MEAN Stack實作班03

上課資源

講義

https://goo.gl/J1NUSP

檔案

https://www.dropbox.com/sh/mnxu5fosuhjpug0/AAAdrOWn8dcVPP5QsYCK7vdpa/1210?dl=0

cloud9

https://ide.c9.io/kuolun/meanstackcourse

今日課程內容

- FB 驗證及登入
- 建立相關購物車及user相關的route
- Stripe API
- EJS

Authentication

- HTTP stateless特性無法實作authentication
- 利用session

概念：

在request上存放關於這個user的一些資訊，加密後一起送到server

server上的passport收到後，解開去看裡面資訊，判斷是哪個user

提供該user可以看的資料回傳response

DEMO

使用者登入可看到secret頁面

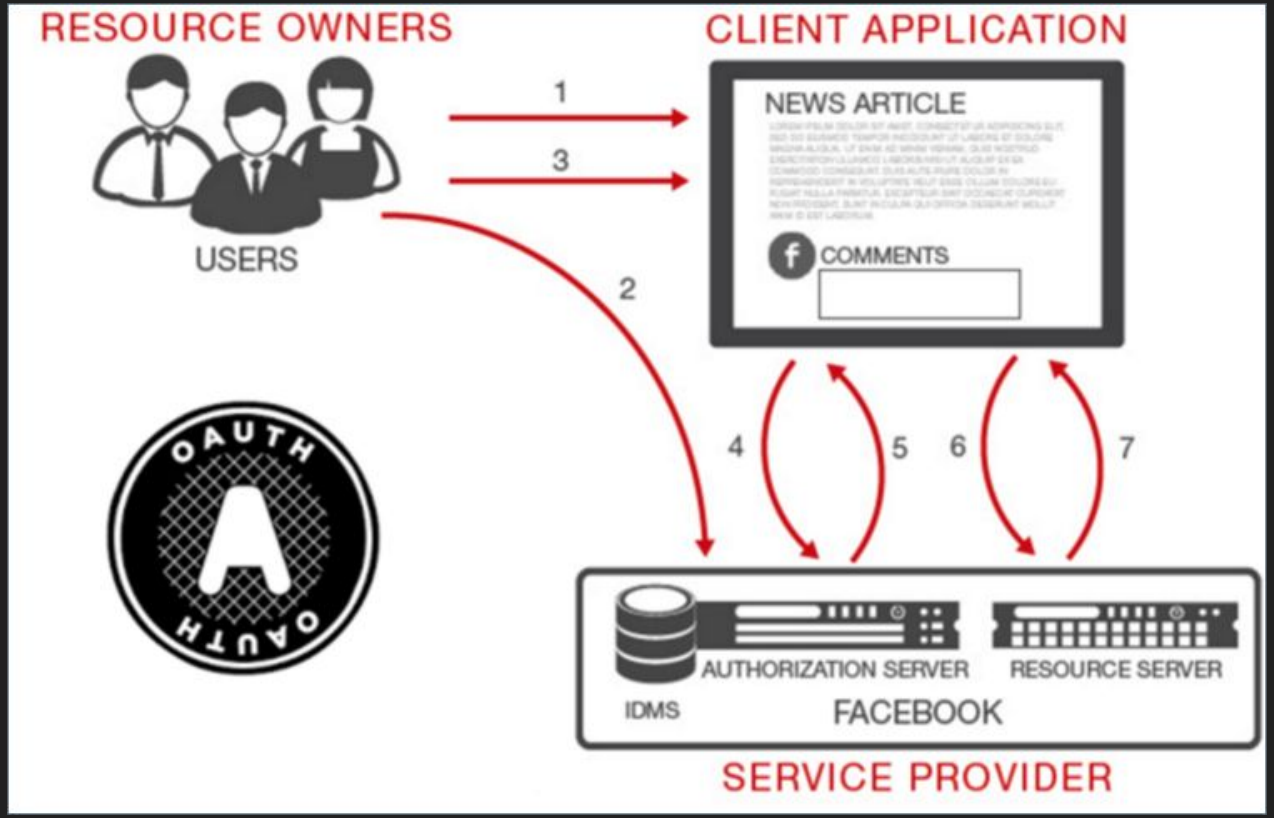
沒登入看不到

Facebook OAuth

OAuth：第三方授權的標準

- 讓第三方服務能讀取用戶資料
- 當我們開發Web App想要透過FB授權，會將網頁導到FB的OAuth介面，讓使用者同意授權，接著導回我們的網站繼續操作

驗證流程與token


RESOURCE OWNERS
CLIENT APPLICATION
USERS
NEWS ARTICLE
COMMENTS
1
2
3
4
5
6
7
OAUTH
AUTHORIZATION SERVER
RESOURCE SERVER
IDMS
FACEBOOK
SERVICE PROVIDER

問題是...

每個網站服務如FB，Google，Twitter的OAuth 授權API都不同

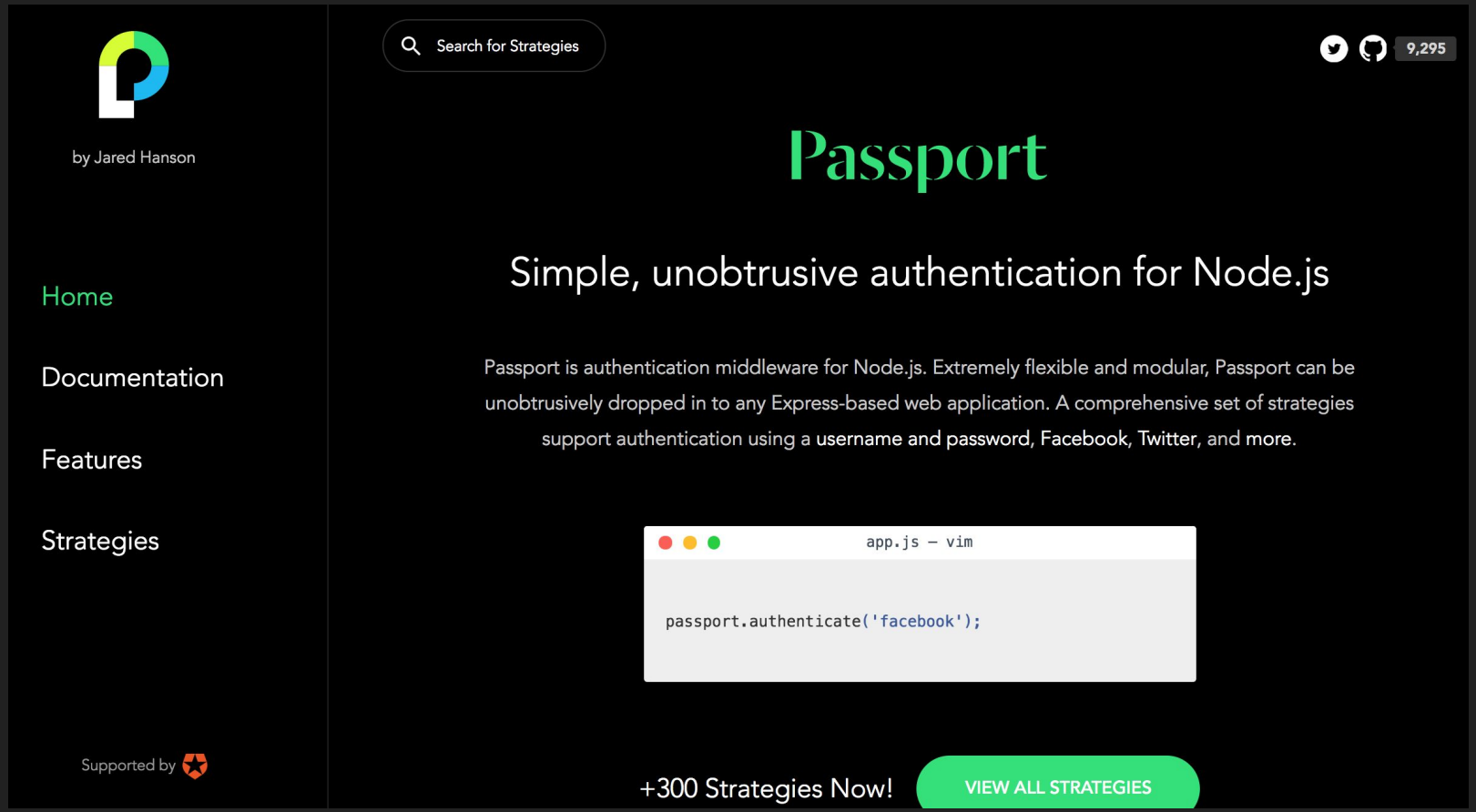
自己一一處理很累

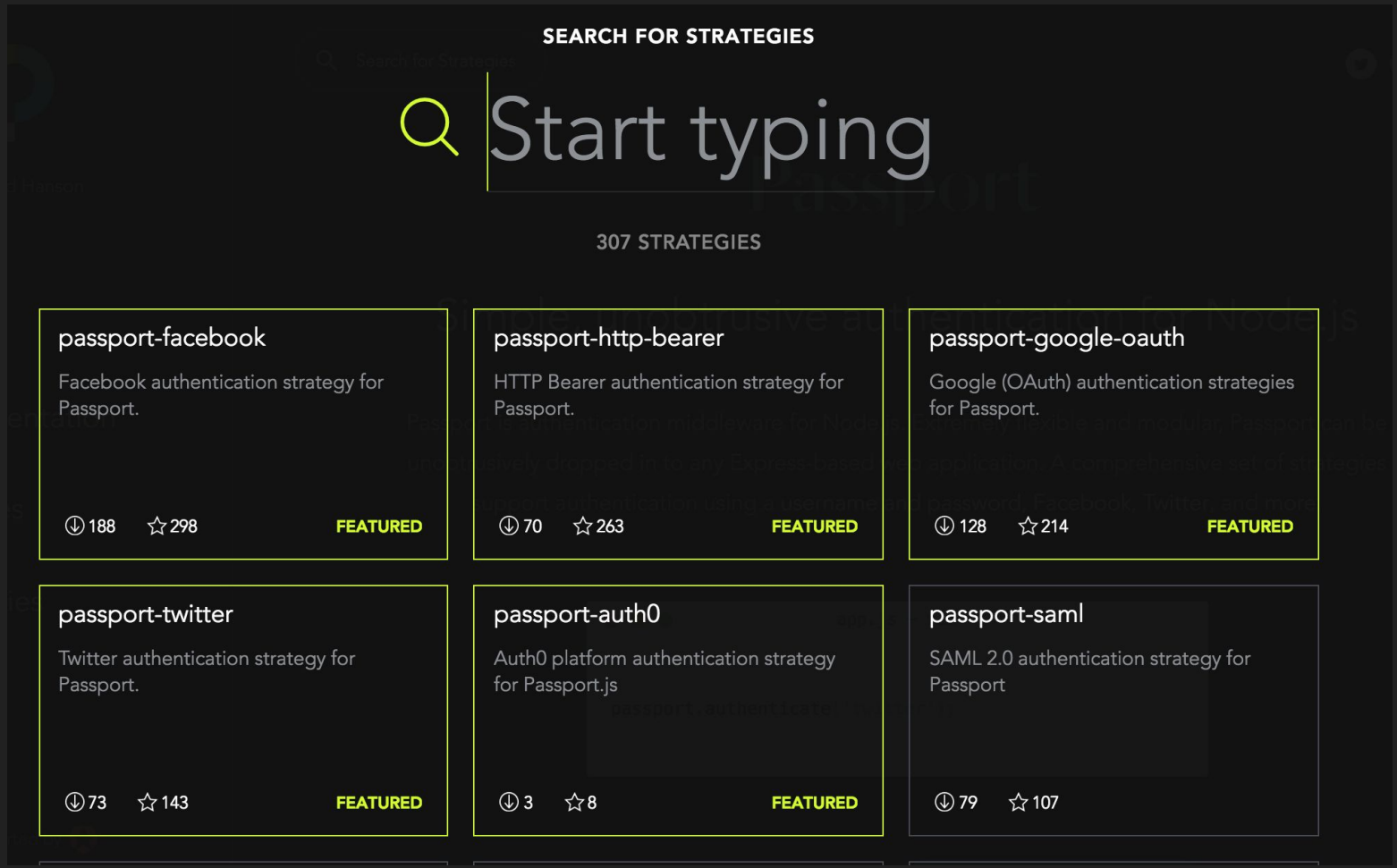
所以有人開發了 Passport

一個做第三方認證登入的程式框架

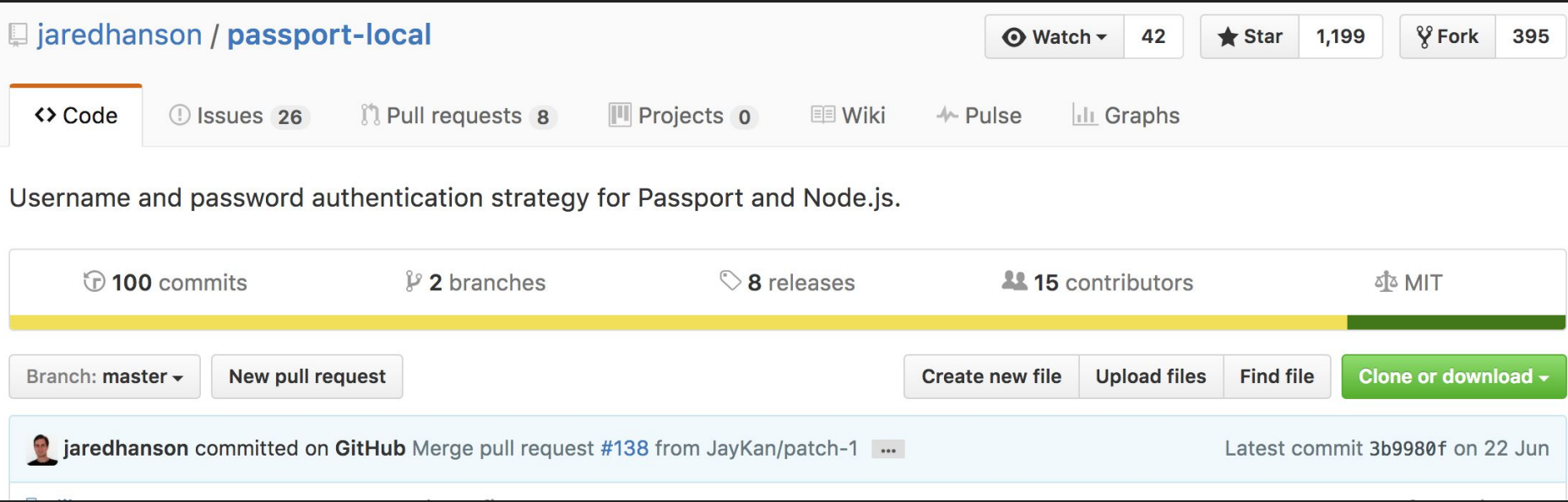
將各大網站的OAuth進行整合包裝成一個module

http://passportjs.org/




SEARCH FOR STRATEGIES
Start typing
307 STRATEGIES
passport-facebook
Facebook authentication strategy for Passport.
188
298
FEATURED
passport-http-bearer
HTTP Bearer authentication strategy for Passport.
70
263
FEATURED
passport-google-oauth
Google (OAuth) authentication strategies for Passport.
128
214
FEATURED
passport-twitter
Twitter authentication strategy for Passport.
73
143
FEATURED
passport-auth0
Auth0 platform authentication strategy for Passport.js
3
8
FEATURED
passport-saml
SAML 2.0 authentication strategy for Passport
79
107

Local Strategy



Passport

- 用途：驗證request
- 容易整合到application
- 認證機制：strategies，支援多種平台如twitter,google

```
app.post('/login', passport.authenticate('local', { successRedirect: '/',
                                                    failureRedirect: '/login' }));
```

Strategies

- 驗證username跟password
- 透過OAuth或OpenID來驗證

Authenticate(驗證)

- 驗證request：passport.authenticate( )並指定要用的strategy

```
app.post('/login',
  passport.authenticate('local'),
  function(req, res) {
    // If this function gets called, authentication was successful.
    // `req.user` contains the authenticated user.
    res.redirect('/users/' + req.user.username);
  });
```

req.user

```
passport.authenticate('local'),
```

- 驗證失敗，Passport會回應401 Unauthorized
  其他route handler不會被呼叫

- 如果驗證成功，handler會被呼叫，且req.user會被passport設定為驗證過的user物件

Verify Callback

目的:找出具有credentials(帳密)的user

以local strategy為例

用passport.use設定strategy

```
passport.use(new LocalStrategy(
  function(username, password, done) {
    User.findOne({ username: username }, function (err, user) {
      1 if (err) { return done(err); }
      2 if (!user) {
          return done(null, false, { message: 'Incorrect username.' });
        }
      3 if (!user.validPassword(password)) {
          return done(null, false, { message: 'Incorrect password.' });
        }
      4 return done(null, user);
    });
  }
```

如果Server exception發生
(如DB無法取用)
傳入err給done

```
return done(err);
```

如果驗證通過, verify callback
會呼叫done function
提供驗證過的user給Passport

```
return done(null, user);
```

驗證失敗(如密碼錯誤)
則要傳入false取代

```
return done(null, false);
```

Session(複習)

如果login request驗證成功,
session會被建立且透過cookie來管理

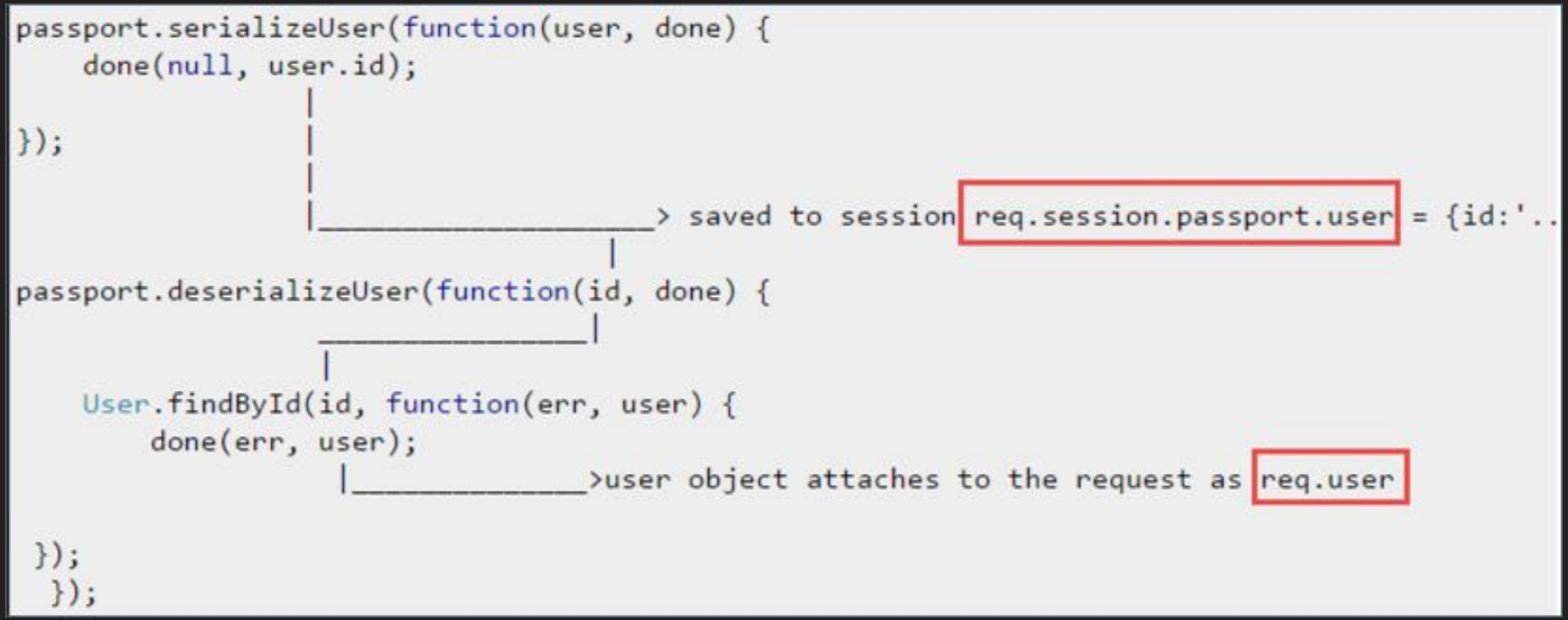
後續的request可由cookie來辨別
為了支援login session,

Passport會**serialize** user instance到session
及
從session **deserialize**出 user instance

```
passport.serializeUser(function(user, done) {
  done(null, user.id);
});

passport.deserializeUser(function(id, done) {
  User.findById(id, function(err, user) {
    done(err, user);
  });
});
```

示意圖


passport.serializeUser(function(user, done) {
done(null, user.id);
});
saved to session req.session.passport.user = {id:'..
passport.deserializeUser(function(id, done) {
User.findById(id, function(err, user) {
done(err, user);
user object attaches to the request as req.user
});
});

實作FB Authentication

FB認證使用OAuth 2.0

$ npm install passport-facebook

- 要先到Facebook Developers網頁建立一個app，會拿到App ID跟AppSecret

- redirect URL：FB 驗證後會把user導到哪裡

FB Developer-建立一個新的APP

https://developers.facebook.com/docs/graph-api?locale=zh_TW

建立新的應用程式編號

開始將 Facebook 整合到你的應用程式或網站

顯示名稱

MEAN ShopApp

否　這是其他應用程式的測試版本嗎？瞭解詳情。

聯絡電子郵件

kuolun@gmail.com

類別

商業類

一旦繼續，就代表你同意 Facebook 開放平台政策

取消　建立應用程式編號

Product Setup

Facebook 登入

世界第一社交登入產品。

開始使用

行動廣告聯播網

使用來自 300 萬位 Facebook 廣告主的原生廣告，透過行動應用程式或網站賺取獲利。

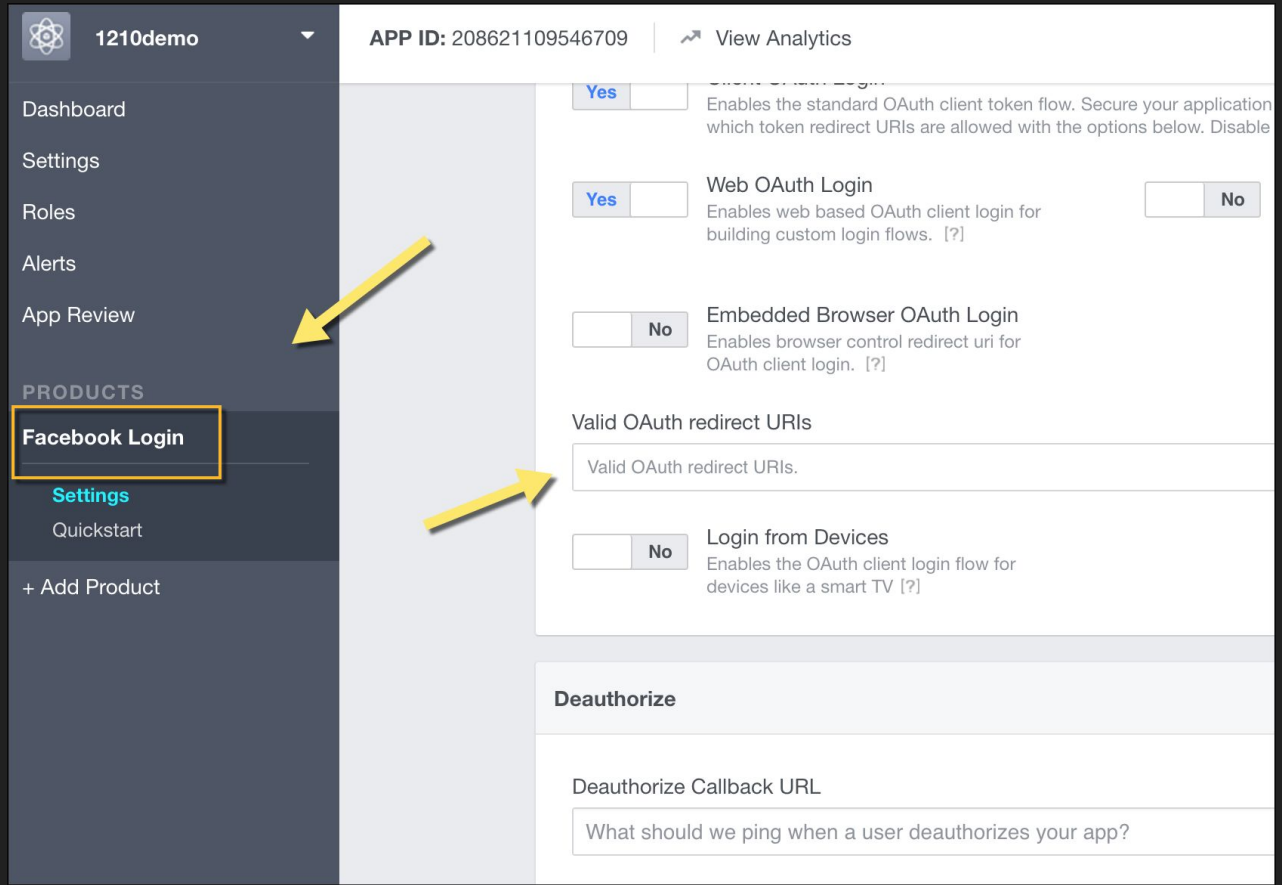
開始使用

Account Kit

順利建立帳號，無須使用密碼。

開始使用

填入http://localhost:3000/auth/facebook/callback



是

用戶端 OAuth 登入

啟用標準 OAuth 用戶端權杖流程。透過以下選項來鎖定允許哪些權杖重新導向 URI，可保護應用程式安全並防止濫用。如果不使用，則可以全域停用。 [?]

是

網路 OAuth 登入

針對建立自訂登入流程啟用網頁型 OAuth 用戶端登入。 [?]

否

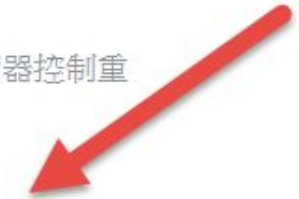
強制網路 OAuth 重新驗證

開啟時，會提醒用戶輸入 Facebook 密碼才能登入網頁。 [?]

否

嵌入的瀏覽器 OAuth 登入

啟用 OAuth 用戶端登入的瀏覽器控制重新導向 URI。 [?]



有效的 OAuth 重新導向 URI

http://localhost:3000/auth/facebook/callback ×

否

從裝置登入

針對智慧型電視等裝置啟用 OAuth 用戶端登入流程 [?]

Callback URL

設定callback URL為

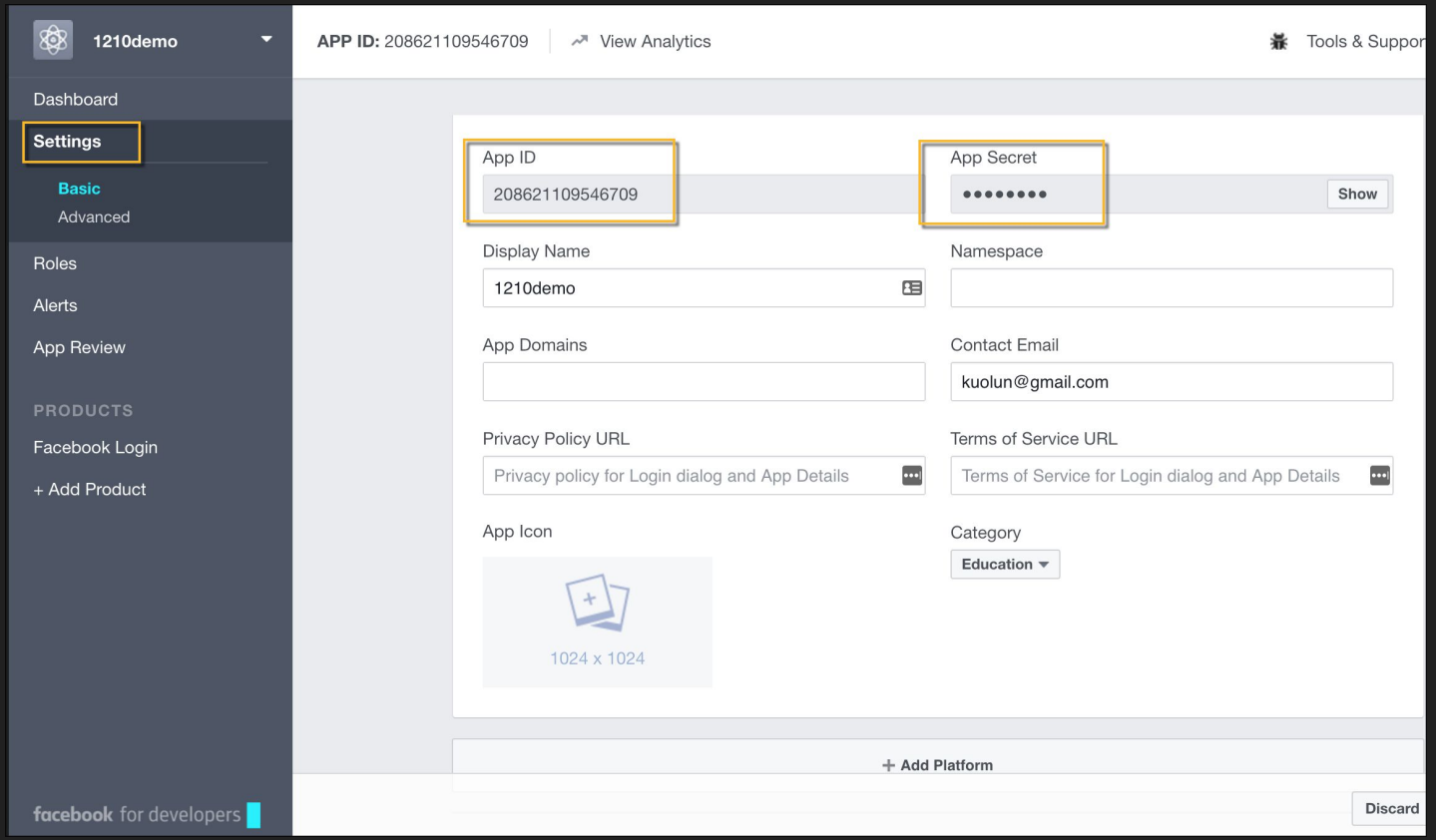
http://localhost:3000/auth/facebook/callback

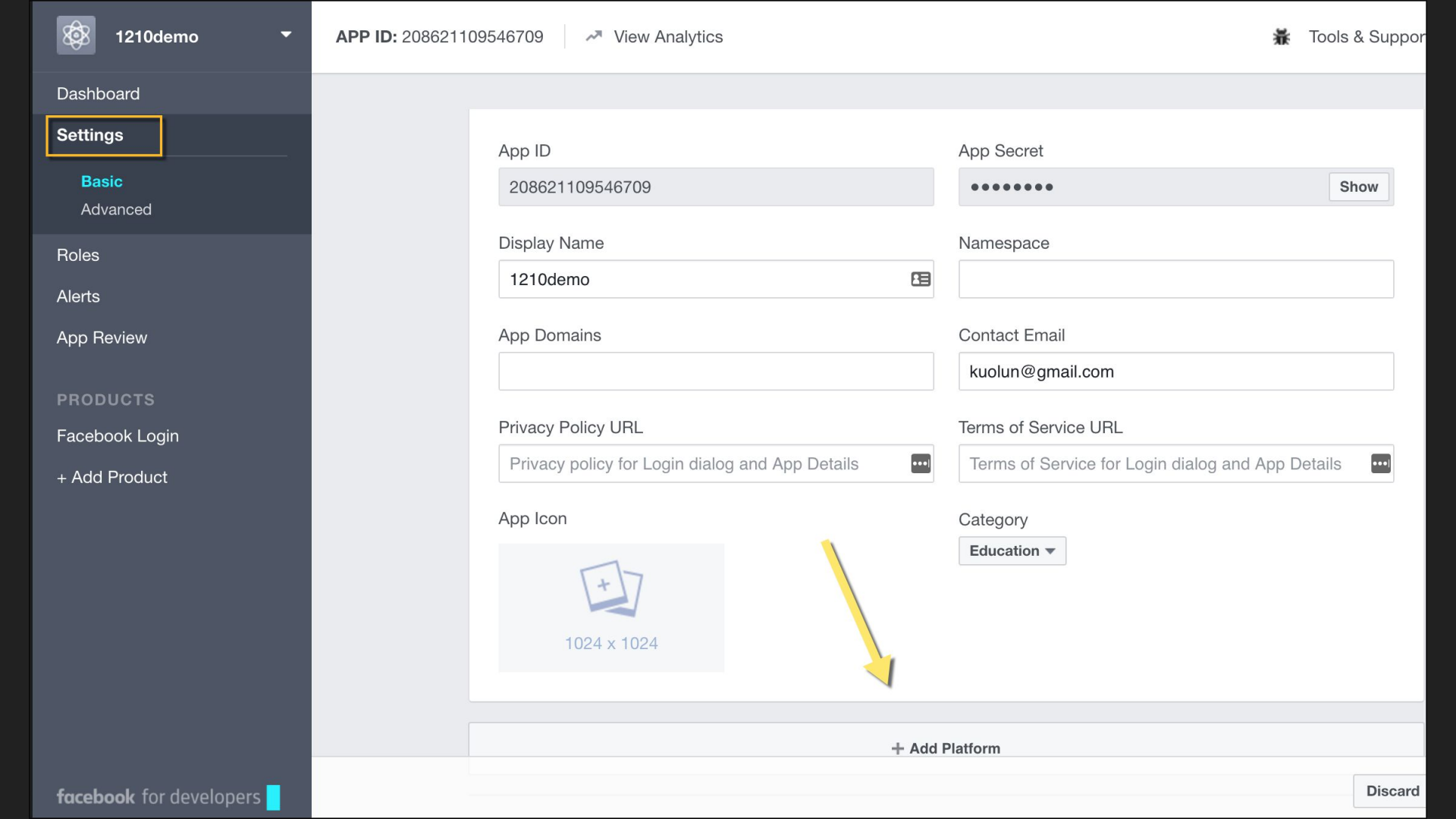
確保app透過FB認證後會redirect users 回我們的app

callback是我們要儲存user information的地方

記錄App ID 跟 App Secret

先copy出來




1210demo
APP ID: 208621109546709
View Analytics
Tools & Suppor
Dashboard
Settings
Basic
Advanced
Roles
Alerts
App Review
PRODUCTS
Facebook Login
+ Add Product
App ID
208621109546709
App Secret
Show
Display Name
1210demo
Namespace
App Domains
Contact Email
kuolun@gmail.com
Privacy Policy URL
Privacy policy for Login dialog and App Details
Terms of Service URL
Terms of Service for Login dialog and App Details
App Icon
1024 x 1024
Category
Education
+ Add Platform
Discard
facebook for developers

應用程式網域

聯絡電子郵件

kuolun@gmail.com

隱私政策網址

登入對話方塊和應用程式詳細資料的隱私政策

服務條款網址

登入對話方塊和應用程式詳細資料的服務條款

應用程式圖示

1024 x 1024

類別

商業類

+ 新增平台

選擇平台

Facebook Canvas

網站

iOS

Android

Windows 應用程式

粉絲專頁頁籤

Xbox

PlayStation


應用程式網域
localhost
聯絡電子郵件
kuolun@gmail.com
隱私政策網址
登入對話方塊和應用程式詳細資料的隱私政策
服務條款網址
登入對話方塊和應
應用程式圖示
1024 x 1024
類別
商業類
網站
網站網址
http://localhost:3000/

發佈APP

Live vs Sandbox

- Sandbox mode：只有你能登入application
- Live mode：大家都可以登入你的application

FB 驗證畫面

練習 : 10分鐘
新增一個 FB APP

Passport

passport.js

- 配置Strategy：
  local, facebook, twitter, and google

- 儲存User到session：
  serializeUser and deserializeUser functions

相關檔案 (v3)

建立FB APP、APP ID及Secret	config/auth.js
配置Passport FB Strategy	config/passport.js
產生Routes	app/routes.js

config/auth.js　　FB認證設定

```
// expose our config directly to our application using module.exports
module.exports = {

    'facebookAuth': {
        // your App ID
        'clientID': '1035056279864779',
        // your App Secret
        'clientSecret': '7c35c679194c13abc9b381f7fa5f5f69',
        'callbackURL': 'http://localhost:3000/auth/facebook/callback'
    },
    // 作業
    'googleAuth': {
        'clientID': 'your-secret-clientID-here',
        'clientSecret': 'your-client-secret-here',
        'callbackURL': 'http://localhost:3000/auth/google/callback'
    }

};
```

config/passport.js 載入相關檔案

npm install passport-facebook passport --save

```
// 載入需要的module
var FacebookStrategy = require('passport-facebook').Strategy;


//載入User model
var User = require('../app/models/user');

//載入auth variables
var configAuth = require('./auth');
```

config/passport.js 序列化user id for session

```
// expose 這個function
module.exports = function(passport) {

    // 用來serialize user.id給session
    passport.serializeUser(function(user, done) {
        done(null, user.id);
    });

    // used to deserialize the user
    passport.deserializeUser(function(id, done) {
        User.findById(id, function(err, user) {
            done(err, user);
        });
    });
```

config/passport.js

new 一個FacebookStrategy

```
passport.use(
    new FacebookStrategy({
            // 從auth.js抓app id跟 app secret
            clientID: configAuth.facebookAuth.clientID,
            clientSecret: configAuth.facebookAuth.clientSecret,
            callbackURL: configAuth.facebookAuth.callbackURL,
            profileFields: ['id', 'emails', 'displayName']
        },
```

如何取得user profile的特定欄位？

回傳的FB profile會包有很多user資訊，但預設不是所有欄位都會回傳

如果app有需要特定欄位可在這邊指定

```
new FacebookStrategy({
        // 從auth.js抓app id跟 app secret
        clientID: configAuth.facebookAuth.clientID,
        clientSecret: configAuth.facebookAuth.clientSecret,
        callbackURL: configAuth.facebookAuth.callbackURL,
        profileFields: ['id', 'emails', 'displayName']
    },
```

config/passport.js

設定verify callback

```
function(token, refreshToken, profile, done) {
    // find the user in the database based on their facebook id
    User.findOne({
        'facebook': profile.id
    }, function(err, user) {···
    });
})
```

設定要存到DB的FB資訊

```
// if the user is found, then log them in
if (user) {
    return done(null, user); // user found, return that user
} else {
    // if there is no user found with that facebook id, create them
    var newUser = new User();
    console.log(profile);
    //profile是FB回傳的資訊
    // set all of the facebook information in our user model
    // set the users facebook id
    newUser.facebook = profile.id;
    newUser.token = token;
    // look at the passport user profile to see how names are returned
    newUser.profile.username = profile.displayName;
    // facebook can return multiple emails so we'll take the first
    newUser.email = profile.emails[0].value;
    newUser.profile.picture = 'https://graph.facebook.com/' + profile.id + '/picture?width=10';
    // save our user to the database
    newUser.save(function(err) {···
    });
}
```

Profile

Verify Callback 會回傳 user profile 資訊
(Facebook, Twitter, and Google傳回方式不同)

Passport 標準化回傳的 profile object.

更多關於Passport如何format **user profile**
http://passportjs.org/docs/profile

新增一筆user document (設定過FB資訊)

存成功的話，也要傳入到done
(passport會去實作done)

```
newUser.save(function(err) {
    if (err) {
        console.log('save error');
        throw err;
    }
    // if successful, return the new user
    return done(null, newUser);
});
```

設定相關FB驗證相關routes

app/routes.js

增加FB 相關3個route

/auth/facebook

/auth/facebook/callback

/logout

```
app.get('/auth/facebook', passport.authenticate('facebook', {
    scope: 'email'
}));
```

```
app.get('/auth/facebook/callback',
    passport.authenticate('facebook', {
        failureRedirect: '/login'
    }),
    function(req, res) {
        // Successful authentication, redirect home.
        res.redirect('/');
    });
```

```
app.get('/logout', function(req, res) {
    // provided by passport.
    req.logout();
    res.redirect('/');
});
```

如何跟User要求額外權限？

到scope設定

```
app.get('/auth/facebook', passport.authenticate('facebook', {
    scope: 'email'
}));
```

email：存取用戶的主要電子郵件地址
user_likes：存取用戶按讚的內容清單

會自動要求存取用戶的公開個人檔案

*public_profile是預設值

對於用戶公開個人檔案的項目子集，

提供存取權

server.js

server設定使用

- passport
- cookie
- session

npm install express-session cookie-parser --save

```
// FB authentication
var passport = require('passport');
var session = require('express-session');
var cookieParser = require('cookie-parser');
// 引入body-parser 模組
var bodyParser = require('body-parser');

// 設定passport及route
require('./config/passport')(passport);

// read cookies (needed for auth)
app.use(cookieParser());
```

```
// required for passport
app.use(session({
    secret: 'ilovekk',
    resave: false,
    saveUninitialized: false
}));

app.use(passport.initialize());
// persistent login sessions
app.use(passport.session());
```

server.js

```
// 設定路徑
require('./app/routes.js')(app, passport);
```

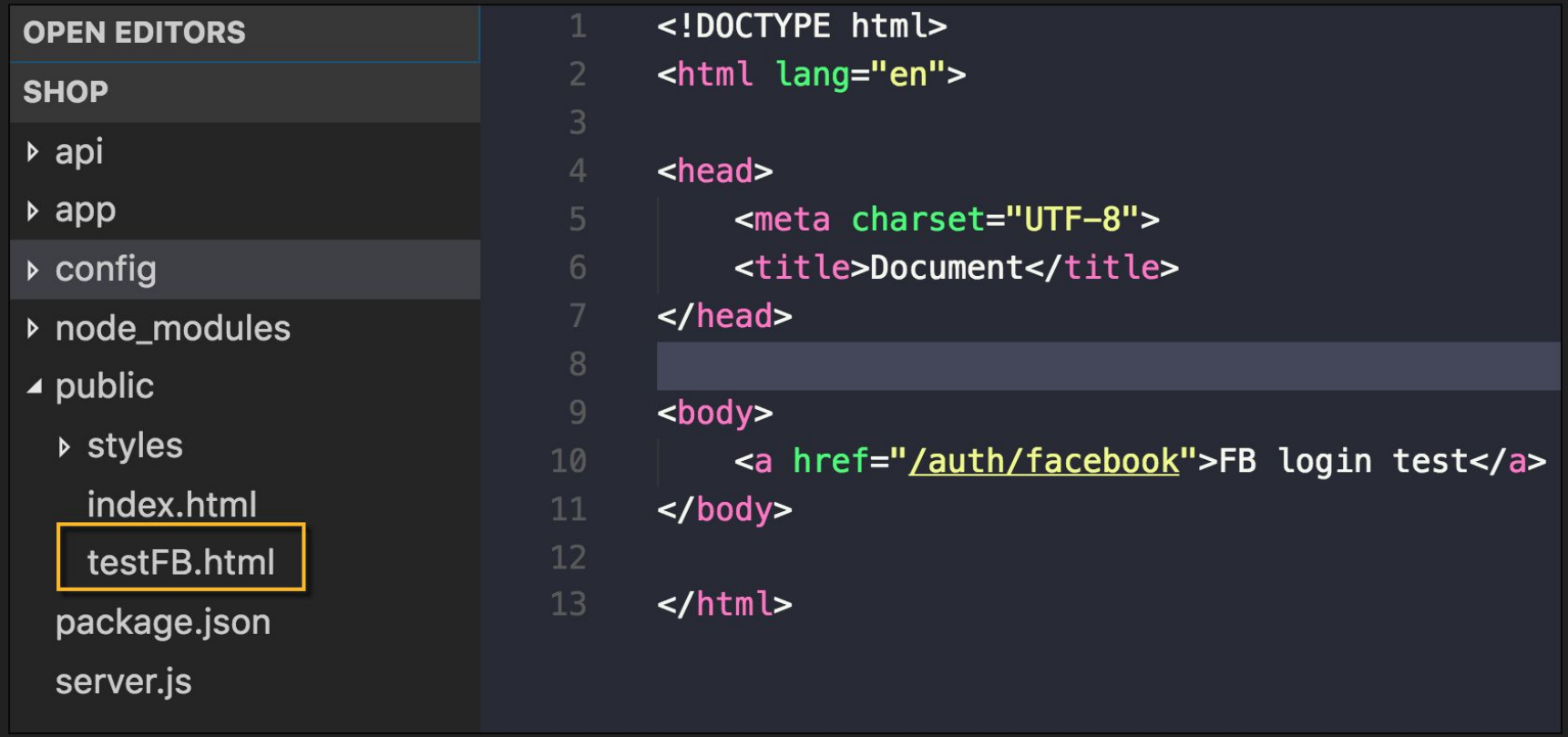
routes.js

```
module.exports = function(app, passport) {
    // 建立API
```

新增 /fbtest route來測試

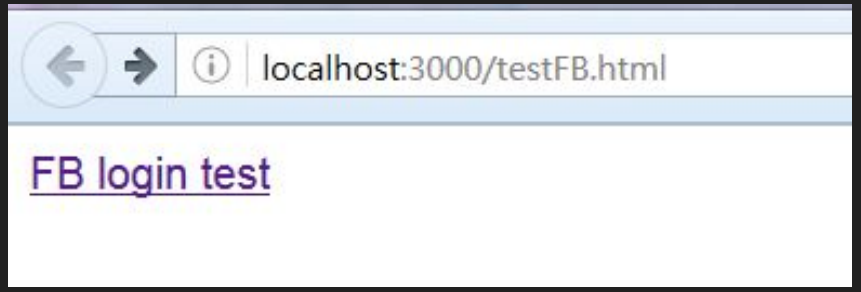
```
// test FB login
app.get('/fbtest',function (req,res) {
    console.log("req.session in /fbtest:");
    console.info(req.session);
    if (req.session.passport.user) {
        res.send("Hello ,user:" + req.session.passport.user);
    } else {
        res.send("logout successful!");
    }
});
```

在public下新增 tesFB.html



簡單測試

http://localhost:3000/testFB.html



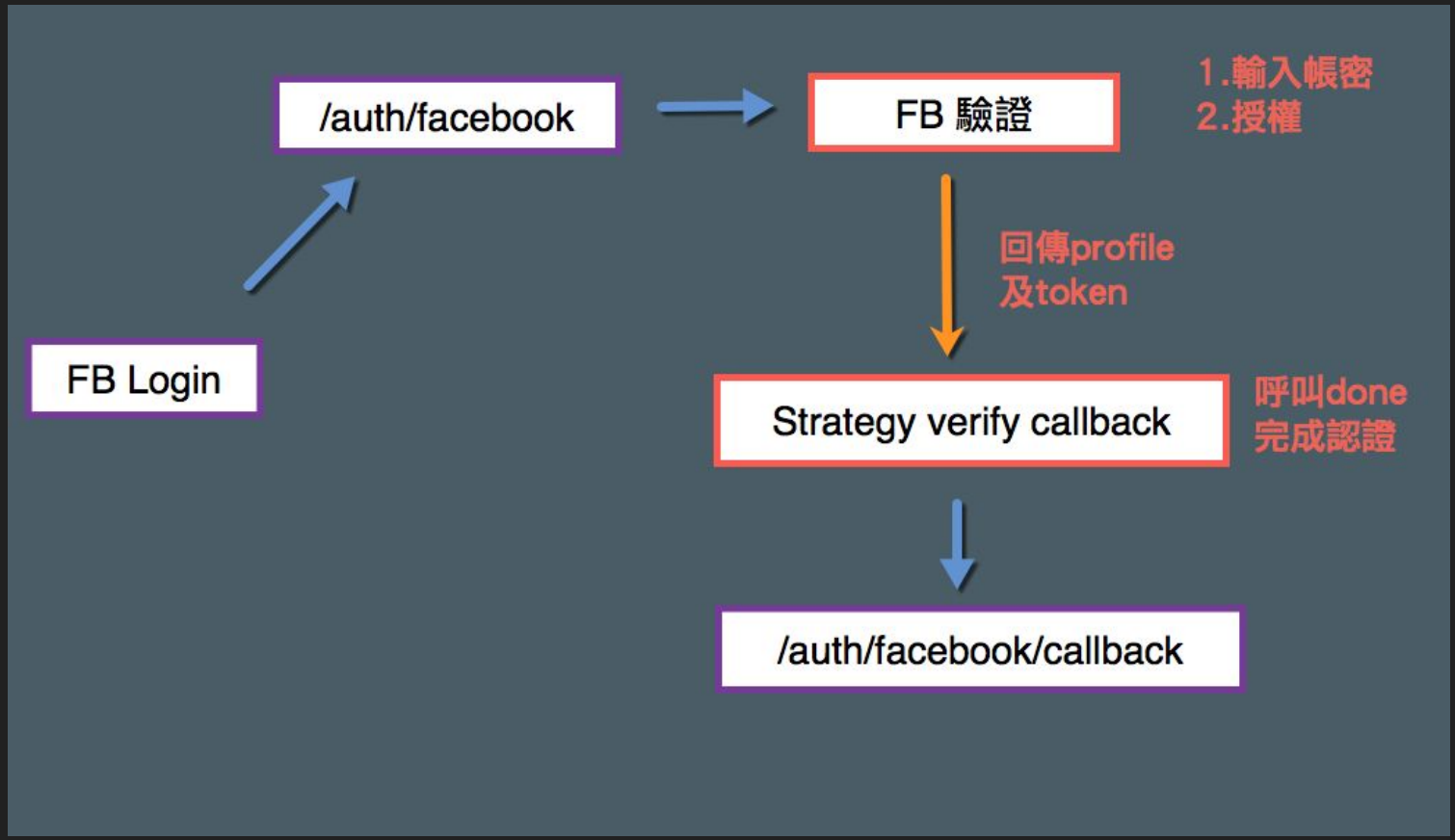

```
GET /testfb 200 201.577 ms - 14
req.session in /fbtest:
Session {
  cookie:
   { path: '/',
     _expires: null,
     originalMaxAge: null,
     httpOnly: true },
  passport: { user: '584ba11e1df1d41c709c90e7' } }
```

練習時間:10~15分鐘

- 調整config/auth.js
- 調整config/passport.js
- 調整server.js
- 到/fbtest

  測試是否取得FB回傳的user資料
- check mlab是否有多一筆user資料

流程


/auth/facebook
FB 驗證
1.輸入帳密
2.授權
回傳profile
及token
FB Login
Strategy verify callback
呼叫done
完成認證
/auth/facebook/callback

Passport 參考資料

http://passportjs.org/docs#facebook

https://github.com/jaredhanson/passport-facebook

https://github.com/passport/express-4.x-facebook-example/blob/master/server.js

建立購物車
及user相關的routes

app/routes.js

取得user model

```
//取得user model
var User = require('./models/user');
```

(1) GET /cart 取出購物車內容

```
app.get('/cart', function(req, res, next) {
    // 利用req.user._id去DB比對是否有此user
    User.findOne({
            _id: req.user._id
        })
        //因為product的type為ObjectId所以要populate
        .populate('data.cart.product')
        .exec(function(err, user) {
            if (err) return next(err);
            res.json({
                user: user
            });
        });
});
```

```
data: {
    totalValue: {…
    },
    //購物車array 每個elem
    //reference到product
    cart: [{
        product: {…
        },
        quantity: {…
        },
        subtotal: {…
        }
```

測試

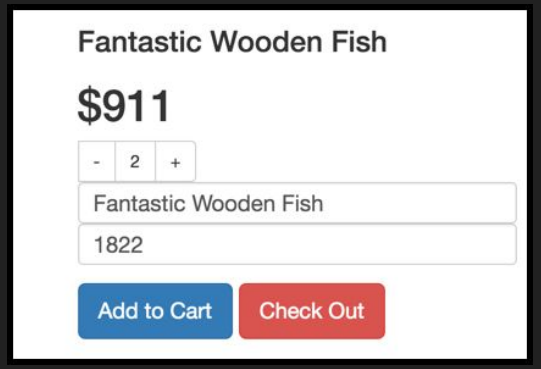
- 先透過testFB.html登入
- 再到/cart
- 安裝chrome套件
  JSON Viewer


```
// 20161113134654
// http://localhost:3000/cart

{
  "user": {
    "_id": "582692c352c42cfb237ef621",
    "email": "kuolun@gmail.com",
    "token":
"EAAOtYNHiRcsBAKeD0JphIsvWJZAv3uU4IYTK5oUOzvkZBThvRjkh8axzBXVdj
LgxDnUxW9SqRB2aOLoL4LDFk8F4vmqrU2IbUrD9LyseZC7ws8QZChY4sa59CnNs
    "facebook": "10157090854765075",
    "__v": 1,
    "data": {
      "cart": [
        {
          "product": {
            "_id": "581eb9f21d915c43f5b517eb",
            "image": "http://lorempixel.com/640/480/nature",
            "price": 350,
```

(2) PUT /updateCart　更新User購物車的內容

```
app.put('/updateCart', function(req, res, next) {
    User.findById({
        _id: req.user._id
    }, function(err, user) {
        user.data.cart.push({
            //put傳來的
            product: req.body.productid,
            quantity: parseInt(req.body.quantity),
            subtotal: parseInt(req.body.subtotal)
        });
```



PUT /updateCart

```
// req.body內為JSON，是string(透過bodyParser處理)
user.data.totalValue = (user.data.totalValue + parseInt(req.body.subtotal));
user.save(function(err, user) {
    if (err) return next(err);
    // 回傳save後的user
    return res.json({
        user: user
    });
});
```

測試是否會遇到錯誤？ 用testFB.html內表單(v4)

- 改route

或

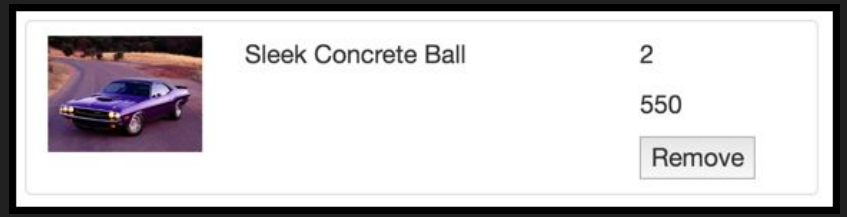
- 使用method-override
  npm install method-override --save

```
var methodOverride = require('method-override');
app.use(methodOverride('_method'));
```

測試


```
"__v": 2,
"data": {
  "cart": [
    {
      "product": "581eb9f21d915c43f5b517eb",
      "_id": "58269f3d66a7908a249cb802",
      "subtotal": 99,
      "quantity": 1
    },
    {
      "product": "581eb9f21d915c43f5b517ec",
      "_id": "582803853ff7d877df4ef74e",
      "subtotal": 55,
      "quantity": 1
    }
  ],
```

(3) PUT /remove 移除購物車內資料



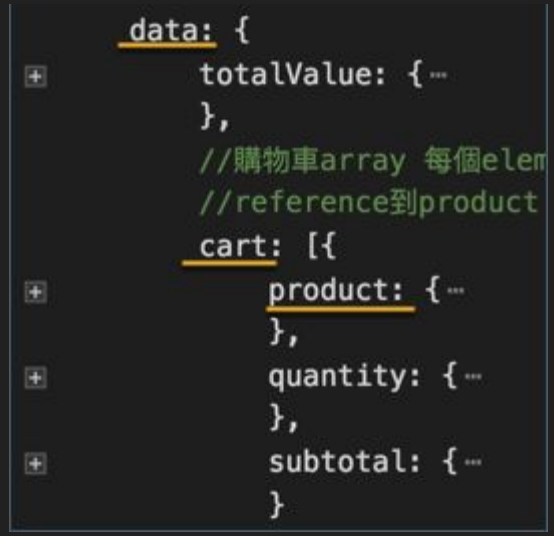

```
app.put('/remove', function(req, res, next) {
    User.findOne({
        // 有登入的傳進來的req會帶有user資料(req.user)
        _id: req.user._id
    }, function(err, foundUser) {
        // 利用ObjectId移除該item
        foundUser.data.cart.pull(String(req.body.itemid));
        foundUser.data.totalValue = (foundUser.data.totalValue - parseInt(req.body.subtotal));
        foundUser.save(function(err, found) {
            if (err) return next(err);
            res.json(found);
        });
    });
});
```

(4) GET /me　載入目前已登入user的資料

```
app.get('/me', function(req, res) {
    //check是否有user登入
    if (!req.user) {
        return res.status(401).
        json({
            error: 'User Not logged in!'
        });
    }
```

GET /me　　populate user購物車中的product

```
req.user.populate({
    path: 'data.cart.product'
}, function(error, user) {
    //錯誤處理
    if (error) {
        return res.status(500).
        json({
            error: error.toString()
        });
    } //資料找不到
    if (!user) {
        return res.status(404).
        json({
            error: 'Not found'
        });
    }
    // populate完回傳
    res.json({
        user: user
    });
});
```



(5) GET /logout 登出

req.logout( ) 會移除req.user及清除login session

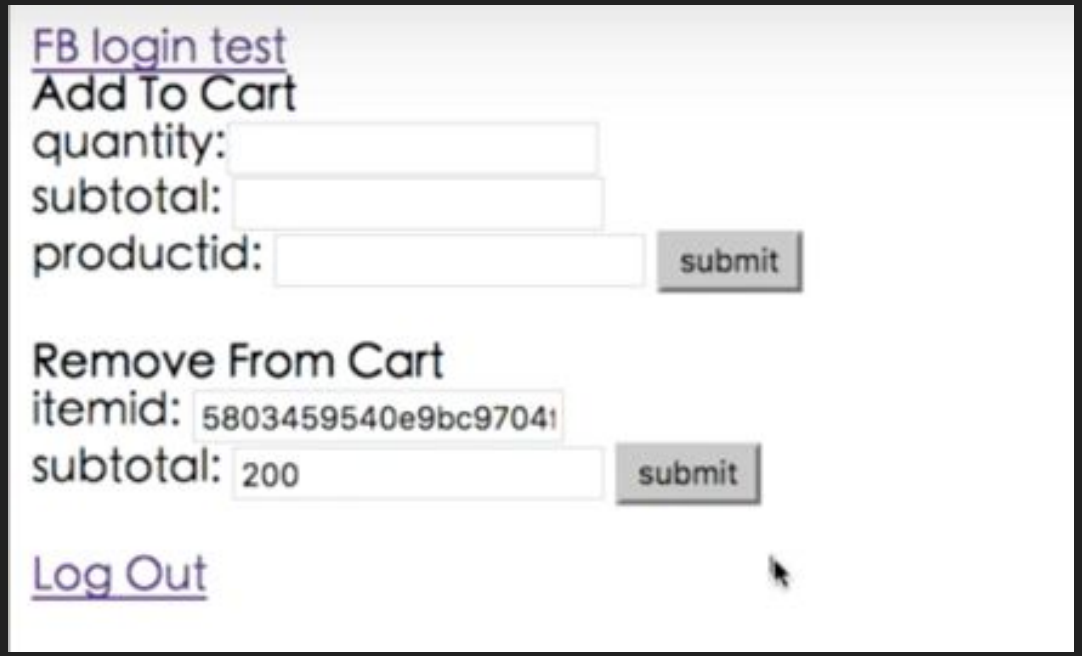
```
app.get('/logout', function(req, res) {
    // provided by passport.
    req.logout();
    res.redirect('/');
});
```

練習15~20分鐘

- 建立這5個routes
- 測試加product到購物車
- 測試移除該product
- 測試logout

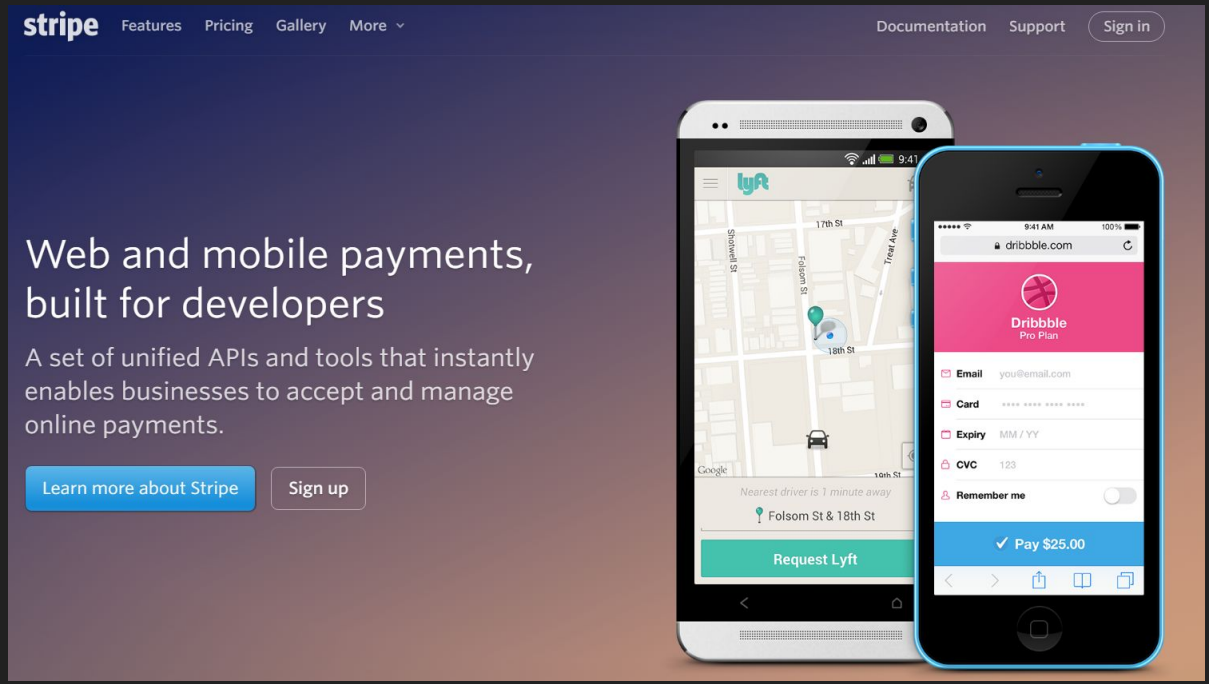
Question ?

測試成功嗎？



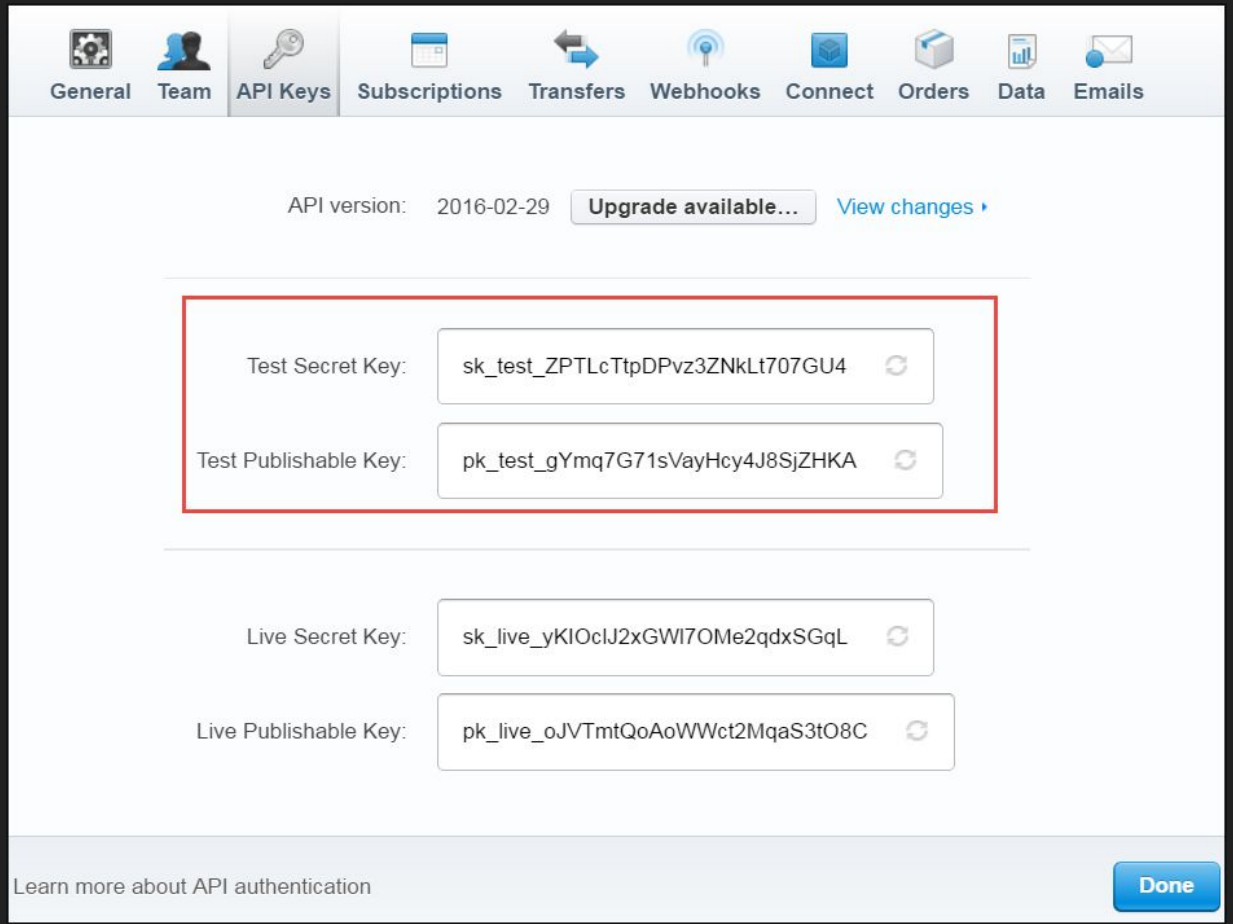
Stripe

https://stripe.com/



註冊取得sk/pk key

一樣先copy出來



步驟:

後端:

1. npm install stripe --save
2. 引入stripe js檔 (public/views/index.html)
3. 新增POST /payment routes (app/routes.js)

前端:(angular 1)

- 新增$scope.pay function (public/js/app.js)
  - $scope.stripeToken={ }
  - Stripe.card.createToken( )
  - $http.post('/payment',function

testStripe.html

```
<!--include stripe.js-->
<script type="text/javascript" src="https://js.stripe.com/v2/"></script>
<!--設定publishable key-->
<script type="text/javascript">
    Stripe.setPublishableKey('');
</script>
```

先處理後端

建立Stripe（app/routes.js)

Test secret key(sk)

```
//Stripe結帳
var Stripe = require('stripe')('sk_test_ZPTLcTtpDPvz3ZNkLt707GU4');
```

POST /payment
/app/routes.js

用POST method

建立charge 物件
-數量(amount)

-貨幣(currency)

-stripeToken:req.body.stripeToken
前端傳入

-描述(description)

```
app.post('/payment', function(req, res) {
    Stripe.charges.create({
            //Stripe的價格要用cents所以x100且四捨五入
            amount: Math.ceil(req.user.data.totalValue * 100),
            currency: 'usd',
            source: req.body.stripeToken, //取得stripeToken
            description: 'Example charge from kuolun'
        },
        //成功的話會拿到charge object
        function(err, charge) {
            if (err && err.type === 'StripeCardError') {…
            }
            if (err) {…
            }
            // 清空購物車
            req.user.data.cart = [];
            req.user.data.totalValue = 0;
            req.user.save(function() {
                // 成功的話回傳id及狀態
                return res.json({
                    id: charge.id,
                    status: charge.status
                });
            });
        });
});
```

amount先改任意數字for test

- 實際資料是由req.user.data抓

```
app.post('/payment', function(req, res) {
    Stripe.charges.create({
            // 從req.user.data去抓要charge的資料
            //Stripe的價格要用cents所以x100且四捨五入
            // amount: 888,
            amount: Math.ceil(req.user.data.totalValue * 100),
            currency: 'usd',
            source: req.body.stripeToken, //取得stripeToken
            description: 'Example charge from kuolun'
        },
```

先改成可以測試的API版本

```
app.post('/payment', function(req, res) {
    Stripe.charges.create({
        // 從req.user.data去抓要charge的資料
        //Stripe的價格要用cents所以x100且四捨五入
        // for test
        amount: 888,
        // amount: Math.ceil(req.user.data.totalValue * 100),
        currency: 'usd',
        source: req.body.stripeToken, //取得stripeToken
        description: 'Example charge from kuolun'
    },
```

```
function(err, charge) {
    if (err && err.type === 'StripeCardError') {…
    }
    if (err) {…
    }
    // 清空購物車
    // req.user.data.cart = [];
    // req.user.data.totalValue = 0;
    // req.user.save(function() {
    //     // 成功的話回傳id及狀態
    //     return res.json({
    //         id: charge.id,
    //         status: charge.status
    //     });
    // });
    res.send("charge success!");
});
```

用form測試

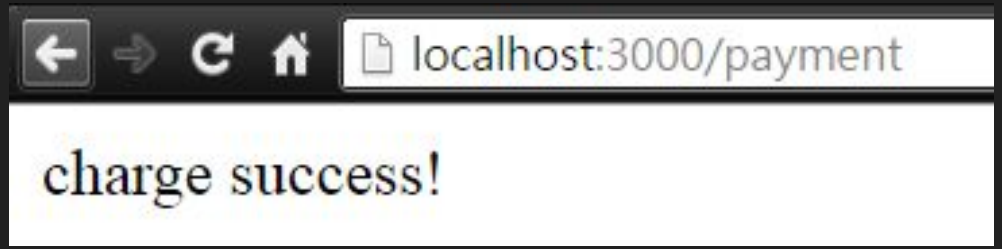
public/testStripe.html

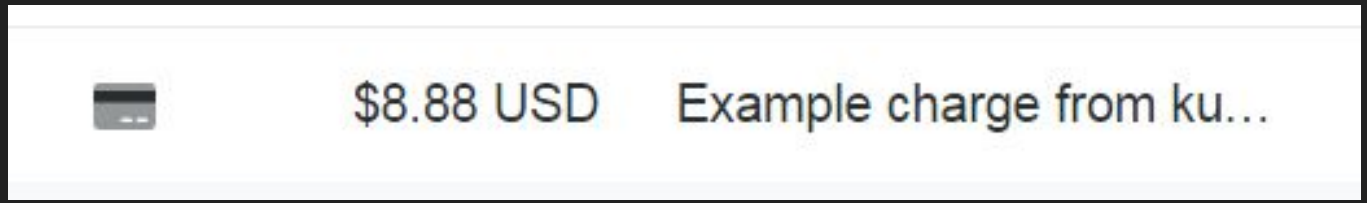
改成自己的Test publishable key

```
<script type="text/javascript">
    Stripe.setPublishableKey('pk_test_gYmq7G71sVayHcy4J8SjZHKA');
</script>
</head>
```

測試網址（v5）

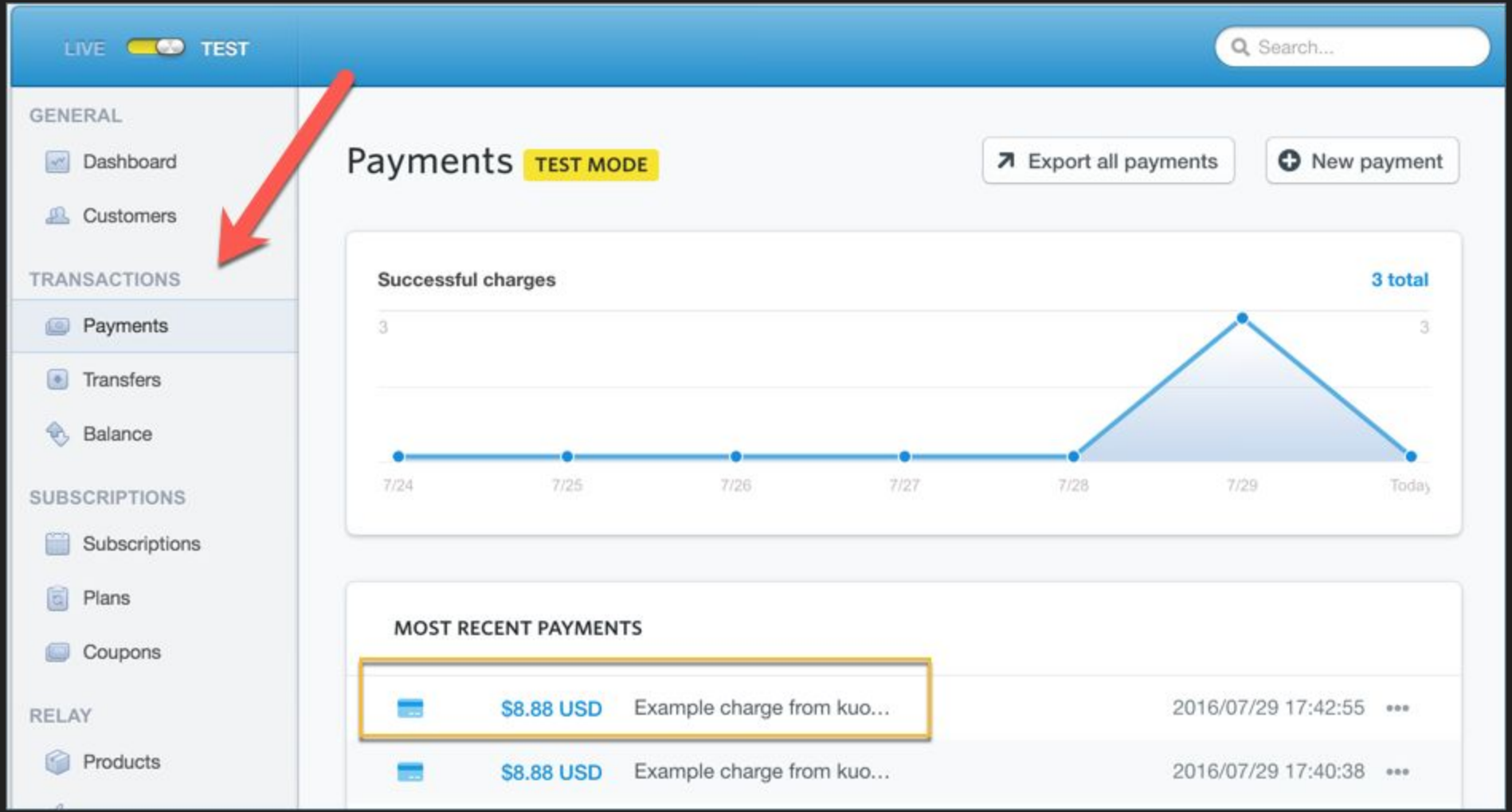
http://localhost:3000/testStripe.html





測試卡片

Card Number :4242 4242 4242 4242　(16碼無空白)
mm/yy 11/17
cvc 123


LIVE
TEST
Search...
GENERAL
Dashboard
Customers
TRANSACTIONS
Payments
Transfers
Balance
SUBSCRIPTIONS
Subscriptions
Plans
Coupons
RELAY
Products
Payments
TEST MODE
Export all payments
New payment
Successful charges
3 total
3
3
7/24
7/25
7/26
7/27
7/28
7/29
Today
MOST RECENT PAYMENTS
$8.88 USD
Example charge from kuo...
2016/07/29 17:42:55
$8.88 USD
Example charge from kuo...
2016/07/29 17:40:38

練習 : 15分鐘

- 取得Stripe pk/sk key
- 調整app/routes.js
- 修正成可測試版
- 用http://localhost:3000/testStripe.html測試
- 確認stripe是否有payment

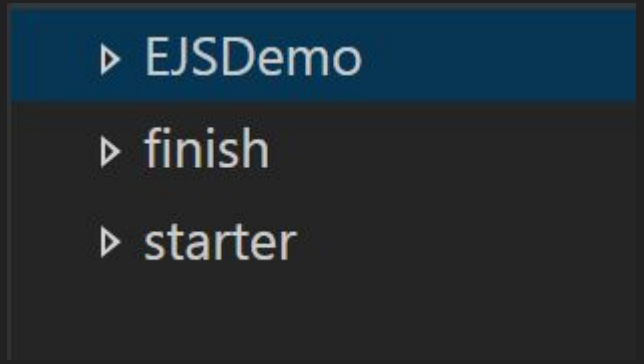
EJS

Rendering HTML and Templates

- 學會使用res.render( )來render HTML (從EJS file)
- 了解EJS是什麼及為什麼要用它
- 學會傳入變數到EJS template

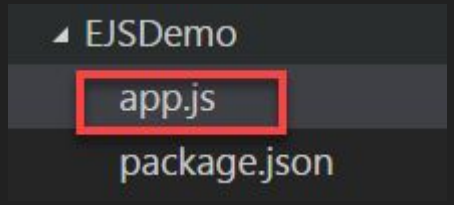
新增EJSDemo folder

- 執行npm init建立package.json
- entry point改為app.js




```
name: (EJSDemo) ejsdemo
version: (1.0.0)
description:
entry point: (index.js) app.js
test command:
git repository:
keywords:
author:
license: (ISC)
```

- 新增app.js
- 執行npm install express ejs --save




```
`-- express@4.14.0
  +-- accepts@1.3.3
  | +-- mime-types@2.1.11
  | | `-- mime-db@1.23.0
  | `-- negotiator@0.6.1
  +-- array-flatten@1.1.1
  +-- content-disposition@0.5.1
  +-- content-type@1.0.2
  +-- cookie@0.3.1
  +-- cookie-signature@1.0.6
  +-- debug@2.2.0
  | `-- ms@0.7.1
```

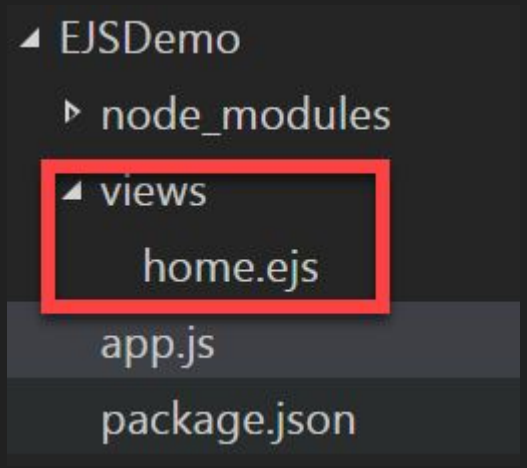
- 建立server
- 建立/ route
- 測試localhost:3000/

app.js

```
1  var express = require('express');
2  var app = express();
3
4  app.get('/',function (req,res) {
5      res.send('<h1>Home Page</ha>');
6  })
7
8
9  app.listen(3000,function(){
10     console.log('Server is running....');
11 });
```

使用EJS

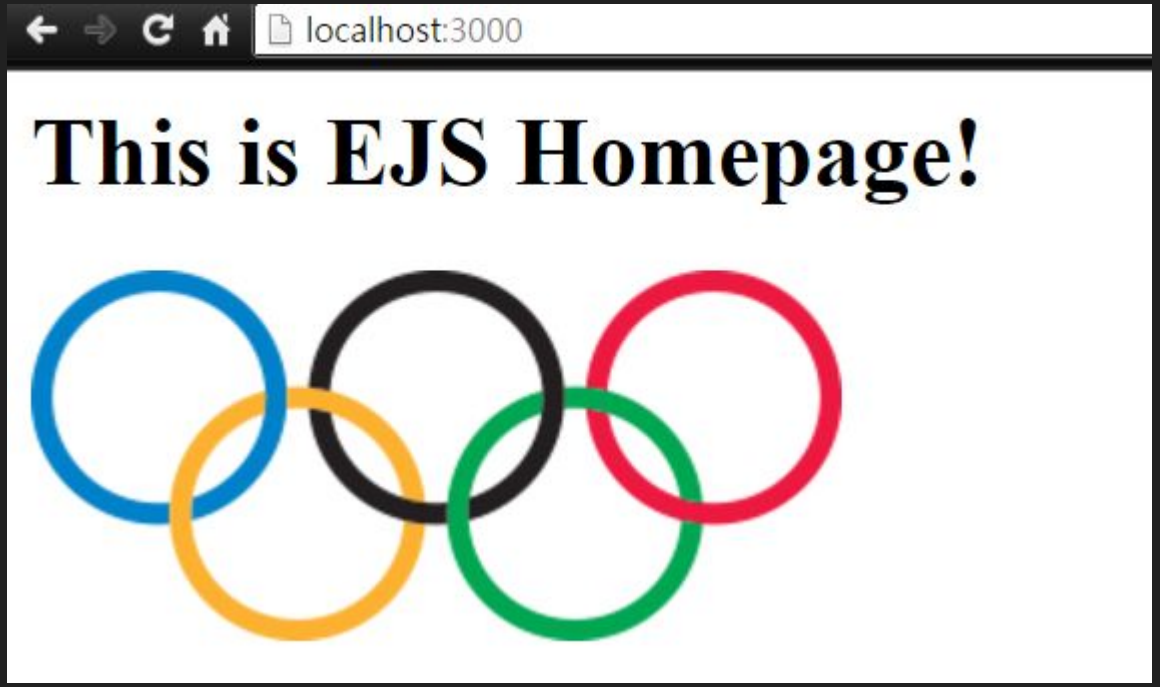
- 改用res.render
- 建立views folder及home.ejs
- 這邊的'home.ejs'會對應到 views/home.ejs




```
var express = require('express');
var app = express();

app.get('/',function (req,res) {
    res.render('home.ejs');
})


app.listen(3000,function(){
    console.log('Server is running....');
});
```


localhost:3000
This is EJS Homepage!

新增/books/:category

```
var express = require('express');
var app = express();

app.get('/',function (req,res) {
    res.render('home.ejs');
})

app.get('/books/:category',function (req,res) {
    res.send("You choose :"+req.params.category);
})

app.listen(3000,function(){
    console.log('Server is running....');
});
```

新增views/category.ejs

app.js | category.ejs | home.ejs

```
1  <h1>You choose category:</h1>
2
3  <p>this is from category.js</p>
```

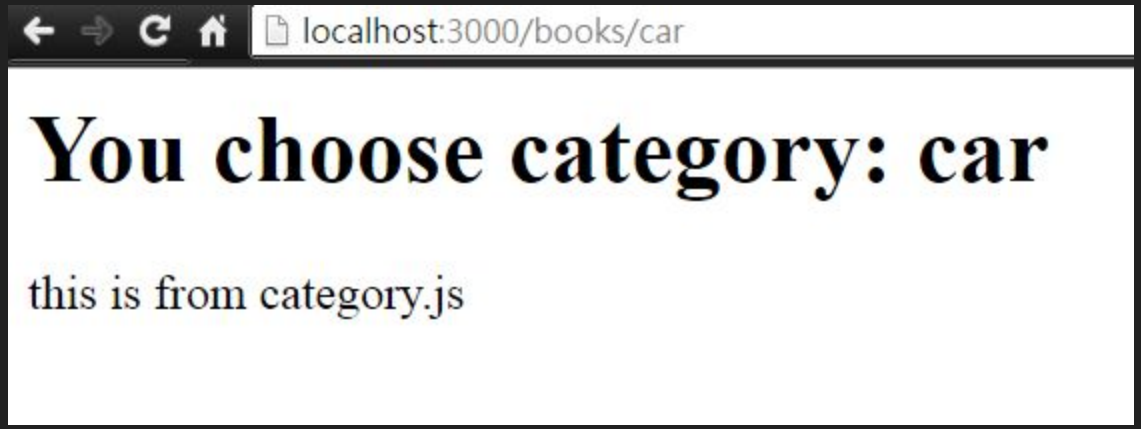
render category.ejs

app.js × | category.ejs | home.ejs

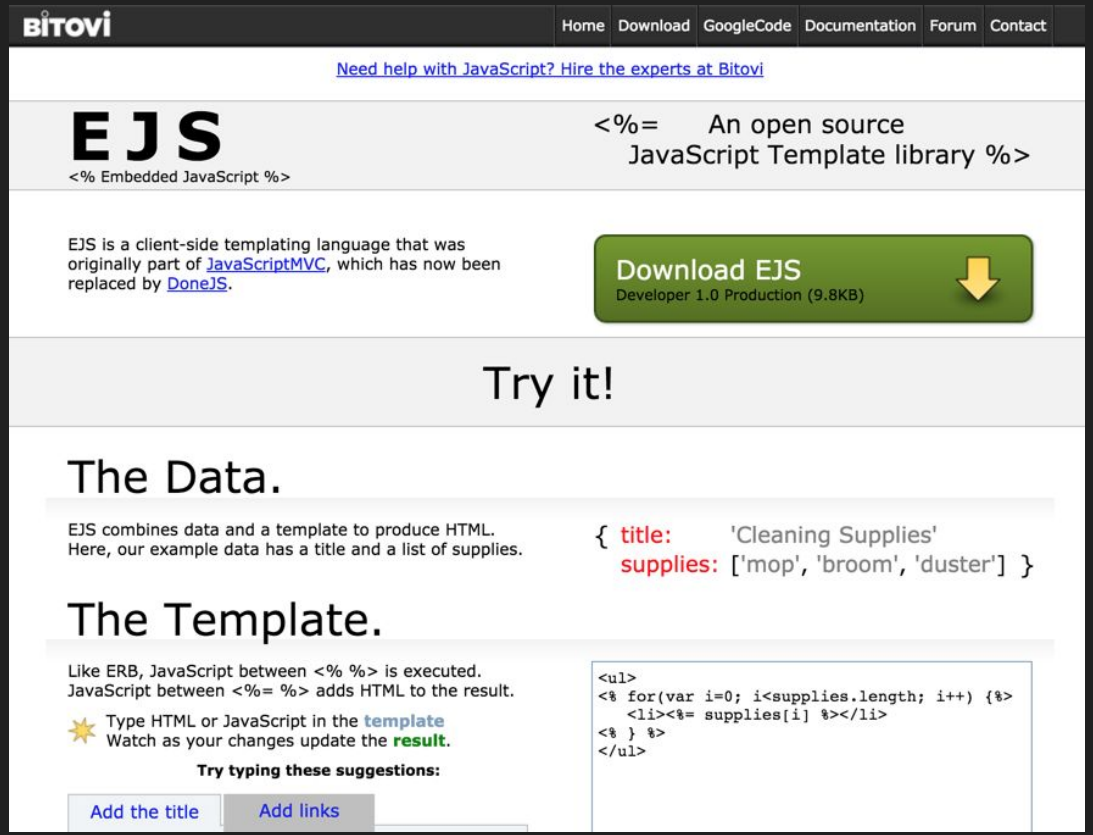
```
1  var express = require('express');
2  var app = express();
3
4  app.get('/',function (req,res) {
5      res.render('home.ejs');
6  })
7
8  app.get('/books/:category',function (req,res) {
9      // res.send("You choose :"+req.params.category);
10     var category = req.params.category;
11     res.render('category.ejs',{cate:category});
12 })
13
14 app.listen(3000,function(){
15     console.log('Server is running....');
16 });
```

app.js | category.ejs | home.ejs

```
1  <h1>You choose category: <%=cate%></h1>
2
3  <p>this is from category.js</p>
```


localhost:3000/books/car
You choose category: car
this is from category.js

EJS官網 http://www.embeddedjs.com/



EJS Demo

https://docs.google.com/presentation/d/12RbvtxMekzDSq21hJ_bX22kxWsr-ZDzJcLV4z6ZVO2A/edit?usp=sharing

建立Profile Page(練習)

button點下去導到FB去做驗證

驗證完後，導到profile頁面