出了許多現有的功能，而且在持續成長中：



這些功能基本分為兩大類：開發測試 APP 或 Web 使用的，例如即時資料庫、認證系統、雲端檔案儲存、網頁代管、Android 自動測試等。另一類是分析管理用的，包括 Google Analytics 分析、Ad Mob 整合等。

但大多數 APP 開發者都是為其中一個或多個功能而使用，有些功能很容易理解，有些就要特別說明一下。個人認為幾個要特別說明的項目分別為

Authentication 認證、Realtime Database、Analytics、Cloud Messaging，其他的項目功能相對來說比較容易，我在最後面一次列表說明，首先從認證談起。下面是 Firebase 功能中最常用的項目說明，也是本書會詳實製作的部份。

Authentication

帳號密碼認證，是 APP 最常見也最需要的一個東西。認證是一個很常見的功能，但要做到完成卻不太容易，Firebase 除了提供認證外，也提供了許多 APP 需要的 API，比如說：

- **手機自動重驗證**

  開發者不用自己分析記錄登入時間、位置、帳號，只要使用者有正確通過認證，SDK 就會自動記住，即使從背景移除，也能自動與伺服器連絡，取得更新狀況。其過程不用使用者輸入帳號密碼，只靠機器憑證與 ID 運作，有完整的安全設計，同時自動登入也會檢查其帳號的完整性，如密碼更新狀態變更、自動登出等。

- **匿名登入功能**

  這是一種安全設計，有時我們並沒有計畫讓使用者登入，但後台 API 要確認是我們自己的 APP 在存取，而不是任意程式可以存取，此時就可以產生一個匿名級別的憑證，我們可以依據該憑證來確認是自己的 APP 在存取，而不是駭客在存取資料。

- **整合多種社交認證**

  可以整合 Facebook、Twitter、GitHub、Google 等社群帳號，使用者可整合成一個，用任意一個通過認證，認證完同樣可以自訂個人屬性。

- **電子郵件認證功能**

  可以讓使用者在使用 APP 進階功能前，先透過寄送電子郵件方式認證連結的方式，確認使用者有正確的電子郵件可以使用。

- **手機簡訊認證**

  可以讓使用者在開啟進階功能前，透過簡訊確認方式，確認使用者的手機能正確收到簡訊。

- **忘記密碼功能**

  在使用者忘記密碼時，可以透過電子郵件的認證方式，重新設定密碼。

- **建立與設定基本資料**

  可以自動化或人工的方式新增帳號密碼、變更密碼等。

以上功能你不一定會用到，但是規劃時，必須了解其實許多功能已經完成，必要時調用即可，但也不必開啟每一個功能，除了成本問題，必要性才是最重要的關鍵。例如以台灣為主要市場的 APP，就不一定要整合 Twitter 驗證；個人識別不需要很嚴謹的 APP，也沒有必要強制電子郵件與簡訊認證，只要通訊或電商軟體做即可。

另一個狀況是公司已經有認證時，需不需要 Firebase 呢，答案很可能是需要，Firebase 提供了相對應的整合方法。因為一般的認證系統，如 LDAP 資料雖然完整，但不是針對 APP 所設計，所以在一些功能上並不完整，如電子郵件忘記密碼功能、簡訊認證、自動登入等，若要用到這些功能，你就需要規劃開發後台的新功能，或直接整合 Firebase 認證解決問題。

Realtime Database

Realtime Database 是許多 APP 應用開發者使用 Firebase 的最重要原因，但它本身與大多數開發者熟練的關聯式資料庫 SQL 有很大的不同。

因為要讓 APP 開發者直接使用，因此使用前並沒有任何的初始化動作，也沒有建立資料 Table 欄位等問題，所以需要不同的方法處理即時資料庫，實際上是一種 no SQL 資料庫，它是一個巨大的 JSON Tree，所以它存取時不是用 SQL 指令，另外因為沒有固定欄位的設定，所以它的資料結構必須使用不同於 SQL 的思考方式。這樣做的好處是在擴充或改變時，不用針對資料庫做設定。

即時資料庫提供的不只是資料的存取，它的 SDK 功能也包括了監聽等功能，也就是說，我可以在開啟應用的同時，當資料庫資料有更新時，很容易的即時同步到本機中，而不需要特別在後台設定一些監聽封包程式。這在需要即時監聽的應用特別好用，如討論區、即時通訊等，而且大多數的同步都是以毫秒計的時間完成同步的，不但支援 APP，也支援 JavaScript 同步到網頁或其他應用平台上。

即時資料庫的 SDK 還提供離線功能，這表示你的社群對話、通訊資料都可以在網路不通的狀態下運作，當網路恢復時，自動同步回資料庫。

安全性設定也是很重要的，Firebase 設計了相當完整的資料存取，可以精準的讓每一個使用者都只看到自己應該看到的資料，而不會看到不該看到的資料。

當然 no SQL 也並非沒有缺點，比如說在存取資料時，由於並不是固定的 Table 結構，若沒有好好管理每一次的存取，或在更新版本時未能正確更新，就有可能造成不正確的欄位、缺少的欄位、不對的格式等問題，它不會也不可能幫我們檢查，這都是程式設計者必須自己動手管理、修正的部份。查詢時若不擅於設計 no SQL 的架構，而要經常性的大量存取資源的話，費用就會非常高，因為這部份的價格是以存取量來計算的。一個熟悉 APP 開發、但不熟悉 Firebase 的開發者最常見的錯誤之一是，把所有資料同步到本機，轉成 SQL lite，再用自己熟練的方法處理，這雖然快速解決了技術上的問題，但造成相當大的存取量。另一個初學常犯的錯誤是，為了開發方便，關掉權限設定，但忘了把它再開啟，這個錯誤經常造成有心人試著存取不需要的資料，或者用自己的資料，服務不明的第三方。雖然我們不是在討論防駭技術，但基本的安全觀念還是要有，所以最低限度是要設定為匿名登入者。

使用 no SQL 很重要的一個思考是分散與減少查詢。

在使用關聯式資料庫時，經常使用大量的查詢，往往在大資料量的狀況下，浪費了大多數的查詢能量。在後面的章節中會繼續探討如何設計一個良好的資料庫。

Analytics

Analytics（分析）會在iOS和Android應用程序中顯示有關用戶行為的數據，使你能夠更好地優化產品。這個工具完全免費且無使用量限制。除非你想自己追蹤使用者特定的行為，否則這個工具幾乎不用設定，對使用者也無任何影響，所以我認為每一支 APP 都該啟動它，以便了解使用者有多少人、在哪兒使用、是男是女、用什麼機器、都在什麼狀況當機。

如果你有心想做更多分析，Analytics 還提供了很多工具。

比較特別的是，Firebase Analytics 的各項設定，是和 Firebase 其他資料整合在一起，例如登入使用者可以自訂屬性，用來追蹤不同屬性人的不同行為，如愛棒球的人與愛籃球的人有什麼行為上的不同，之後就可對不同屬性的人作不同的優化。

Cloud Messaging

這其實就是 APP 的推播（Notifications），一般文件簡稱 FCM，它幫助 APP 開發者在沒有後台開發者或電腦程式執行的協助下，前端開發者就可以發送訊息。

進一步來說，它也協助客戶群的分眾化與效能化，使用者可以簡單分類或訂閱，就不用後台開發者來分類。後台開發者只負責分類的項目和發送，不需要自己分析名單、建立分割再送到發送器，只要告訴 Firebase 送某個訊息給訂閱某項目的使用者即可。

Cloud Storage

其實就是檔案空間，和 Google Drive 或 iCloud 不同的是，這是給所有 APP 使用者用的，所以存取是用 SDK 存取，而且效率更高。

Remote Config

這是一種遠端參數提供機制。很多時候，大家對設計的方法有不同的意見，但往往都只能在經過發佈之後，才能得到真正的結果。但 A/B 之類的工作是相當麻煩的，特別是要設定不同比例的使用者用不同的設定時，要設定的東西還不少。決定最終設計之後，還要經過改版、上架、更新版本等過程，更是花時間。Remote Config 就是這個問題的解決方案，可以使用遠端的方式設定參數。

其他功能

Firebase 還有許多本書尚未提到的項目，因篇幅關係未能包括在內容中，包括 App Indexing、AdMob、AdWords、Invites 等提供 Google 各項搜尋與廣告業務的進階整合，或者是與行銷整合推廣 APP 有關，多屬於 APP 成熟階段的進階功能。在本書最後一章會有詳細的介紹，若有機會寫到進階版，再和大家討論這一部份。

訂價問題

對於開發 APP 剛起步的人來說，Firebase 幾乎是可以零成本、完全不用錢的。但整體來說，系統並不是完全免費，而是部份功能免費、部份功能限量免費。Analytics、Cloud Message（推播）、認證系統對於新創團隊的助益就非常大，每月免費 10 G 流量的資料庫，也可支援你的 APP 直到賺到錢為止。

Google 把使用者消費分為三個不同的計畫方案，Spark Plan（免費）、Flame Plan（USD$25/月）、Blaze Plan（以量計價）。

常用免費的功能有：

Authentication（簡訊認證有用量限制10k/月）、Analytics、Cloud Messaging等。

限量的常用免費功能有：

- **即時資料庫**：同時一百個連線數，1G 的資料量與 10G /月的流量免費。
- **雲端儲存**：有 5G 的資料量，下載 1G /日的流量，另有上傳要求 20k 次/日，下載要求 50k 次/日的免費量。
- **網頁空間**：1G 的資料量與 10G /月的下載量免費。

這些免費的用量，絕對足夠在開發使用，因為並沒有一用到就會收費的項目，在 APP 使用者不多時，也不足以動到要收費的項目。

早期比較容易會用到需要費用的項目，像是線上討論區或通訊類軟體，若是上百人同時監聽資料庫就容易達到收費項目標準，另外大多是流量的問題了。

最初一定是用免費計畫，若使用量超出預期，建議一開始採用 Blaze Plan。

※備註：引用 2018 年初官方網站的資料，詳情請參考官方網站。

教育與協助你的團隊

要決定在你的專案使用 Firebase，比較難的項目其中之一是，雖然整體來說 Firebase 解決了許多的問題，但對大多數開發者來說，它就是新的東西，是要學習的，因此多少會有反對的聲音。

所以需要讓使用者了解所執行的工作，其實是比以往更輕鬆的，「只需 30 分鐘就可以完成建立帳號、登入、忘記密碼、簡訊認證」的討論與實作，最後達成業主與老闆的目標，就是我們最終的目的。

要達成這個目的，就要深入了解 Firebase 並做出最適合的判斷，在實做架構上，使用 Firebase 可以達到節省前後台 RD 的時間，並提供給 RD 最需要的協助工具（例如提供本書）才能達成我們的目標。

其中影響費用最大的部份，就是資料庫的存取方式、離線設計、排序設定等，如果沒設計好，成本會相差數倍到數十倍。在某些狀況下，差個數千倍也不是太奇怪的事，這都是需要特別留意的事。

深入了解每一個功能是需要一點時間的，除了看本書的案例外，思考自己的需求才是重點。可以由幾個項目先入手，首先登入這項目就幾乎不會有比使用 Firebase 還方便的方法，除非認證功能已在別的 APP 完成過且可以複製使用的方案，否則只是新增一個忘記密碼功能，就會花數週的規劃設計與測試。