OAuth 2.0 framework

Authorization and API Integration

Assignment 02

Group Members

Student Registration N0	Name
MS21908224	K.G.A.P.WIJERATHNE
MS21908606	R. M. M. M. RATHNAYAKE
MS21909078	CHATHURA PATHIRANA

Software Security - IE5042

Master of Science in Information Technology: Cyber Security

Sri Lanka Institute of Information Technology

Project milestones

1. Setup Project in Google Cloud Platform

First, we need to create an API to setup the project in Google cloud developer console which will be used to integrate with the web application. We have selected web application option for which the web application is integrated via APIs.

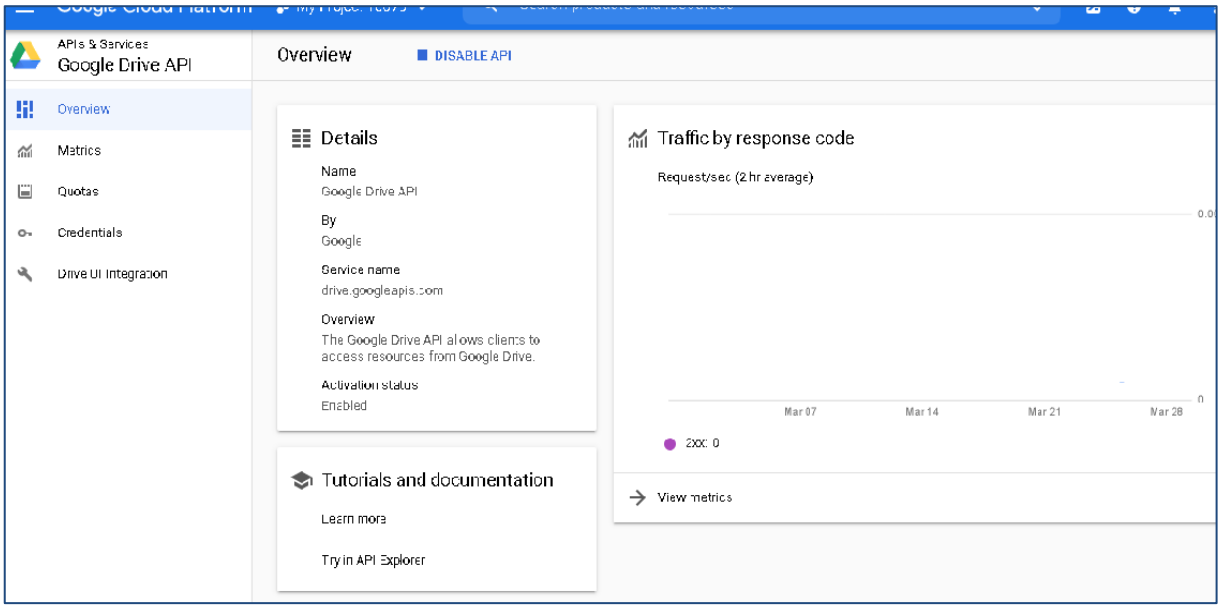

Figure 01: Setting up project

We have to enable Google drive API in order to integrate with web application. After enabling API, developers are able to start the configurations as per the requirements specified in the project.

Authorized JavaScript origin should properly be defined where the authorization request is originated and manage redirect URL for the requests which are receiving from web server. We run this in a personal PC therefore, localhost was provided to connect the path. We can use a public IP or domain to connect the same.

Figure 02: Authorized JavaScript origin

Once the above information is given, a token (client ID) will be generated from API server along with '**Client Secret'.** This both inputs (client ID and secret) should be provided in order to authenticate web application via API. Here, user email address is pre-defined in the API to eliminate additional permission problems.

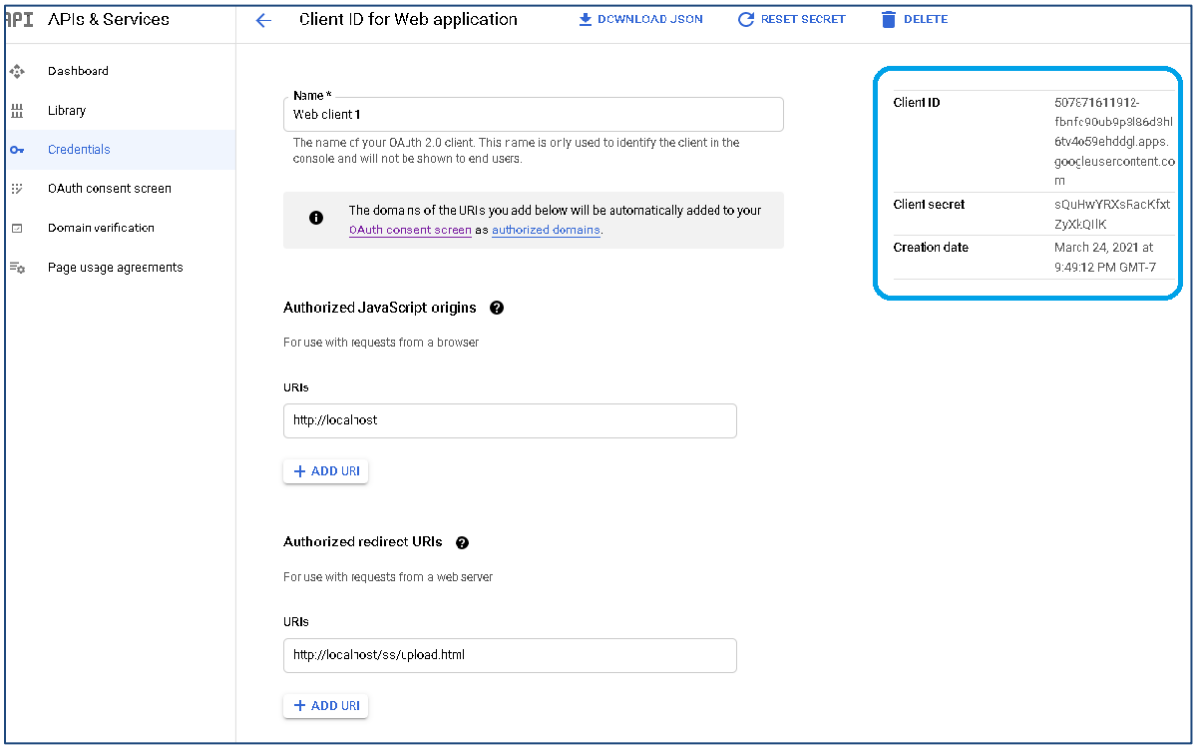

Figure 03: Client ID and secret generation

If the process is successfully completed the below mention output can be seen in JASON file.

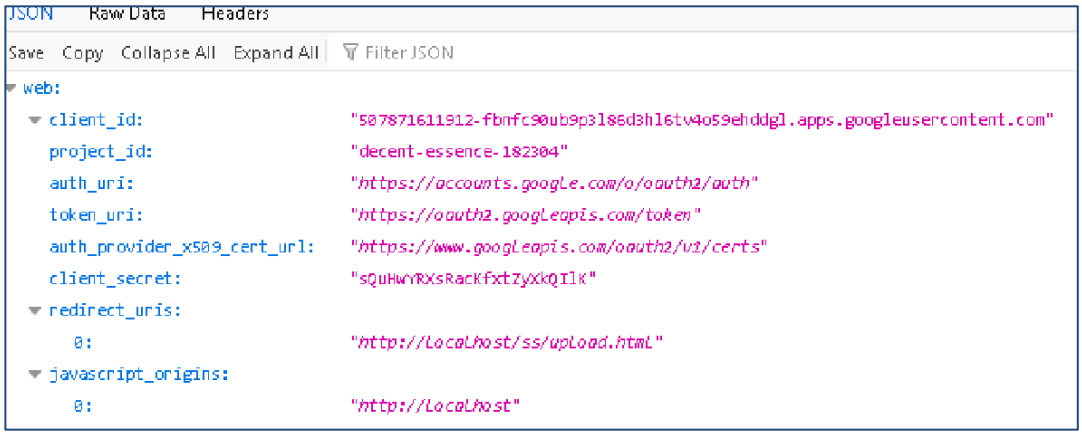

Figure 04: JASON file output

Once the API is created as depicted above, next step is to create the web application for which the API is connected.

2. Integrating Web Application with API

There is a button click process in index page for authenticating users through google account. When the button is clicked by, the respective user will be redirected to the page where user is authenticated;

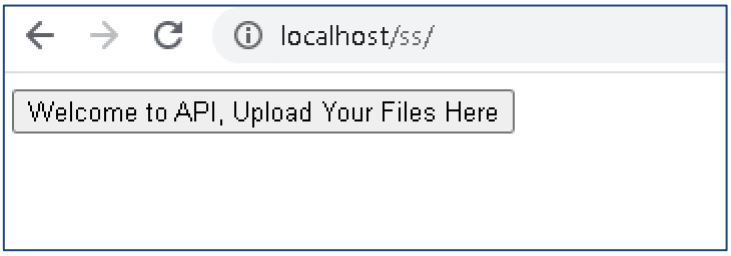

Figure 05: File uploading

Below is the code that was used to developed the index page;

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Git Login App</title>
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
    <script src="main.js"></script>
</head>
<body>
  <div>
      <button id="login">
          Welcome to API, Upload Your Files Here
      </button>
  </div>
</body>
</html>
```

Figure 06: Code: indexing page

Index.html file is located in the same main.js file. Once user clicks the button, file uploading wizard (to the drive) will be appeared by redirecting as per the given permission.

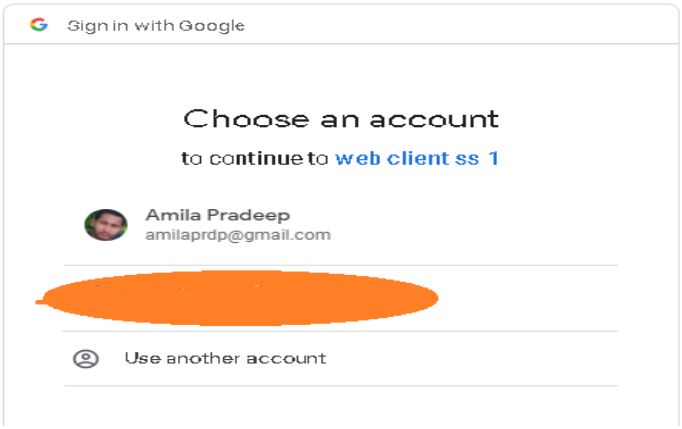

Figure 06: Authenticating

When a user clicks on his account, he will be given an error message informing "Google hasn't verified this app". This is mostly due to the program was created as a research project on a personal computer and has not yet been checked by Google. To get rid of this alert, go through the Google verification process, which verifies the security of the code used to create the app as well as the app's authenticity.

Users of the application have the option of clicking the continue button, which will redirect them to a page asking them to confirm their option.

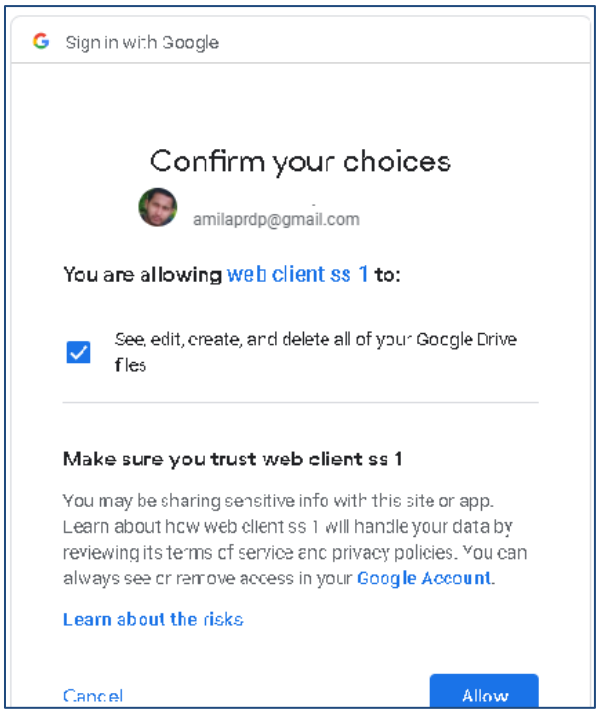

Figure 07: Permitting access

Then user has to allow this app to access drive by clicking allow;

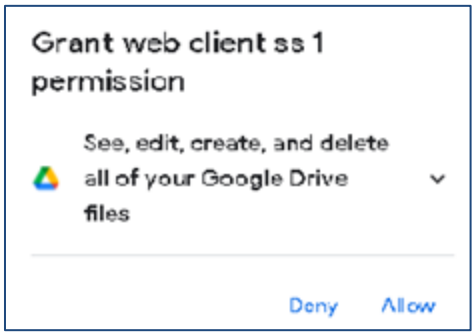

Figure 07: Permitting access

Here, user is given a warning message saying you are going to provide permissions to access the service however; this warning message comes at the first time only. If the permitting process is success, user is redirected to the page where he/she is able to upload the file.

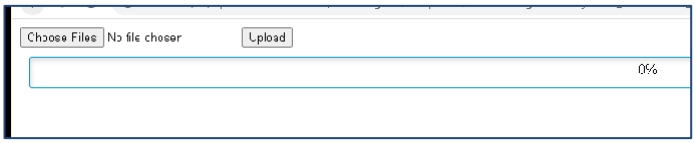

Figure 08: Choose files

Now, user is able to upload many file as he/she wishes from the personal computer to the given cloud location.

Here, newly created API will be used to load files and integrate the web application.

```
$(document).ready(function(){


    const urlParams = new URLSearchParams(window.location.search);
    const code = urlParams.get('code');
    const redirect_uri = "http://localhost/ss/upload.html" // replace with your redirect_uri;
    const client_secret = "sQuHwYRXsRacKfxtZyXkQIlK"; // replace with your client secret
    const scope = "https://www.googleapis.com/auth/drive";
    var access_token= "";
    var client_id = "507871611912-fbnfc90ub9p3l86d3hl6tv4o59ehddgl.apps.googleusercontent.com"// replace it with your client id;

```

Figure 09: API loading

When a user clicks the upload button after adding a file to be uploaded, the file is successfully uploaded into Google Drive.

APPENDIX:

01. Mail.JS

```
$(document).read(function(){

Var

clientId= "507871611912-fbnfc90ub9p3l86d3hl6tv4o59ehddgl.apps.googleusercontent.com";

    var redirect_uri = "http://localhost/ss/upload.html"; var scope =
    "https://www.googleapis.com/auth/drive";var url = "";

    $("#login").onclick(function(){ signIn(client_ID,redirect_uri,scope,url);

    });

    function signIn(client_ID,redirect_uri,scope,url){

      url = "https://accounts.google.com/o/oauth2/v2/auth?redirect_uri="+redirect_uri

      +"&prompt=consent&response_type=code&client_id="+client_ID+"&scope="+scope

      +"&access_type=offline";
      window.location = url;

    }

});
```

02. Upload.html

```
<html>
<head>
     <title>Google Drive File Upload</title>
   <link rel="stylesheet" type="text/css" media="screen" href="upload.css" />
   <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
   <script src="upload.js"></script>
</head>
<body>
  <div>
    <input id="files" type="file" name="files[]" multiple/>
    <button id="upload">Upload</button>
    <div id="progress-wrp">
```

```
      <div class="progress-bar"></div>
      <div class="status">100%</div>
    </div>
   </div>
   <div id="result">
   </div>
 </body>
 </html>
```

03. Upload.js

```
 $(document).ready(function(){
    const URLParameters = new URLSearchParams(window.location.search);
    const code = URLParameters .get('code');
    const redirect_URL = "http://localhost/ss/upload.html"
    const client_SECRET = "sQuHwYRXsRacKfxtZyXkQIlK"; const scope
 ="https://www.googleapis.com/auth/drive";
    var
    access_token= "";
    var client_ID =
 "507871611912-fbnfc90ub9p3l86d3hl6tv4o59ehddgl.apps.googleusercontent.com" with your
 client id;
    $.ajax({
       type: 'POST',
       url:
       "https://www.googleapis.com/oauth2/v4/token",
       data: {code:code
          ,redirect_uri:,
          client_secret:client_secret,
       client_id:client
       _ID,
       scope:scope,
       grant_type:"authorization_cod
       e"},dataType: "json",
       success: function(resultData) {
         localStorage.setItem("accessToken",resultData.access_token);
         localStorage.setItem("refreshToken",resultData.refreshToken);
         localStorage.setItem("expires_in",resultData.expires_in);
         window.history.pushState({}, document.title, "/GitLoginApp/" + "upload.html");
       }
  });
    function stripQueryStringAndHashFromPath(url) {
       return url.split("?")[0].split("#")[0];
    }
    var upload = function (file) {
       this.file = file;
    };
```

```
Upload.prototype.getType = function() {
    localStorage.setItem("type",this.file.typ
    e);return this.file.type;
};
Upload.prototype.getSize = function() {
    localStorage.setItem("size",this.file.size
    );return this.file.size;
};
Upload.prototype.getName = function() {
    return this.file.name;
};
Upload.prototype.doUpload = function ()
    {var that = this;
    var formData = new FormData();
        formData.append("file", this.file, this.getName());
    formData.append("upload_file", true);
    $.ajax({
        type: "POST",
        beforeSend: function(request) {
            request.setRequestHeader("Authorization", "Bearer" + " " +
localStorage.getItem("accessToken"));
        },
        url: "https://www.googleapis.com/upload/drive/v2/files",
        data:{
            uploadType:"media"
        },
        xhr: function () {
            var myXhr =
            $.ajaxSettings.xhr();if
            (myXhr.upload) {
                myXhr.upload.addEventListener('progress', that.progressHandling, false);
            }
            return myXhr;
        },
        success: function (data) {
            console.log(data);
        },
        error: function
            (error) {
            console.log(error)
            ;
        },
        async: true,
        data:
        formData,
        cache: false,
        contentType:
        false,
```

```
            processData:
            false,   timeout:
            60000
        });
    };

    Upload.prototype.progressHandling = function
        (event) {var percent = 0;
        var position = event.loaded ||
        event.position;var total = event.total;
        var progress_bar_id = "#progress-
        wrp";if (event.lengthComputable) {
            percent = Math.ceil(position / total * 100);
        }
        // update progressbars classes so it fits your code
        $(progress_bar_id + " .progress-bar").css("width", +percent + "%");
        $(progress_bar_id + " .status").text(percent + "%");
    };

    $("#upload").on("click", function (e) {
        var file =
        $("#files")[0].files[0];var
        upload = new Upload(file);
      // maby check size or type here with upload.getSize() and upload.getType()

      // execute upload
      upload.doUpload();
   });

});
```