主流視覺框架應用於 App 開發效率上之比較研究
App Development Efficiency Comparisons on Major UI Design Frameworks

郭芳瑜
國立臺北科技大學
Fang-Yu Kuo
wendy814111@gmail.com

陳英一
國立臺北科技大學
Ing-Yi Chen
ichen@mail.ntut.edu.tw

摘　要

隨著滑世代的出現，使用者感官、感受漸受關注，凸顯使用者介面(User Interface, UI)以及使用者體驗(User Experience, UX)的重要性。UI 著重於視覺上的介面設計，將產品定位與操作流程加以編排並美化，提供給使用者正面感受；UX 可稱為互動設計，秉持 UCD (User-Centered Design)設計[1]來優化產品或是流程。但是，如何能有效率實踐並優化使用者體驗，在開發中是一大重點。

以資訊產業之行動應用程式為例，實踐使用者體驗方法大多採用視覺框架，但如何在眾多框架中採取最適合的框架，以達到提升開發效率、降低學習成本。現今市場中主流框架如 Ionic Framework、Famo.us Framework 以及 Framework7 Framework，著重的載體不盡相同。本研究首先進行視覺模式之分析，並針對行動裝置之主流視覺框架進行探討，分析 Ionic Framework、Famo.us Framework 以及 Framework7 Framework 三種框架之特性，將研究成果應用於資訊產業，以擴大研究成果之效益。

本研究遵循軟體工程模式，以行動應用程式實例進行測試，驗證其三者框架對於介面與體驗的支援度，評估各框架之支援項目與學習成本。透過本研究提供最適合之視覺框架，提升開發行動介面之效率，強化後續開發與維護效率。

關鍵字：使用者介面、行動應用、視覺框架、Ionic、Famo.us、Framework7

一、前言

隨著行動裝置的快速演變，智慧型手機、平板電腦等載體層出不窮，再加上行動網路(3G、3.5G、4G)的蓬勃發展，使用者能不受到時間與地域的限制下，暢行使用各種雲端服務。行動裝置所開發的應用程式—行動應用程式(Mobile App)在各類平台上提供的數量日益龐大，在開發方式上，大致分為：原生應用程式(Native App)、網頁應用程式(Web App)、混合式應用程式 (Hybrid App) 開發，其中混合式應用程式使用趨勢增加，其在開發成本上最能降低，本論文將以混合式應用程式為例，探討在行動應用視覺框架上如何選擇，以利三方(使用者、客戶、開發商)能有效解決其需求，達到最大效益之目的。

二、研究目的

智慧型手機普及化，其中行動應用程式種類越來越多元，開發商欲更快速獲利，便會縮短開發速度，也間接影響其產出的應用程式是否能擁有先前的品質亦或是更佳？對於開發方式的選擇上也成極具探討的議題之一。

在現今快速變遷的商業環境，開發商如何能透過高效能的視覺應用框架，並且進一步提升開發效率，減輕維護負擔，正是現在所面臨的重要課題之一。在 The Elements of User Experience[2]中提到，對於開發網站、行動應用程式的過程提供了清晰的架構，如圖 1 所示。

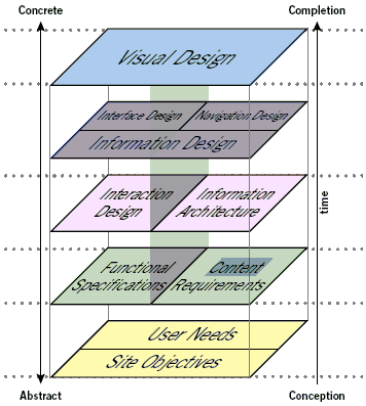

圖 1 使用經驗的元素

有上述資料可知，在製作網站或是行動應用程式中，影響到使用者感受不外乎為視覺介面、流程互動、內容功能等。而營造使用者經驗的過程中由先至後，概念到實作，下到上，抽象到具體，可分為五個平面(Layer)：Strategy(策略)、Scope(範圍)、Structure(結構)、Skeleton(骨架)以及 Surface(表皮)。五個平面環環相扣，大多數的情況下每個平面的各項決策都是基於在下方平面的決策；每一個平面都有其不同的設計技巧與可用的設計工具。

三、相關研究

3-1　行動應用程式風格

在行動應用之視覺介面範疇中，Human Interface Guidelines[3]按照視覺、行為特性、數據模型、使用者體驗等定義三種不同應用程式類別，以下分別敘述：

3.1.1 沉浸型應用程式

沈浸型應用程式特點在於設計模式會讓使用者長時間在與其互動，使用者是否跳離取決於內容豐富多寡，在使用上花費時間並無固定，因此此類型的介面設計上通常會使用全螢幕模式，讓使用者能專注於內容，實例上如遊戲、影片等。

3.1.2 公用型應用程式

公用型程式主要提供使用者每日所需，原則上對於使用者輸入要求極低，透過簡單的操作達到使用者需求，通常依照標準使用者介面元件開發即可完成，流程簡單、極簡設定是一大特色，如：奇摩氣象 App、系統內建鬧鐘等，使用者只需開啟、點選即可獲取資訊。

3.1.3 產出型應用程式

產出型應用範圍從社群經營至網路銀行等皆可包含在內，除此之外，使用的時間會依據使用者的場合、工作等有所不同。產出型能提供給使用者較多元資訊，相對於公用型應用程式來說，其功能設計上較於複雜與繁瑣，在視覺設計上通常包含著列表檢視、細節說明等階層式結構，舉例來說，如：Gmail、FaceBook 等。

3-2　視覺模式分析

以產出型應用程式為例，在 Mobile Design Pattern Gallery[4]中提及各式視覺模式，對此我們針對產出型應用程式進行分析，大致來說，產出型應用程式在視覺介面中需要七種視覺模式，以下分別敘述：

3.2.1 Navigation Patterns

導覽設計，在行動裝置中佔據重大意義，重點能讓使用者點選進入所選功能介面上。對於使用者來說，好的導覽設計在使用能直覺，簡單操作即可獲得所需資訊；反之讓人感到不知該如何下手。此模式將分成主要導覽以及次要導覽，以下分別敘述：

主要導覽，通常用於主選單上，常駐於畫面某處，擁有索引標籤控制引導使用者至其他功能介面，主要導覽的模式有如 Tab Menu、List Menu 以及 Gallery，如圖 2 所示。現今行動應用程式而言，最廣泛使用的模式為 Tab Menu，常駐於畫面下方提供使用者輕易跳轉至其他功能。

圖 2 主要導覽設計範例圖

次要導覽則是基於主要導覽模式下的導覽設計，將主要導覽區分後的功能再進一步分類，亦及相對於主要導覽模式而言的導覽模式即為次要導覽，如下圖，Tab Menu 為主要導覽，則 List Menu 為次要導覽。Tab 與 Tabs、Tabs 與 Lists 皆是常見組合。

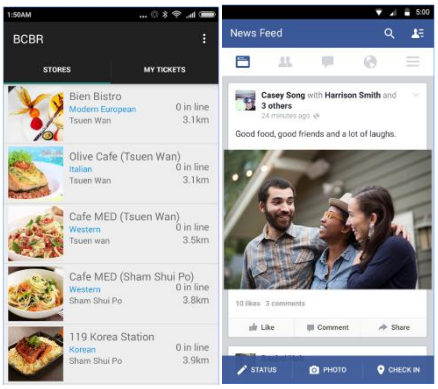

圖 3 次要導覽設計範例圖

3.2.2 Forms Patterns

在網頁應用程式的應用上，依舊大量依賴表單來處理資料輸入等設定，但由於載具的改變，螢幕尺寸縮小以及可視範圍減少，對於表單的應用如：註冊登入、電子商務(Electronic Commerce, EC)購買運送資料等，其頁面設計尤為重要，因此在行動裝置上需減少輸入事件增加可讀性，舉例來說，註冊頁面上通常資料需要有姓名、帳號、電子郵件、生日、密碼、密碼確認等個人資訊，但為了降低使用者理解的困難與輸入的負擔，因此通常建議依據使用性質調整並移除「無重要功能的元素」，如姓名、生日、密碼確認等非必要性資料，以及採用直式標籤與浮水印文字欄位增加使用可讀性（如圖 4）。

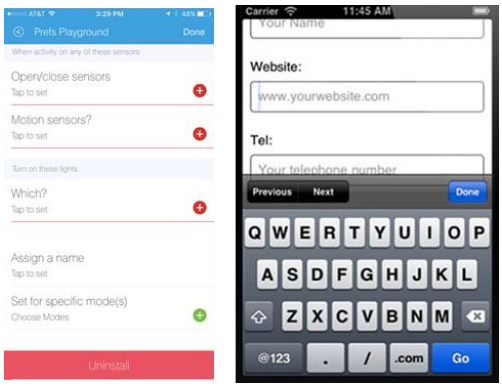

圖 4 表單設計範例圖

3.2.3 Tables Patterns

對於產出型應用程式來說，表格的呈現是不可避免的，尤其對於財金股票、航空業來說，但若將網頁的設計方式直接平移至行動裝置上，對於使用者而言，因可視範圍縮小、內容資訊過多，導致呈現需要增加卷軸，此做法將完全違反 UCD (User-Centered Design)設計思想[5]，因此，必須重新評估只呈現重要資訊的過程是必須的。

如下圖，對於表格顯示方式有許多種，如 Headerless Tables、Grouped Rows 是較常用的做法。Headerless Tables 適合用於顯示項目組合與查詢結果，舉例來說：威秀 FUN 電影此款 App，電影資訊眾多，若全呈現於同一頁面則會導致使用者難以理解與選擇，因此需要進行兩階層分類，先以簡單明瞭的資訊（如電影海報、電影名稱）讓使用者快速選擇，後才顯示該電影詳細資訊（如電影簡介、影城訂票資訊等），如同清單方式排列，強化瀏覽與選取可讀性。

Grouped Rows 能使表格資料容易理解，如同 iOS 聯絡人，依據聯絡人姓名進行分類，對於使用者能夠加速查找效率。

圖 5 表格設計範例圖

3.2.4 Search, Sort, and Filter

使用行動應用程式大多已有採用查詢、排序與篩選模式，此種模式能夠提升使用者查找資料的效率，對於電子商務而言尤為重要，以下分別說明其重點：

在查詢模式下，最被廣泛使用的有 Search with Auto-Complete 以及 Dynamic Search，前者設計透過演算法分析出一組使用者可能要的結果，只需點選即可進行查詢，減少使用者輸入次數，實例用於 Google Search、Google Play Store 上；後者模式可視為動態篩選，對於有限資料組上快速查找是有益處，常見的實例為 Line、手機內建通訊錄等(如圖 6)。

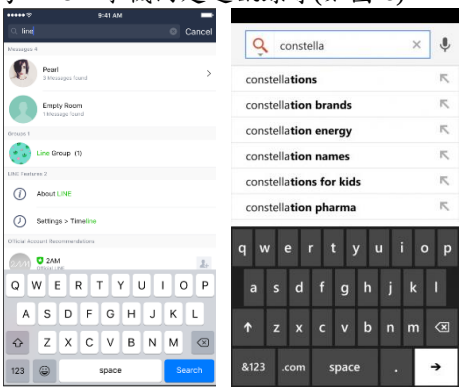

圖 6 查詢設計範例圖

排序模式區分成外在排序與內在排序，區分依據在於介面是否提供事件與使用者進行互動，外在排序上被廣泛用於電子商務(Electronic Commerce, EC)，使用者可以依據產品相關度、價格高低等進行排序；反之內在排序上，則是在程式端處理後，直接呈現於載具上的排序，對此使用者無法對其做排序上的改變(如圖 7)。

圖 7 排序設計範例圖

篩選模式主要仰賴使用者選取擬定條件，並以此條件加以優化查詢過後的結果，常見方式為 Onscreen Filter，相似於畫面排序，將篩選項目與查詢結果呈現於畫面中，點擊即可切換篩選條件，如 Google Search 將條件分類成圖片、新聞、影片、地圖等項目，加以分類使得使用者能縮小查詢範圍(如圖 8 所示)。

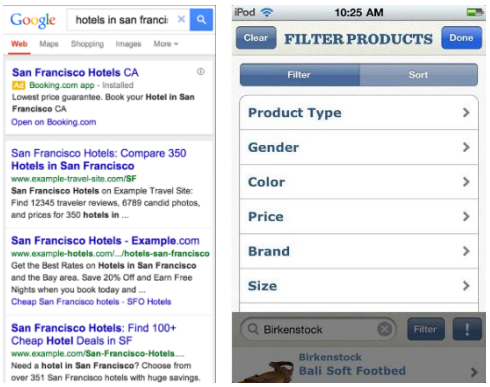

圖 8 篩選設計範例圖

綜合以上三種模式，對於網頁與行動裝置而言，適當的搜尋排序機制能夠提供給使用者便利性，但必須避免增加使用複雜化與呈現複雜化，過度設計導致使用者困惑。

3.2.5 Tutorials and Invitations

對於產出型應用程式而言，因功能設計上較為繁瑣，因此對於初次使用者來說，如何能夠快速上手是一大重點。邀請設計模式能夠引導使用者至預定功能，主要分成三項目，首先為 Dialog，簡單說明文字呈現為目前最簡易、廣泛使用的邀請模式，也因簡易呈現導致容易被使用者忽略；其次是針對行動裝置而設計的 Tour 導覽頁面，利用簡潔文字、視覺圖像以及最後明確的關閉按鈕組合而成，通常僅透過左右滑動等簡單瀏覽的方式，強調主要功能，如圖 9。

圖 9 邀請設計範例圖

3.2.6 Tools Patterns

Bill Scott 曾提出設計上的六大法則[6]，其中兩項法則為：操作直接性以及保持輕量化，此法則亦即參考於行動介面的工具與動作設計。操作直接性主張將輸入區與輸出區放置於同一位置，介面應直接回應給使用者進行互動；保持輕量化則是針對操作步驟優化，以不影響使用感官的前提下縮減操作次數，簡化使用者操作上的困難。

綜和上述兩項法則並套用於工具模式，可以將工具模式分成四項，分別為 Toolbar、Call to Action Button、Inline Action 以及 Bulk Actions，以下分別說明：

Toolbar 工具列，用來放置畫面上可操作的按鈕區域，如音樂播放軟體 KKBOX(圖 10 左)，將下方工作區域放置音樂處理基本事件；iOS Safari 下方皆放置網頁基本事件按鈕(圖 10 右)，Toolbar 的設計方式遵照操作直接性能給予使用者便利的操作模式。

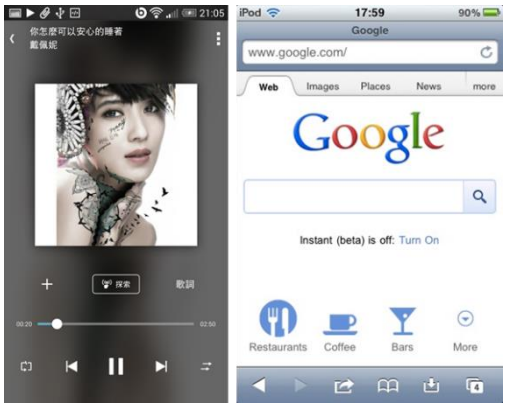

圖 10 Toolbar 設計範例圖

Call to Action Button，如圖 11 所示，使用 Toolbar 的前提為擁有多樣功能需要呈現，但當畫面上只擁有唯一主要功能時，則使用 Call to Action Button 較為適當，包含 ACCUPASS、花旗行動市集等，以顯著的視覺效果引導使用者。

圖 11 Call to Action Button 設計範例圖

Inline Action 行內行動可以用於畫面的特定物件上，如圖 12 左，在組合項目旁加設圖表樣式，引導使用者點入導頁至別頁；行內行動的另一做法如同圖 12 右，KKBOX 提供下載功能，點選下載時下方歌曲詳細資料將替換成下載進度，以便使用者了解進度狀況，透過此種雙態模式節省視覺空間。

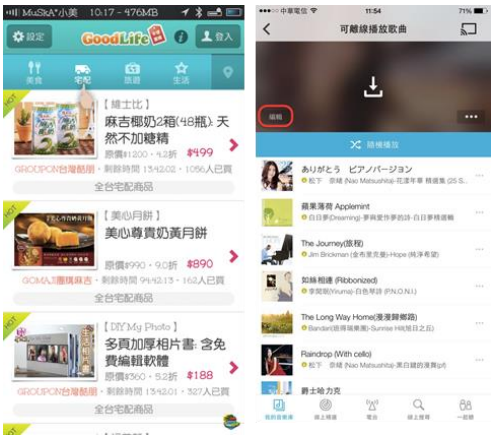

圖 12 Inline Action 設計範例圖

Bulk Actions 批次行動的設計模式優化一般功能設計上，重複操作的缺點，舉例來說，如欲傳送給他人多張圖像檔，若按照基礎功能上必須重複多次傳送操作；但增設批次行動後，縮短許多功能點選的操作，對於使用者而言更加便利，現今 Line、相簿等行動程式皆增設此模式。

3.2.7 Feedback and Affordance

根據 Jakob Nielsen 所提出的十大易用性原則[7]中提及「Help users recognize, diagnose, and recover from errors」，幫助使用者識別、診斷並且從錯誤中恢復，減少損失；或是給予明確文字說明指導方向(如圖 13)，而非給予開發商理解的錯誤代碼。

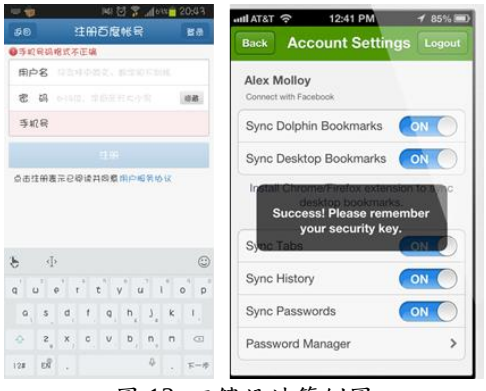

圖 13 回饋設計範例圖

四、技術分析

開發商在開發過程上追求最佳效益、最低成本，為了也解決此狀況，對於開發平台、架構的抉擇是一大議題，因此本研究將探討三套以 AngularJS 為基礎開發之視覺框架(UI Framework)，分別是 Ionic、Famo.us、Framework7，以下分別敘述：

4.1 Ionic Framework[8]

Ionic 採用 HTML5 應用程式開發框架，以 Cordova 為平台，並結合 AngularJS 語言進行開發，其框架控管 ui-router，因此在開發設計上能著重於功能與介面；除此之外，因此框架基於 Cordova 為平台，因此在設計上可以使用許多 Cordova 插件，能控制行動裝置內之硬體提升效能，使其在使用感官上與 Native 模式並無差異。Ionic 在設計著重在視覺感官與體驗上，在視覺介面上擴充 Syntactically Awesome StyleSheets (SASS)強化應用，較適合於 Hybird 模式之應用程式的開發研究。

上述所提及此框架著重於設計體驗上，其具體實例在於此框架提供全套視覺組件，並符合人機界面(Human Interface Guidelines)原則(如圖 14)，因此在行動裝置上模擬並能符合使用者視覺感受。除此之外，Ionic 也提供許多自定義之圖標樣式與標準化輸入，採用 IconFont 向量檔案的方式適應不同載具；HTML5 輸入元件應用讓開發者能立即應用開發。

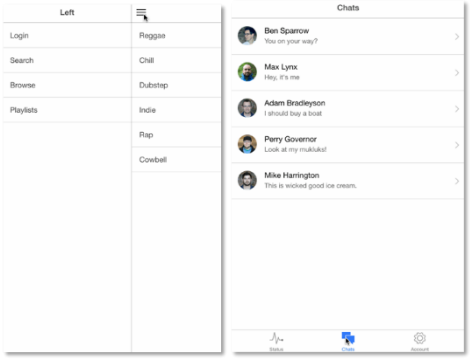

圖 14 Ionic 元件範例圖

4.2 Famo.us Framework[9]

Famo.us 為近幾年開發出來的全新框架，此框架主要著墨於 Webkit 核心當中，針對 DOM 與瀏覽器渲染的部分，開發了一套全新的渲染方式、全新的物理引擎，用來取代原本瀏覽器中，對於整個 DOM 的操作與渲染方式。此框架切入 HTML 方式與現有方式不同，其使用 WebGL 技術將 HTML 的 DOM 組合在一起，把所有的內容存放於一個 canvas 裡顯示，相似於 HTML 遊戲引擎的做法，此技術能夠讓行動應用程式執行速度維持在 60fps 下，如同原生應用程式般流暢。

框架主要進行下列項目的優化，以下針對第一點 Render Tree 進行說明：

1. 重新產生自定義的 HTML Elements
2. 簡化 DOM 渲染流程
3. 優化 DOM/CSS/JavaScript 效能
4. 解決瀏覽器之間的實作差異
5. 簡化應用程式開發流程

由圖 15 中所示，右側為現有 Render Tree 結構，在 DOM 的結構上為高度耦合、多階層的呈現，對於渲染效能上無法與原生應用程式相比擬；使用 Famo.us 時，其 Render Tree 則是如圖左側，將整個 HTML 的 DOM 扁平化，呈現整齊的樹狀支結構，所以在整個 Render Tree 的結構上，效能上就會有提升。

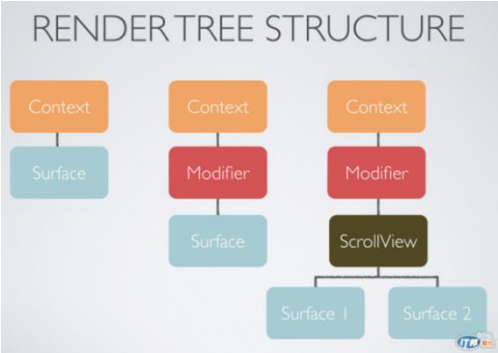

圖 15 Render Tree 結構圖

4.3 Framework7 Framework[10]

Framework7 又稱為 F7，提供原生視覺組件作為開發，符合 iOS 設計標準—扁平化設計，另外也提供 Material 設計標準，其框架主要針對 iOS 進行研究並實作如原生應用程式所應有的操作體驗，舉例來說：iOS 手機若從螢幕左側邊緣向右滑動時，會執行返回前頁指令，在設計上此為 SwipeBack，對於畫面呈現上如圖 16，iphone 手機皆為無實體返回鍵的設計上，此設計能強化使用者操作上的體驗。

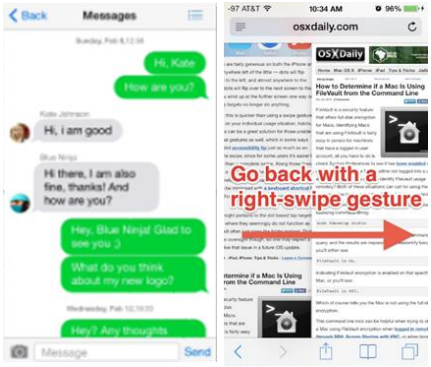

圖 16 Framework7 元件範例圖

五、系統設計與成果

開發行動應用程式，包含使用者介面、商業邏輯層以及本地服務層，本論文著重於使用者介面並搭配部分商業邏輯實驗實際開發成果，基於視覺模式設計準則並設計行動應用程式雛形，分別採用三種視覺框架實作出擬訂介面，探討其成果效益與開發成本。

5-1 系統設計

基於上述產出型應用程式常用之七項視覺模式準則，並以線上購票行動應用程式為例實作簡易雛形，其功能設計(Functional Map)如圖 17，著重視覺模式準則應用，其視覺流程(UI flow)如圖 18，大量視覺模式用於列表之中，可從 Wireframe 圖清楚理解視覺元件結構以及配置(如圖 19)，Wireframe 為低保真原型設計(Lo-Fi Design)，除去所有視覺設計細節之下，進行頁面結構、功能、內容規劃。

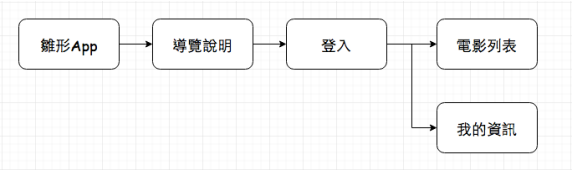

圖 17 系統功能流程圖

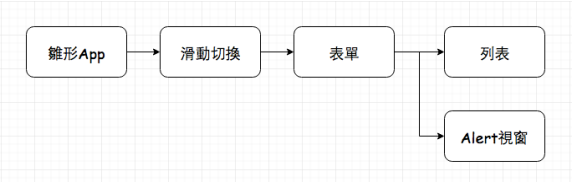

圖 18 系統視覺流程圖

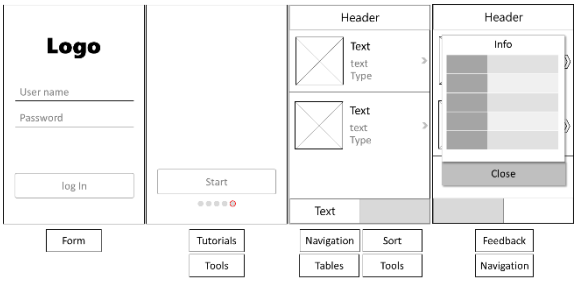

圖 19 系統 Wireframe 圖

5-2 系統成果

5.2.1 Ionic 建置成果

Ionic 框架著重於視覺元件的發展，擁有全套的視覺組件，對於前端元件設計上能更加快速開發，除此之外，Ionic 提供一套基於 AngularJS 的拓展功能，開發者只需要透過簡短程式即可完成複雜功能，如圖 20 為開發者需撰寫之程式、圖 21 為框架協助產生之結構。

```
<ion-list>
  <ion-item ng-repeat="item in items">
    Hello, {{item}}!
  </ion-item>
</ion-list>
```

圖 20 建置列表結構程式

```
▼<ion-list show-delete="data.showDelete" show-reorder="data.showReorder" class="disable-user-behavior">
  ▼<div class="list">
      <!-- ngRepeat: item in items -->
    ▼<ion-item ng-repeat="item in items" item="item" href="#/item/0" class="item-remove-animate item item
    item-left-editable item-right-editable">
      ▶<a class="item-content ng-binding" ng-href="#/item/0" href="#/item/0">…</a>
      ▶<div class="item-left-edit item-delete enable-pointer-events">…</div>
      ▶<div class="item-options invisible">…</div>
      ▶<div data-prevent-scroll="true" class="item-right-edit item-reorder enable-pointer-events">…</div>
      </ion-item>
  </div>
</ion-list>
```

圖 21 執行列表顯示結構程式

若從視覺元件導向進行探討，該框架對於上述之視覺模式皆提供相對應之視覺模組，如下左圖使用 ScrollBox、右圖則是使用 Cards 結構。

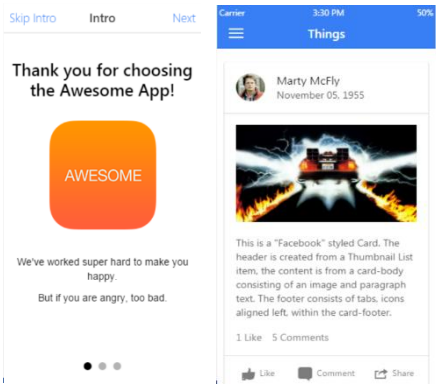

圖 22 Ionic 元件應用範例圖

5.2.2 Famo.us 建置成果

Famo.us 框架著重在於動畫應用，可以設計出 2D、3D 風格，對於視覺元件的支援度較低，且在開發也較煩瑣，例如：導覽模式 Tab 的用法，此框架介面設計大多在於 js 上進行開發，如圖 23，定義一個 Tab 與捲軸，並將捲軸放置 Tab 所屬之畫面內，成果圖如圖 24 所示。

```
var homeSection = myApp.section('home');
homeSection.setOptions({
    title: 'List',
    navigation: {caption: 'List', icon: '<sp
});
var homeItems = new ViewSequence();
var homeScroll = new Scrollview();
homeSection.link(homeScroll);
```

圖 23 Famo.us 建置導覽按鈕程式

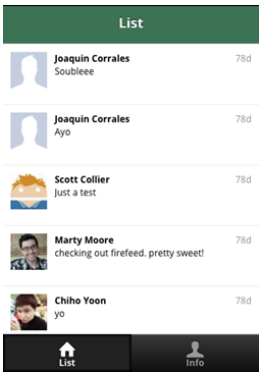

圖 24 Famo.us 建置成果圖

5.2.3 Framework7 建置成果

Framework7 視覺框架元件著重於原生設計風格，因此在呈現上如同原生應用程式，舉例來說，iOS 在設計上傾向扁平化設計(flat design)，扁平化設計放棄了所有 3D 化、維度概念，將畫面完完全全的以平面方式呈現。捨棄陰影、光影、斜角等相對於平面的凹凸效果，狹義一點定義甚至可以說，用色塊來組成視覺介面，如圖 25 左所示。

圖 25 扁平化設計

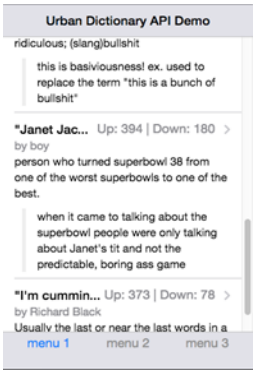

圖 26 F7 建置成果圖

在開發上，此框架詳細描述元件使用方法以及給予即時結果，對於開發者而言，能夠從文件檔案中的說明應用於開發上，整體學習曲線較為平緩，圖 26 為使用 Framework7 開發之成果圖。

六、結論

本研究對於三種框架進行綜合比較，其比較指標依據供給之視覺組件多寡、學習門檻難易度進行比較。

表 1 框架之視覺組件比較表

	Ionic	Famo.us	F7
Navigation			
List	3	1	2
Tabs	3	1	2
Gallery	3		2
Forms			
inputType	3	3	3
Table			
Headerless	3	1	2
Grouped Rows	3	1	2
Search			
SearchBar	3	2	3
Tutorials			
Slider	3		
Tools			
Inline Action	3	2	3
Feedback			
Modal	3	1	2
Popup	3	1	2

由上表可知道，Ionic Framework 在此之中能呈現之效果較佳，其次為 Framework7 Framework，最後才是 Famo.us Framework。其主要原因在於此三種框架所追求的目的些微不同，Ionic 與 Framework7 著重於視覺呈現；而 Famo.us 則是著重於性能流暢度。

表 2 框架之學習門檻比較表

	Ionic	Famo.us	F7
快速入門教學	3	1	2
框架使用文件	3	1	3
範例程式	3	1	3
社群討論度	3	1	1

由上表能知道對於三者框架來說，Ionic Framework 整體的學習門檻較為平緩，開發者在開發時能從中快速理解進而開發；而 Famo.us 官方對於文件的說明並未完善，如具體的範例加快使用者掌握其使用方法等，因此對於其學習曲線而言較為陡峭。

對於開發者而言，透過前端頁面上進行排版在維護或開發皆較能掌握；相對而言，若將視覺結構、程式邏輯皆放置於 JS 檔案則難以維護與開發，因此以上述三項視覺框架而言，採取 Ionic Framework 能夠提升開發效率、降低開發成本，對於後續維運而言也能較容易上手。

參考文獻

[1] Wikipedia, UCD (User-Centered Design), https://en.wikipedia.org/wiki/User-centered_design

[2] Jesse James Garrett, "The Elements of User Experience", 2000.

[3] iPhone Dev Center, "iOS Human Interface Guidelines", https://developer.apple.com/library/ios/documentation/UserExperience/Conceptual/MobileHIG/index.html, 2016.

[4] Theresa Neil, "Mobile Design Pattern Gallery", 2012.

[5] 周陟, "UI 進化論—行動裝置使用者介面設計", 2010.

[6] Bill Scott, Theresa Neil , "Designing Web Interfaces", 2009

[7] Jakob Nielsen, "10 Usability Heuristics for User Interface Design", https://www.nngroup.com/articles/ten-usability-heuristics/, 1995

[8] Drifty, Ionic, http://ionicframework.com/

[9] Steve Newcomb, Famo.us, http://famous.co/

[10] Vladimir Kharlampidi, Framework7, http://framework7.io/