

# DASA PROJECT REPORT

Team

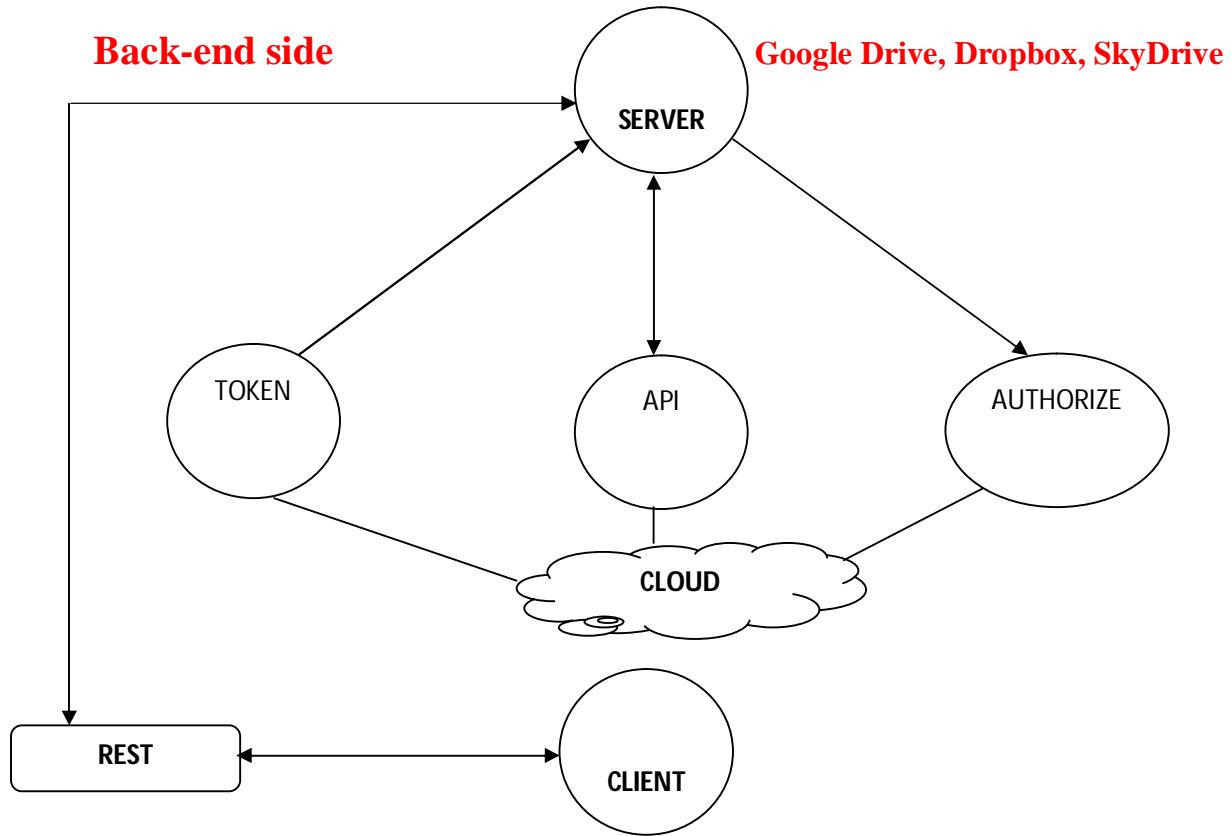
**Trần Thanh Duy**  
**Nguyễn Nho Dũng**

## Part 1: Requirement of application

See in the file project.pdf

## Part 2: Application structure (suggestion)

### Back-end side



Each user has only token when he/she registers API. And, the server archives this token for authorization (The token is secret and nobody can't share with others people)

## Part 2: Framework and Library

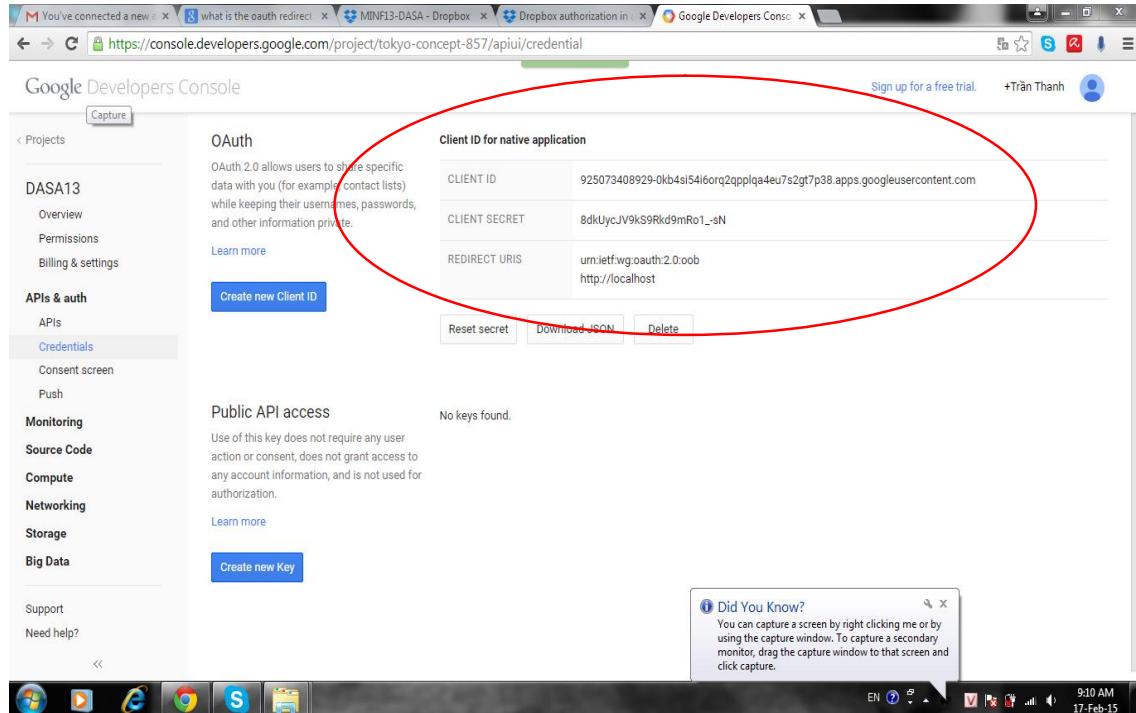
org.scribe: the simple OAuth for Java lib

## Part 3: REST API

In this project, our team use 3 types of API: Google Drive API, Dropbox API and Sky Drive API (Sky Drive is a cloud application of Microsoft)

## Part 4: Some screenshots

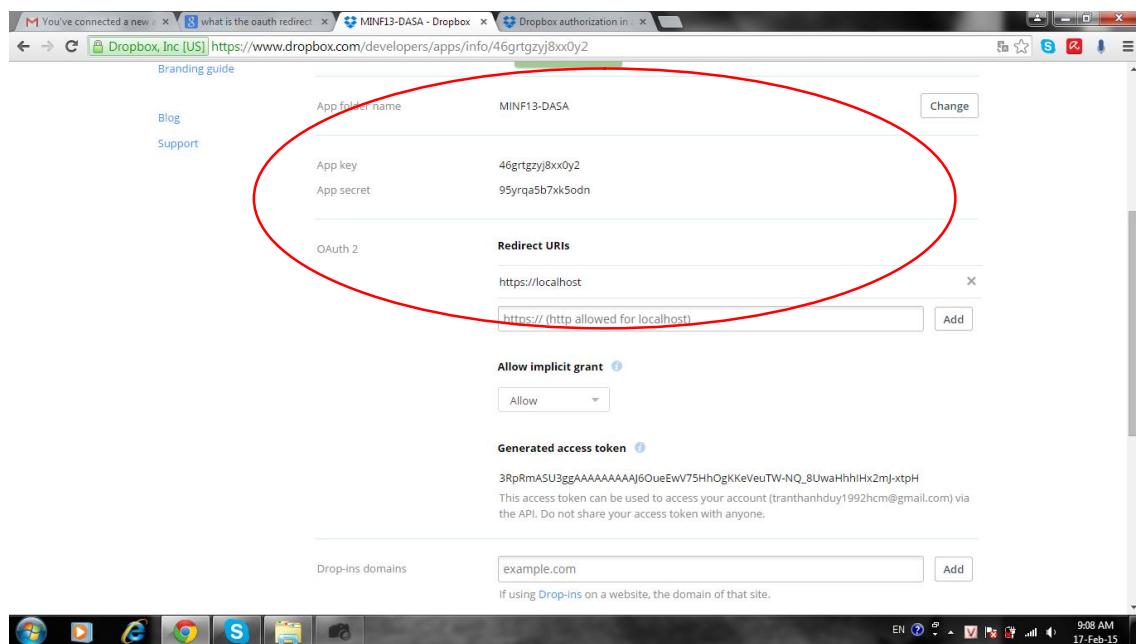
### API key of Google and Dropbox



The screenshot shows the Google Developers Console interface. On the left, there's a sidebar with 'Projects' (DASA13 selected), 'APIs & auth' (Credentials selected), and various monitoring and source code options. The main area is titled 'OAuth' and shows a detailed view of a client configuration. A red oval highlights the 'Client ID for native application' section, which contains the following information:

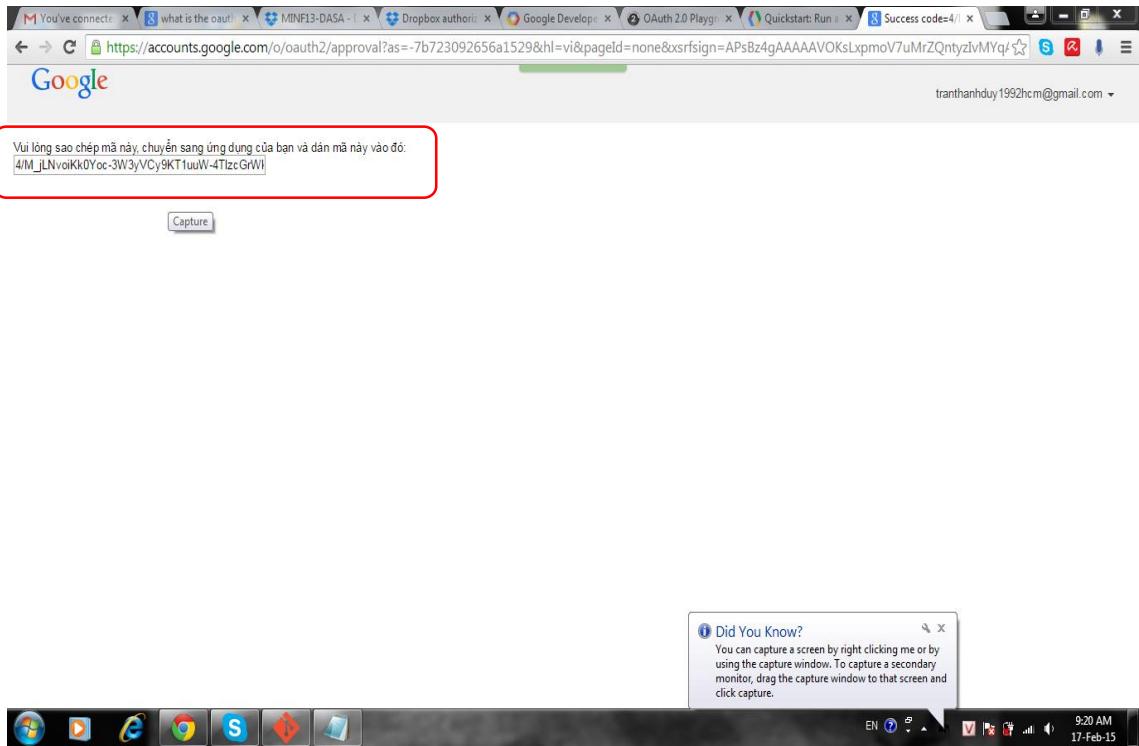
Client ID	Value
CLIENT ID	925073408929-0kb4si54i6orq2ppqlqa4eu7s2gt7p38.apps.googleusercontent.com
CLIENT SECRET	8dkUycJV9kS9Rkd9mRo1_sN
REDIRECT URIS	urn:ietf:wg:oauth:2.0:oob http://localhost

Below this, there's a 'Public API access' section with a note: 'No keys found.' and a 'Create new Key' button. At the bottom right, a tooltip says: 'Did You Know? You can capture a screen by right clicking me or by using the capture window. To capture a secondary monitor, drag the capture window to that screen and click capture.'

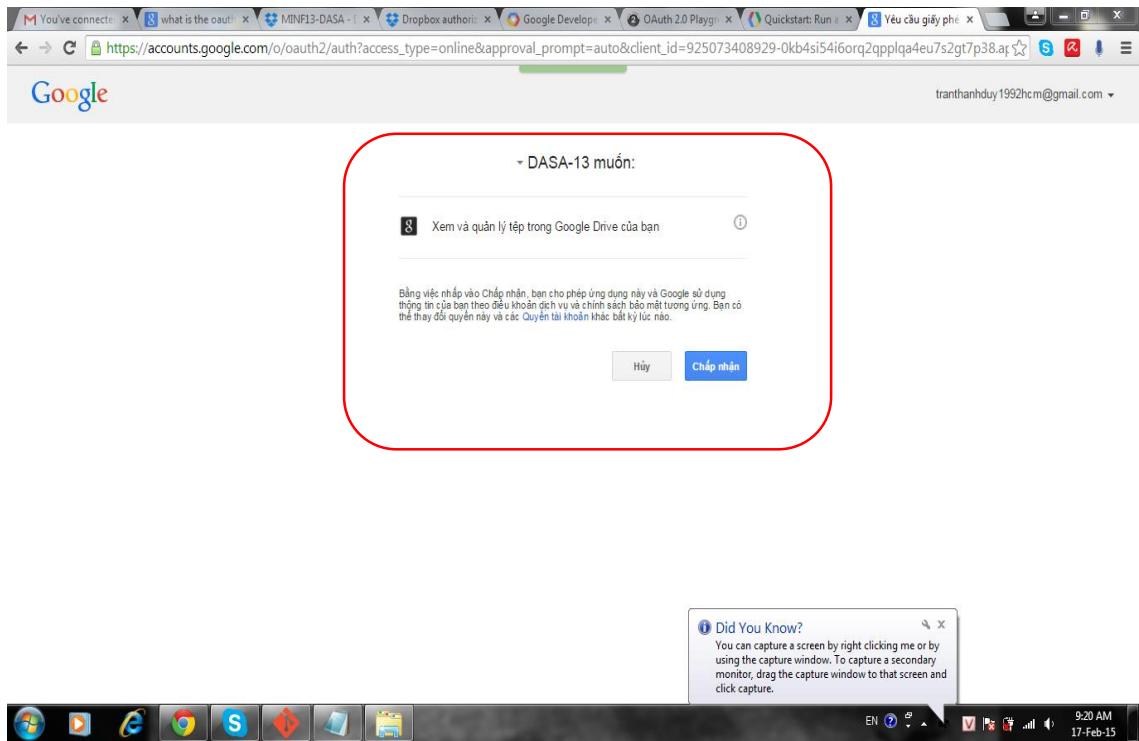


The screenshot shows the Dropbox developer account settings page. The top navigation bar includes links for 'Branding guide', 'Blog', and 'Support'. The main area has sections for 'App folder name' (set to 'MINF13-DASA'), 'App key' (value '46grtgzyj8xx0y2'), 'App secret' (value '95yrqa5b7xk5odn'), and 'OAuth 2'. A red oval highlights the 'OAuth 2' section. Below it is the 'Redirect URIs' section, which lists 'https://localhost' and 'https://(http allowed for localhost)' with an 'Add' button. There are also sections for 'Allow implicit grant' (set to 'Allow') and 'Generated access token' (with a note about access tokens). At the bottom, there's a 'Drop-ins domains' section with a text input field containing 'example.com' and an 'Add' button.

## Get code from Google Drive



## Authorization page from Google



## Get token from Google Drive (access to OAuth playground of Google)

Step 1 Select & authorize APIs

Step 2 Exchange authorization code for tokens

Once you got the Authorization Code from Step 1 click the Exchange authorization code for tokens button, you will get a refresh and an access token which is required to access OAuth protected resources.

Authorization code: 4/n5Jm\_LYxUN1LvWzHae2fU8Vw\_98ydQQXftFIvIQY4Wo.oqeblh\_BIGQUBrG\_bnfDxpJKtrgqlwl

Exchange authorization code for tokens

Refresh token: 1Orin8HNAL8C0ap\_8c0dCY4VYeFc

Access token: ya29.HQGv-k6Tj6rZ0Lm52iTx0\_bEoCl37Rb1NJoLZQxbFfLXgOwtcRYJQExwLVys8ZwL5Pe5NcLj1Q

Auto-refresh the token before it expires.

The access token will expire in 3590 seconds.

Note: The OAuth Playground does not store refresh tokens, but as refresh tokens never expire, user should go to their Google Account Authorized Access page if they would like to manually revoke them.

Request / Response

```
POST /o/oauth2/token HTTP/1.1
Host: accounts.google.com
Content-length: 265
Content-type: application/x-www-form-urlencoded
User-agent: google-oauth-playground

code=4/n5Jm_LYxUN1LvWzHae2fU8Vw_98ydQQXftFIvIQY4Wo.oqeblh_BIGQUBrG_bnfDxpJKtrgqlwl&redirect_uri=https%3A%2F%2Fdevelopers.google.com%2Foauthplayground&client_id=407408718132.apps.googleusercontent.com&scope=&client_secret=*****&grant_type=authorization_code
```

```
HTTP/1.1 200 OK
Alternate-protocol: 443:quic,p=0.08
Content-length: 248
Content-type: text/html; charset=UTF-8
Date: Tue, 17 Feb 2015 02:27:21 GMT
X-Content-Type-Options: nosniff
Content-Disposition: attachment; filename="json.txt"; filename*=UTF-8''json.txt
Expires: Fri, 01 Jan 1990 00:00:00 GMT
X-Frame-Options: SAMEORIGIN
Content-type: application/json; charset=utf-8
Content-Encoding: gzip

{
  "access_token": "ya29.HQGv-k6Tj6rZ0Lm52iTx0_bEoCl37Rb1NJoLZQxbFfLXgOwtcRYJQExwLVys8ZwL5Pe5NcLj1Q",
  "token_type": "Bearer",
  "expires_in": 3600,
  "refresh_token": "1Orin8HNAL8C0ap_8c0dCY4VYeFc"
}
```

EN 9:27 AM 17-Feb-15

Step 1 Select & authorize APIs

Step 2 Exchange authorization code for tokens

Once you got the Authorization Code from Step 1 click the Exchange authorization code for tokens button, you will get a refresh and an access token which is required to access OAuth protected resources.

Authorization code: 4/VSYvhukYq6PWr5ki2aKwfAWZ17AH

Exchange authorization code for tokens

Refresh token: WVH8kAMEudvR5kjSp0R30zcRFq6

Access token: ya29.HQH7wwETvoN9QIf8NTqYaah

Auto-refresh the token before it expires.

The access token will expire in 3584 seconds.

Note: The OAuth Playground does not store refresh tokens, but as refresh tokens never expire, user should go to their Google Account Authorized Access page if they would like to manually revoke them.

Request / Response

```
POST /o/oauth2/token HTTP/1.1
Host: accounts.google.com
Content-length: 265
Content-type: application/x-www-form-urlencoded
User-agent: google-oauth-playground

code=4/VSYvhukYq6PWr5ki2aKwfAWZ17AH&redirect_uri=https%3A%2F%2Fdevelopers.google.com%2Foauthplayground&client_id=407408718132.apps.googleusercontent.com&scope=&client_secret=*****&grant_type=authorization_code
```

```
HTTP/1.1 200 OK
Alternate-protocol: 443:quic,p=0.08
Content-length: 248
Content-type: text/html; charset=UTF-8
Date: Tue, 17 Feb 2015 02:11:25 GMT
X-Content-Type-Options: nosniff
Content-Disposition: attachment; filename="json.txt"; filename*=UTF-8''json.txt
Expires: Fri, 01 Jan 1990 00:00:00 GMT
X-Frame-Options: SAMEORIGIN
Content-type: application/json; charset=utf-8
Content-Encoding: gzip

{
  "access_token": "ya29.HQH7wwETvoN9QIf8NTqYaah",
  "token_type": "Bearer",
  "expires_in": 3600,
  "refresh_token": "1/mis4LaAqb2R0w8Nm0HPVwq"
}
```

Did You Know?  
You can capture a screen by right clicking me or by using the capture window. To capture a secondary monitor, drag the capture window to that screen and click capture.

EN 9:11 AM 17-Feb-15

## Get file from Drive

The screenshot shows the OAuth 2.0 Playground interface. On the left, a sidebar lists steps: Step 1 Select & authorize APIs, Step 2 Exchange authorization code for tokens, and Step 3 Configure request to API. The main area is titled "Request / Response".

**Request / Response**

**Expires:** Tue, 17 Feb 2015 02:13:02 GMT  
**Vary:** Origin, X-Origin  
**Server:** GSE  
**Etag:** "17f8c8c202"  
**Cache-control:** private, max-age=0, must-revalidate, no-transform  
**Date:** Tue, 17 Feb 2015 02:13:02 GMT  
**X-frame-options:** SAMEORIGIN  
**Content-type:** application/json; charset=UTF-8

```
{ "mimeType": "text/plain", "version": "400996", "appDataContents": false, "thumbnails": { "url": "https://lh4.googleusercontent.com/Oo015wVbnb0J1j2hEt0_XO_cugI3xyKFB-41tGFMjEq4YkEvG4s-6cXr0kLDG17f8c8c202", "label": "Image" }, "restricted": false, "starred": false, "visible": true, "added": false, "trashed": true }, explicitlyTrashed: true, etag: "'2a2D6PB1LmDhBqNSNEy5BE/MTQyNDA1MDg1MTQxOA'", lastModifyingUserName: "Tran thanh Duy", writersCanShare: true, owners: [ { emailAddress: "tranthanhduy1992hcm@gmail.com", kind: "driveUser", isAuthenticatedUser: true, displayName: "Tran thanh Duy", permissionId: "6660833771733690272" } ], id: "0B7pg1-vOBo3RYMn0ZU4tNINhYzB" }
```

**Note:** The OAuth access token in Step 2 will be added to the Authorization header of the request.

**Did You Know?**  
You can capture a screen by right clicking me or by using the capture window. To capture a secondary monitor, drag the capture window to that screen and click capture.

Hộp thư đón (120) | what is the oauth re... | MINFI3-DASA - D... | Dropbox autorizatio... | Google Developers | Quickstart: Run a Driv... | OAuth 2.0 Playground |

https://developers.google.com/oauthplayground/?code=4/n5Jm\_LYxN1LwZhaE2F8Vw\_98yldQXFrFIvIQY4Wo.oqeblh\_BIGQUBrG\_bnfDxpJktrgqlwl

Google Developers

Tìm kiếm

tranthanhduy1992hcm@gmail.com Đăng xuất

## OAuth 2.0 Playground

Step 1 Select & authorize APIs

Step 2 Exchange authorization code for tokens

Step 3 Configure request to API

Construct your HTTP request by specifying the URI, HTTP Method, headers, content type and request body. Then click the "Send the request" button to initiate the HTTP Request.

HTTP Method: GET Add headers 0

Request URI: https://www.googleapis.com/drive/v2/files/0B7pgi-v08o3RQ1Bw

Enter request body 0 Content-Type application/json

Send the request List possible operations

Note: The OAuth access token in Step 2 will be added to the Authorization header of the request.

### Request / Response

POST /o/oauth2/token HTTP/1.1  
Host: accounts.google.com  
Content-length: 265  
Content-type: application/x-www-form-urlencoded  
User-agent: google-oauth-playground

code=4/n5Jm\_LYxN1LwZhaE2F8Vw\_98yldQXFrFIvIQY4Wo.oqeblh\_BIGQUBrG\_bnfDxpJktrgqlwl&redirect\_uri=https%3A%2F%2Fdevel...@google.com%2Foauthplayground&client\_id=407408718192.apps.googleusercontent.com&scope=&client\_secret=\*\*\*\*\*&grant\_type=authorization\_code

HTTP/1.1 400 Bad Request  
Alternate-protocol: 443:quic,p=0.08  
Content-length: 85  
X-xss-protection: 1; mode=block  
X-content-type-options: nosniff  
X-google-cache-control: remote-fetch  
Content-encoding: gzip  
Server: GSE  
Via: 1.1 gsa  
Pragma: no-cache  
Cache-control: no-cache, no-store, max-age=0, must-revalidate  
Date: Tue, 17 Feb 2015 02:27:49 GMT  
X-frame-options: SAMEORIGIN  
Content-type: application/json  
Expires: Fri, 01 Jan 1990 00:00:00 GMT

{  
  "error\_description": "Code was already redeemed.",  
  "error": "invalid\_grant"  
}

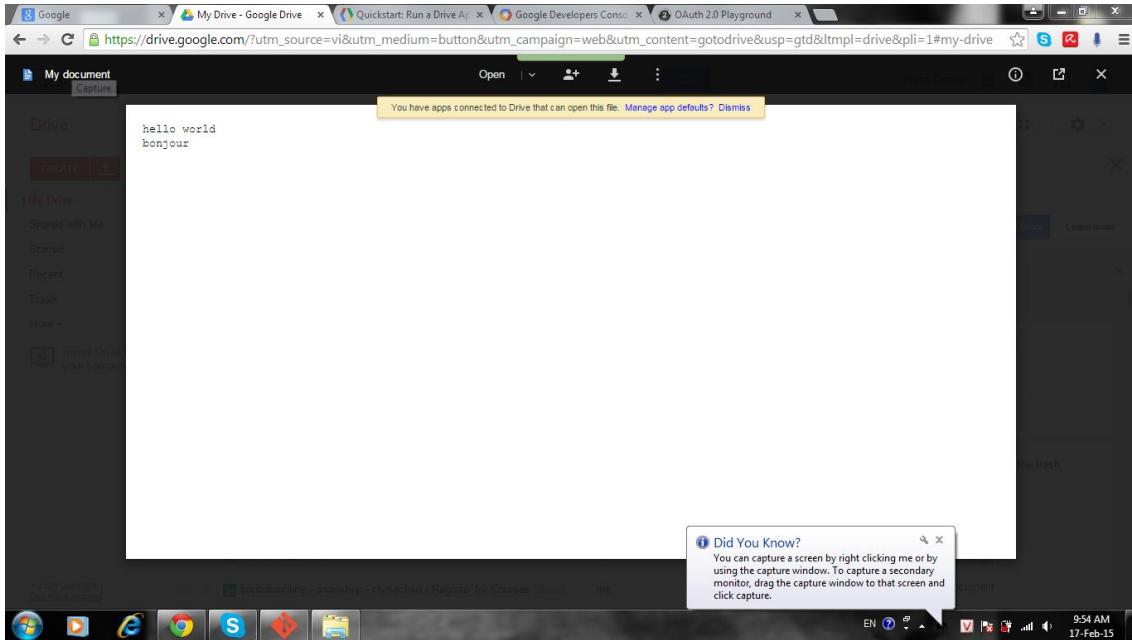
## Test result (test on Google Drive)

### Step 1: Get file information (from Google Drive)

The screenshot shows the Google Developers OAuth 2.0 Playground interface. On the left, under 'Step 1 Select & authorize APIs', there's a 'Capture' button. Under 'Step 2 Exchange authorization code for tokens', there's a 'Request / Response' section showing a successful HTTP request (HTTP/1.1 200 OK) to https://www.googleapis.com/drive/v2/files/0B7pgi-vO8o3RQ1Bw. The response body contains JSON data about a file, including its ID, name, and owner information. Below this, there's a note: 'Note: The OAuth access token in Step 2 will be added to the Authorization header of the request.' At the bottom, there's a toolbar with icons for Windows, Internet Explorer, Chrome, and others, along with system status indicators like battery level and date/time (9:28 AM, 17-Feb-15).

This screenshot shows the same interface after the user has completed Step 2. The 'Request / Response' section now displays detailed JSON data for a specific file, including its parents, last modified date, and various metadata fields. The note at the bottom remains the same. The system status indicators at the bottom show the same information as the previous screenshot.

## Step 2: Final result



## Part 5: Difficulties

Our team just has only difficulties, we can't deploy the front-end side but all results is correct (we tested on Google Drive,...)