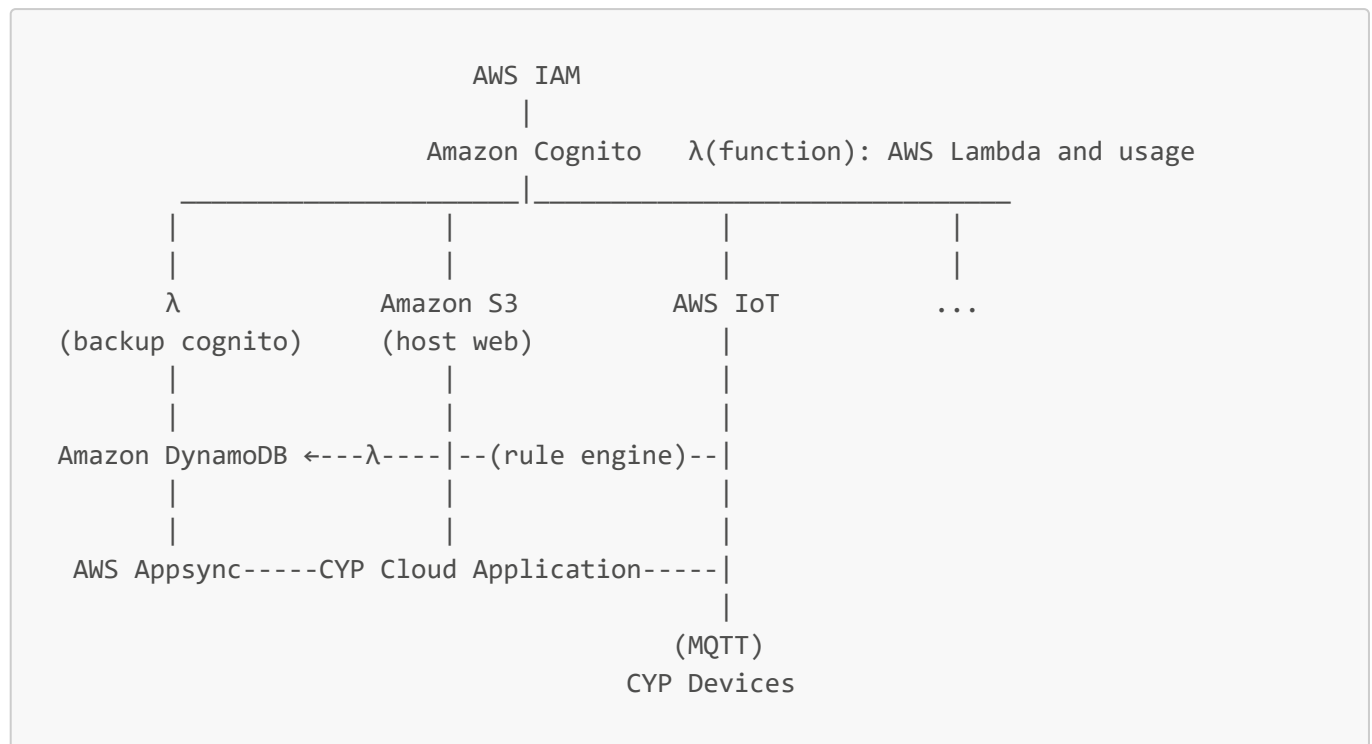


# CYP Cloud Overview



## Techniques

### 1. AWS IAM

**IAM (Identify and Access Management)**為連接所有服務的核心，只有在**IAM**授權過的存取才會在應用中生效。**Access Permission**可以透過**Policy**去制定。

- **[AWS IAM]**

[https://docs.aws.amazon.com/zh\\_tw/IAM/latest/UserGuide/introduction.html](https://docs.aws.amazon.com/zh_tw/IAM/latest/UserGuide/introduction.html)

### 2. Amazon Cognito

**Cognito**透過**User Pool Identity Pool**管理所有應用程式中的使用者。透過使用者池，管理者及開發人員能夠更好的區分不同等級的使用者權限。

- **[Amazon Cognito]**

[https://docs.aws.amazon.com/zh\\_tw/cognito/latest/developerguide/what-is-amazon-cognito.html](https://docs.aws.amazon.com/zh_tw/cognito/latest/developerguide/what-is-amazon-cognito.html)

- **[Serverless-stack create user pool]**

<https://serverless-stack.com/chapters/create-a-cognito-user-pool.html>

- **[Serverless-stack create federated identities pool]** <https://serverless-stack.com/chapters/create-a-cognito-identity-pool.html>

### 3. Amazon S3

**S3**除了提供付費的使用空間供整個Application運用，還能夠對靜態網頁的內容做管理 (*static web hosting*)，使得 **serverless** 的架構更容易實現。

- **[Amazon S3]**  
[https://docs.aws.amazon.com/zh\\_tw/AmazonS3/latest/dev/Welcome.html](https://docs.aws.amazon.com/zh_tw/AmazonS3/latest/dev/Welcome.html)
- **[S3 Static Web Hosting]**  
[https://docs.aws.amazon.com/zh\\_tw/AmazonS3/latest/dev/WebsiteHosting.html](https://docs.aws.amazon.com/zh_tw/AmazonS3/latest/dev/WebsiteHosting.html)

## 4. Amazon DynamoDB

DynamoDB屬於 "非關聯式資料庫"(Non-SQL database) 的一種。傳統的關聯式資料庫具備可靠性、穩定性；非關聯式資料庫則具有容易擴充、彈性的資料結構等優點。Local Secondary Index, LSI 及 Global Secondary Index, GSI的使用也能補足非關聯式資料庫在條件存取的劣勢。

- **[Amazon DynamoDB]**  
[https://docs.aws.amazon.com/zh\\_tw/amazondynamodb/latest/developerguide/Introduction.html](https://docs.aws.amazon.com/zh_tw/amazondynamodb/latest/developerguide/Introduction.html)
- **[DynamoDB Secodary Indexes]**  
<https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/SecondaryIndexes.html>

## 5. AWS IoT

AWS IoT提供了一種透過網路與終端裝置(embedded, mobile, pc, etc.)互動的介面。除了基本的 Message Queuing Telemetry Transport, MQTT訊息傳輸介面外，AWS還在那之上實現了裝置影子(Device Shadow)來掌握即時的控制與狀態更新、以及AWS Greengrass(gateway)來對一群裝置做管理。

- **[AWS IoT Core]**  
[https://docs.aws.amazon.com/zh\\_tw/iot/latest/developerguide/what-is-aws-iot.html](https://docs.aws.amazon.com/zh_tw/iot/latest/developerguide/what-is-aws-iot.html)
- **[AWS IoT Shadow]**  
[https://docs.aws.amazon.com/zh\\_tw/iot/latest/developerguide/iot-device-shadows.html](https://docs.aws.amazon.com/zh_tw/iot/latest/developerguide/iot-device-shadows.html)
- **[AWS IoT Greengrass]**  
[https://docs.aws.amazon.com/zh\\_tw/greengrass/latest/developerguide/what-is-gg.html](https://docs.aws.amazon.com/zh_tw/greengrass/latest/developerguide/what-is-gg.html)

## 6. AWS Appsync

#TODO

## 7. Web Application

第一，由於Amazon S3只提供靜態網頁託管的服務。第二，使用ReactJS開發的網頁應用可以被build成S3能夠託管的內容。ReactJS基於Node.js，Node.js的原身又是Javascript，因此學習使用這三樣東西是必須的。除此之外，Node.js會使用到的套件會使用npm做為套件管理系統，也需要學習如何使用npm來管理每個project的套件。

- **[npm Tutorial]**  
[https://www.w3schools.com/nodejs/nodejs\\_npm.asp](https://www.w3schools.com/nodejs/nodejs_npm.asp)
- **[Javeacript Tutorial]**  
<https://www.w3schools.com/js/>
- **[Node.js Tutorial]**  
<https://www.w3schools.com/nodejs/>
- **[ReactJS Tutorial]**  
<https://www.tutorialspoint.com/reactjs/>

為了要在web上存取AWS的服務，會使用到一些Node.js套件：

- **[AWS SDK for Javascript]**  
<https://docs.aws.amazon.com/AWSJavaScriptSDK/latest/AWS.html>
- **[AWS Amplify on Javascript]**  
<https://aws-amplify.github.io/docs/js/start?platform=purejs>

## Step by Step

---

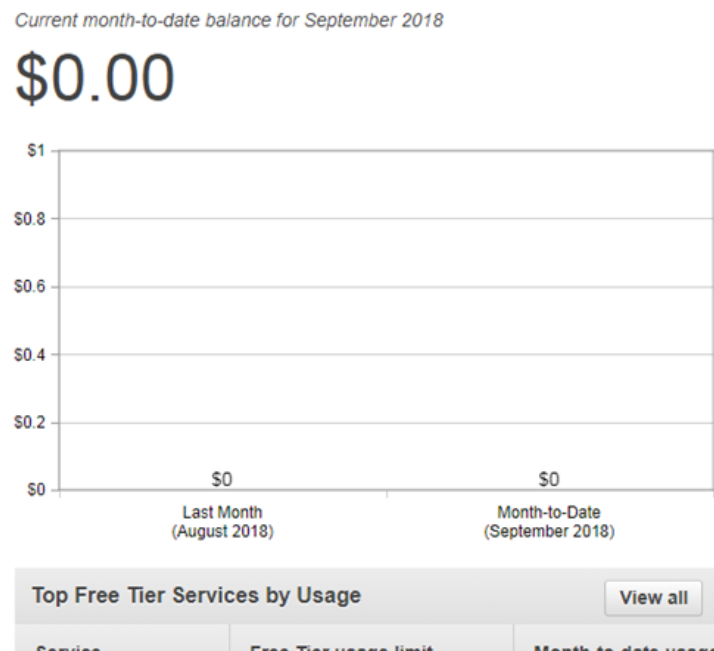
### Step: IAM

#### Basic Rule #1 Protect Your Credential

透過IAM政策提供許可時，僅授予一個對象其最低需要的權限。另外，小心保存您的任何密碼、存取ID、私鑰等等，不要放入任何可以供其他人瀏覽的 "程式碼" 中，像是github、bitbucket等等。舉例：

```
server = None
server_init(server, 'my_server')
server_init_auth(server, 'my_account', 'my_password1234') # dangerous
```

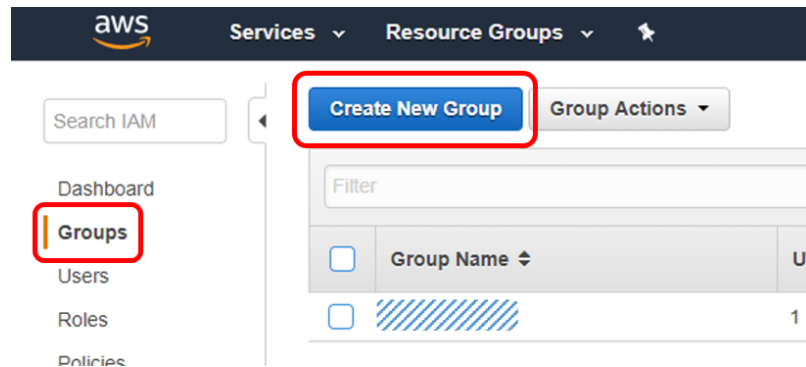
#### Basic Rule #2 Keep an Eye on Resource Usage



"Pay as you go"是AWS的基本原則，釋放任何你不再需要用到的資源，並且到帳單儀表板檢視您的預估帳單。AWS也提供了 [Monthly Calculator](<http://calculator.s3.amazonaws.com/index.html>)做為付費計算工具。

#### - Getting Start

1. 在IAM Console左方導覽列找到「Group」→「Create New Group」
2. 輸入group name→「Next Step」



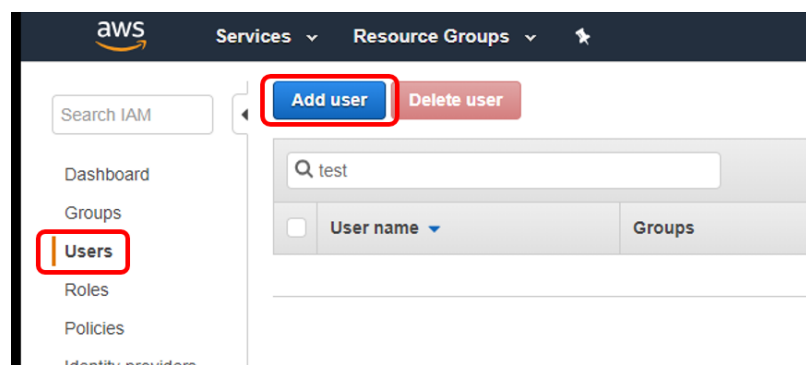
3. Attach policy (permission) 屬於這個「群」的「使用者」都會擁有policy定義的權限。有許多預設policy可以選擇，但也能自訂一個policy

### Attach Policy

Select one or more policies to attach. Each group can have up to 10 policies attached.


Filter: Policy Type <input type="text" value="EC2"/>		
	Policy Name	Attached Entities
<input type="checkbox"/>	AmazonEC2ContainerRegistry...	0
<input checked="" type="checkbox"/>	AmazonEC2ContainerRegistry...	0
<input type="checkbox"/>	AmazonEC2ContainerRegistry...	0
<input checked="" type="checkbox"/>	AmazonEC2ContainerService...	0
<input checked="" type="checkbox"/>	AmazonEC2ContainerService...	0
<input type="checkbox"/>	AmazonEC2ContainerServiceF...	0
<input checked="" type="checkbox"/>	AmazonEC2ContainerServiceF...	0


4. 「Next Step」→「Create Group」  
 5. 在IAM Console左方導覽列找到「Users」→「Add user」



6. 輸入user name、access type選"Programmatic access"→「Next: Permission」 這個選項代表 "此user將會透過程式碼來存取服務"  
 7. 「Add user to a group」→選擇剛才创建的group (將group的權限套用到新user上)→「Next: Review」

### Set permissions

 Add user to group

 Copy permissions from existing user

Add user to an existing group or create a new one. Using groups is a best-practice.

### Add user to group

Create group
Refresh

	Group	Attaches to
<input checked="" type="checkbox"/>	testGroup	Amazon

8. 「Final check」→「Create User」

9. **重要:** 最後是一個可供下載 *Access key ID* 和 *Secret access key* 的頁面，當離開這個頁面後就再也無法下載，請妥善保存。

Services
Resource Groups

### Add user

**Success**

You successfully created the users shown below. You can view an instructions for signing in to the AWS Management Console. This you can create new credentials at any time.

Users with AWS Management Console access can sign-in at: <http://>

Download .csv

	User	Access key
▶ <input checked="" type="checkbox"/>	testUser	AKIAIEJX

Back
English (US)

## Step: Cognito

Amazon Cognito提供了 **Cognito User Pool**, 使用者集區 和 **Cognito Federated Identity Pool**, 聯合身分集區。簡單來說，user pool用來管理使用者的基本資料，例如登入用的使用者名稱、帳號密碼，或是電話號碼等等；而每個使用者的真正身分是在identity pool管理，這決定了某個使用者擁有哪些服務的受限存取權。

更進一步解釋的話，user pool是扮演一個使用者 "提供者"(provider) 的角色，而 federated identity pool則是將所有的 provider 提供的 user 做聯合的身分管理。因此，我們不僅能在AWS增加新使用者，還能透過第三方提供者，例如 Facebook、Google+ 等，來增加服務的使用者，就是基於這個機制。

### - Getting start

1. 進入Amazon Cognito console → 「Manage User Pools」 → 「Create a user pool」 → 輸入Pool name，選擇「Review defaults」

2. 在左方選單選擇「Attributes」→ 勾選「Email address or phone number」→ 開啟「Allow email addresses」讓使用者可以用email來驗證和登入

3. 在左方選單選擇「Review」→ 「Create pool」

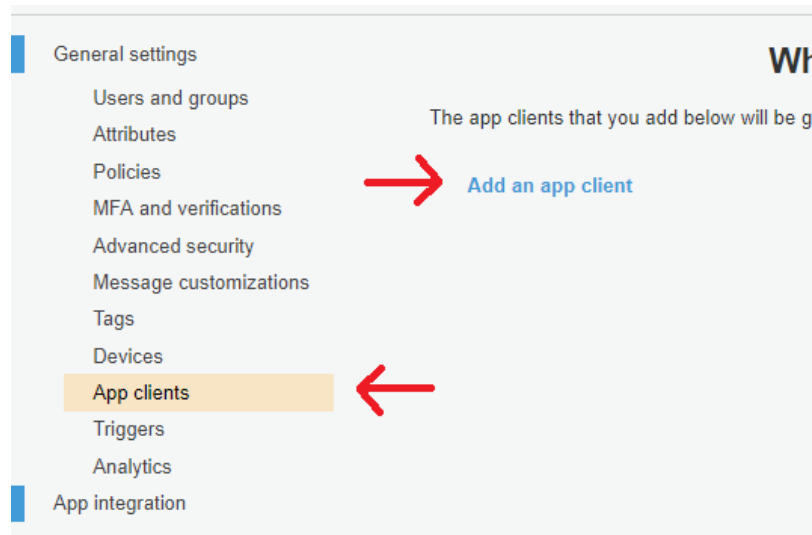
其他設定:

Panel	Function
Attributes	<ul style="list-style-type: none"> <li>• <b>Username</b>: 決定可以用什麼個人資料登入，例如電子郵件、電話、自定等等</li> <li>• <b>Standard Attributes</b>: 決定註冊時需要哪些使用者基本資料(標準)</li> <li>• <b>Custom Attributes</b>: 自訂所需要的註冊基本資料</li> </ul>
Policies	<ul style="list-style-type: none"> <li>• <b>Password Policy</b>: 定義密碼強度需求</li> <li>• <b>Allow Sign-up</b>: 使用者能自行註冊或只允許管理這增加使用者</li> <li>• <b>Expiration</b>: 定義由管理員創建的帳號多久會過期</li> </ul>

Panel	Function
MFA and Verification	<ul style="list-style-type: none"> <li>• <b>Enable/Disable</b>: 開啟/關閉多方驗證功能</li> <li>• <b>Verification Attributes</b>: 決定驗證手段</li> <li>• <b>Verification Role</b>: 定義執行驗證動作的執行角色</li> </ul>
Message customizations	<ul style="list-style-type: none"> <li>• <b>Verification message</b>: 自訂郵件驗證訊息</li> <li>• <b>Invitation message</b>: 自訂郵件邀請訊息</li> </ul>
Tags	在使用者集區加入標籤
Devices	決定是否儲存使用者存取服務的裝置
App clients	決定哪種應用客戶端可以存取此使用者集區
Triggers	<ul style="list-style-type: none"> <li>• <b>Pre Sign-up</b>: 使用 Lambda function 自訂註冊工作流程</li> <li>• <b>Pre Authentication</b>: 使用 Lambda function 自訂認證(登入)工作流程</li> <li>• <b>Custom Message</b>: 使用 Lambda function 自訂校驗或多方驗證訊息</li> <li>• <b>Post Authentication</b>: 使用 Lambda function 自訂登入後的工作流程</li> <li>• <b>Post Confirmation</b>: 使用 Lambda function 自訂校驗使用者後的工作流程</li> <li>• <b>Define Auth Challenge</b>: 使用 Lambda function 初始化客製的認證流程</li> <li>• <b>Create Auth Challenge</b>: 使用 Lambda function 創建認證時的 challenge, 會在 "Define Auth Challenge" 後被呼叫</li> <li>• <b>Verify Auth Challenge Response</b>: 使用 Lambda function 校驗使用者對 challenge 的回應</li> <li>• <b>User Migration</b>: 使用 Lambda function 自訂登入或忘記密碼時的工作流程</li> <li>• <b>Pre Token Generation</b>: 使用 Lambda function 自訂權杖的產生流程</li> </ul>

4. 創建完成後頁面會自動跳轉到該集區的一般設定頁面，也能在Amazon Cognito主控台點選指定集區進入，並記住集區的「Pool id」

5. 在設定介面左方「General settings」中選擇「App client」→「Add an app client」



- 輸入「App client name」和「Refresh token expiration(days)」。
- 取消勾選「Generate client secret」，因為使用Javascript SDK不支援這種 client secret
- 勾選「Enable sign-in API for server-based authentication (ADMIN\_NO\_SRP\_AUTH)」，這讓我們可以通過 AWS CLI 管理使用者集區

**App client name**

Required

**Refresh token expiration (days)**

30

☐ Generate client secret

☒ Enable sign-in API for server-based authentication (ADMIN\_NO\_SRP\_AUTH) [Learn more.](#)

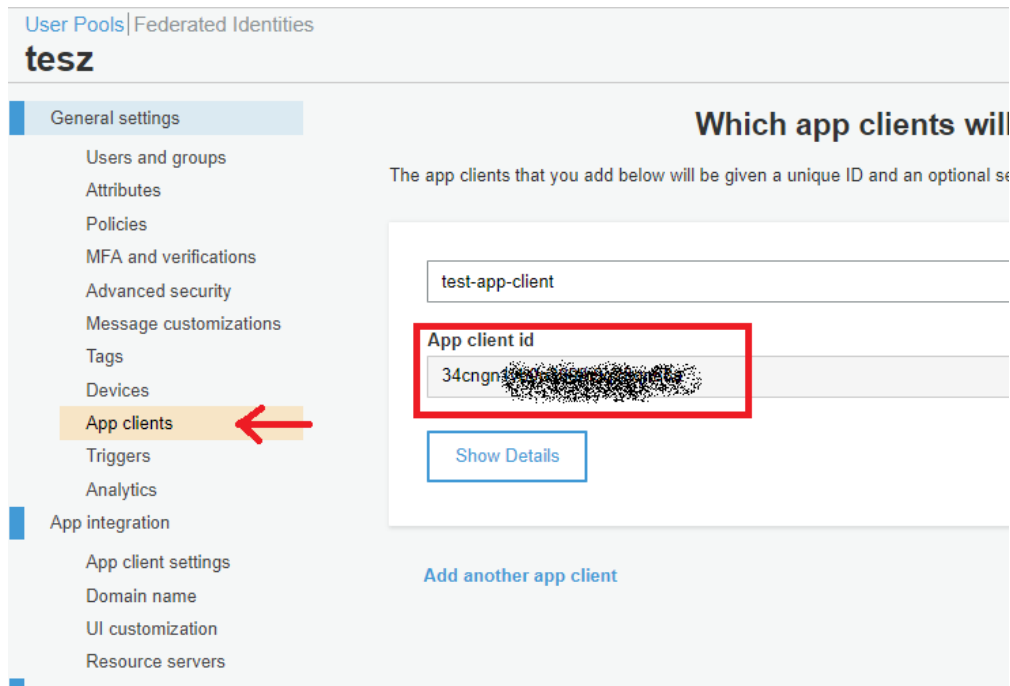
☐ Only allow Custom Authentication (CUSTOM\_AUTH\_FLOW\_ONLY) [Learn more.](#)

☐ Enable username-password (non-SRP) flow for app-based authentication (USER\_PASSWORD\_AUTH) [Learn more.](#)

[Set attribute read and write permissions](#)

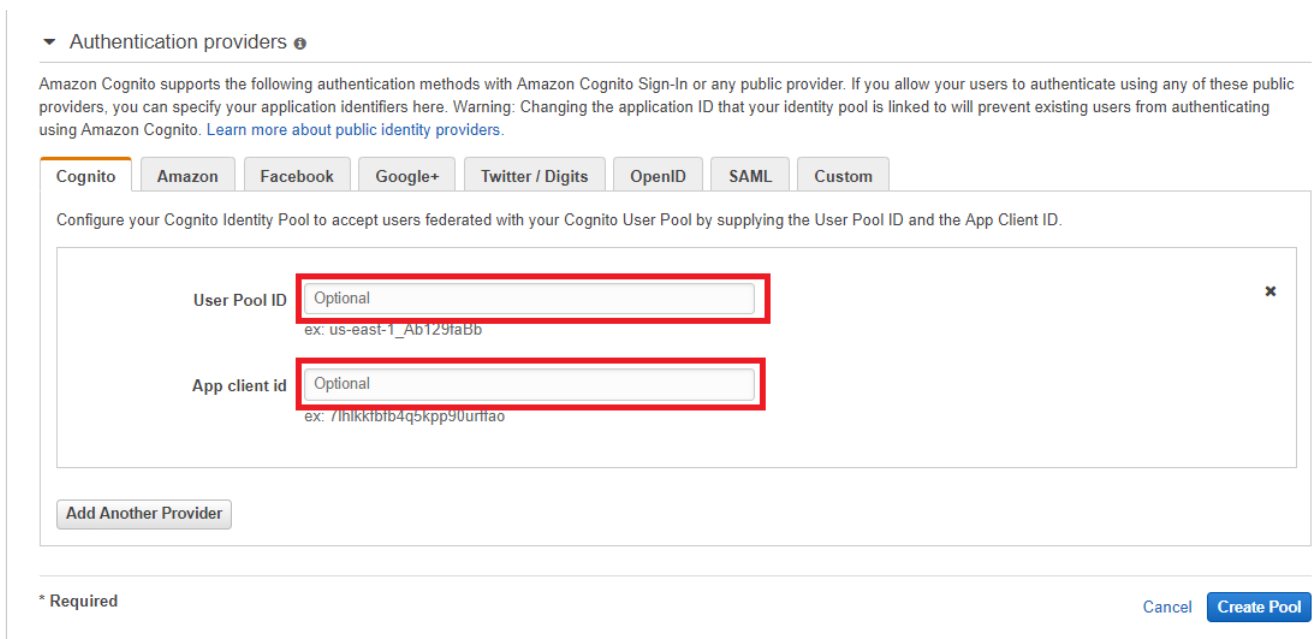
- 點擊「Create app client」並記住「App client id」





10. 進入Amazon Cognito console → Manage Federated Identities→「Create new identity pool」→ 輸入 Identity pool name

11. 在下方「Authentication Providers」區塊中選擇「Cognito」分頁，並輸入新創立的 User pool id 和 App client id



12. 點選「Create Pool」後會跳轉到 one-click role 頁面，展開「Hide Details」→「View Policy Document」可以檢視和編輯"驗證通過"和"驗證失敗"時將會套用的角色(政策) → 點選「Allow」產生這些角色

▼ Hide Details

Role Summary ?

Role Description Your authenticated identities would like access to Cognito.

IAM Role Create a new IAM Role

Role Name Cognito\_testfederatedpoolAuth\_Role

▶ View Policy Document

Role Summary ?

Role Description Your unauthenticated identities would like access to Cognito.

IAM Role Create a new IAM Role

Role Name Cognito\_testfederatedpoolUnauth\_Role

▶ View Policy Document

## Step: ReactJS

在不使用ReactJS的情況下，要建立整個網頁應用程式需要兩個部分，**server(back-end)** 以及 **web page(front-end)**。Programmer編寫網頁內容的同時，也要使用script(javascript)來和伺服器(或任何外部資源)互動。

ReactJS則是結合了這兩者，伺服器直接渲染出DOM(Document Object Model)，簡單來說就是 伺服器直接提供html內容 給使用者閱讀。當然，不使用ReactJS也能使伺服器直接提供內容，像這樣：

```
var AppendStr='';

AppendStr+='<div class="posit switch-fiel"
style="top:10px;left:200px;width:400px;"><fieldset class="setfieldset"
style="height:200px;"><legend>Login</legend>';
AppendStr+='<div id="msg" class="posit fontsize18 fontwidth700"
style="width:380px;top:30px;left:10px;color:#FF1C1C;"></div>';
// ...
$('#body').html(AppendStr);
```

ReactJS則是提供了較容易開發的套件給網頁開發人員。

```
import react, { Component } from 'react';

class App extends Component {
  render() {
    return (
      <div className="posit switch-fiel">
        <fieldset>
          {...}
        </fieldset>
      </div>
    );
  }
}
```

```
    }  
  }  
  ReactDOM.render(<App />, document.getElementById('root'));
```