MS21908910 – M.R.M. Razmeen

MS21909214 – V. Aperame

MS21908392 - Basith Abusalih

1st Year 1st Semester MSc Programme (Cyber Security)

Software Security

Assignment 2

The application which we made for this assignment, is used for to upload and share the files in Google Drive. To do the application, the platform used was "Google Cloud Platform" (GCP).

For the front end development, to view the created application, only html and JavaScript technologies are used. These are developed with the help of "IntelliJ". Also, these files can be viewed by the end users in the localhost of their own Personal Computer (PC).

Step by Step Process of the Developed Application

The following (Figure 1) shows the way how our application runs.

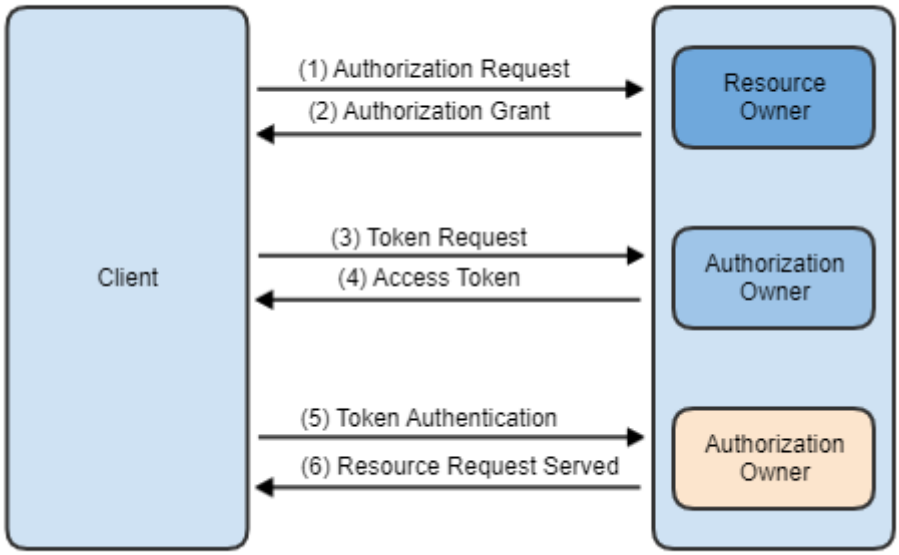

Fig 1: The flow our application works – OAuth Access

Our application was mainly developed to get the access for the Google Drive, in order to upload the files. In the above (Fig 1), the "Resource Owner" depicts the person who uses our application. "Authorization Owner" depicts Google. "Client" depicts the app, which we developed for this assignment.

The workflow of our developed application as follows. At first most, the Authorization Request is send from Client (Developed Application) to the Resource Owner (Person who uses the

application). Then as the acknowledgement, the authorization will be granted to the Client from the Resource Owner.

Then, as the Second Step, when the authorization is granted, again the Token Request is send to the Authorization Owner (Google). Next, the Google provides an Access Token to the Client (The developed app).

As the Third procedure, again the Client provides the Token Authentication to the Authorization Owner. So, finally, the Authorization Owner provides the Resource Requested to the Client. As the result of this procedures, finally, the Client can easily access the files which needed.

Creating an OAuth Client in GCP

The name of our created application is "test". This created application is used for to share the files in Google Drive. As the platform for this application, the "Google Cloud Platform" is used.

Under the "Credentials" tab as shown in (Figure 2), the OAuth Credentials can be created. So, under the "Credentials" tab, the Client ID is created for the web application. Also, at this mean time, the "Client ID" as well as the "Client Secret" are also created. "Creation Date" is also mentioned. In this tab, the OAuth Credentials are being configured for this web application.

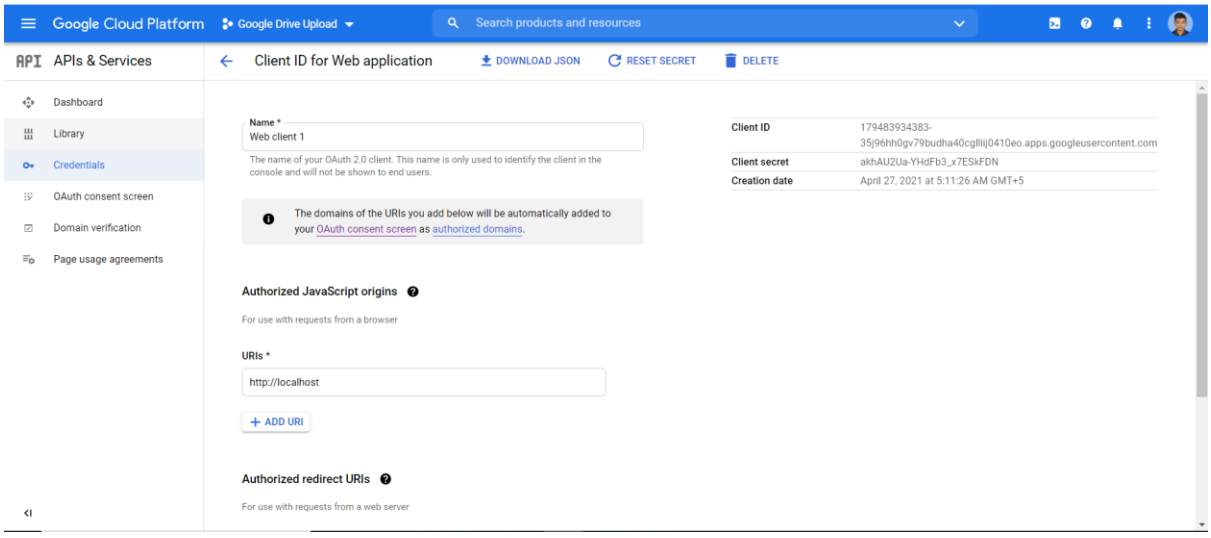

Fig 2: "Credentials" Tab

Also, the URLs which are needed to run this developed application are also configured. This is shown in (Figure 3).

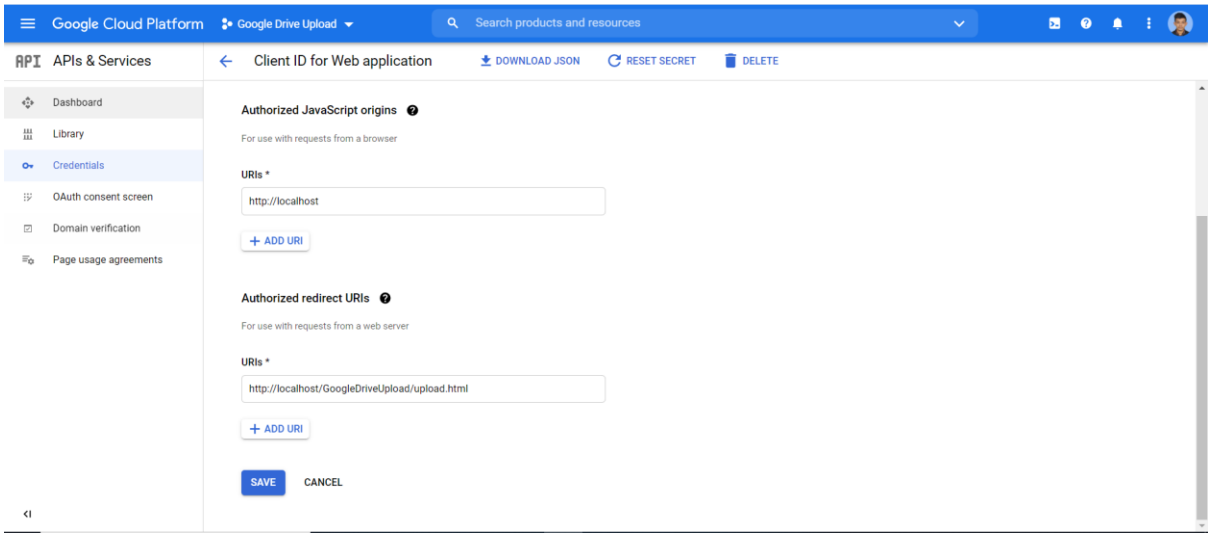

Fig 3: "URLs" Configuration

As shown in (Figure 4) the needed OAuth Credentials are fully configured. And the following shows the overview picture of it.

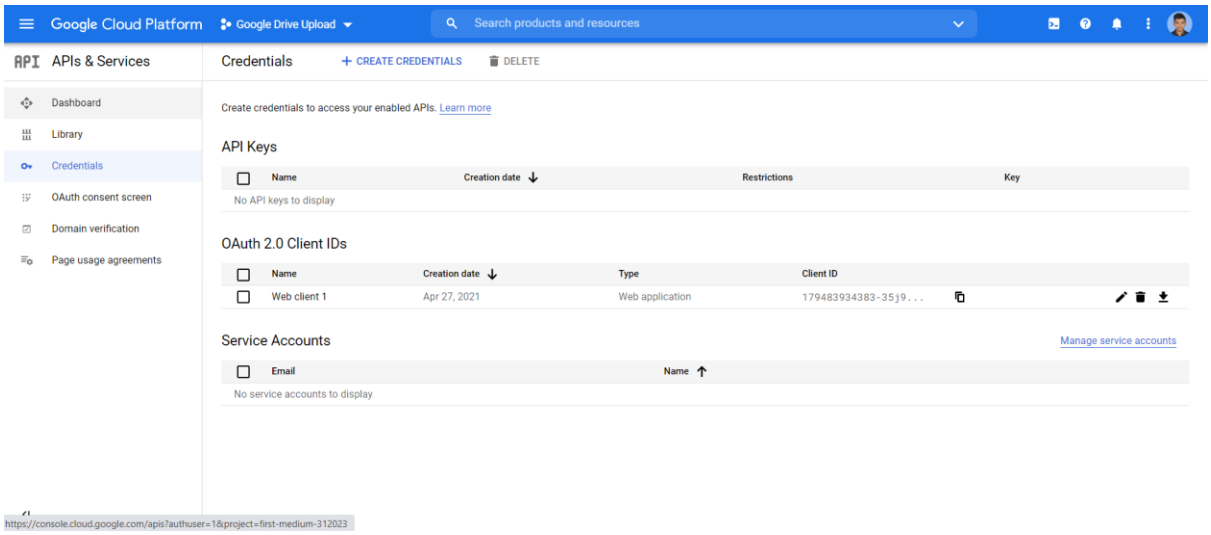

Fig 4: Overview of OAuth Credentials Are Configured

Next, under the tab named as "OAuth consent screen", can add any number of users as we wish. So that, the users that we add will get the access, in order to access the files. So, to add the users, the button named as "Add Users" is used and this is shown in (Figure 5).

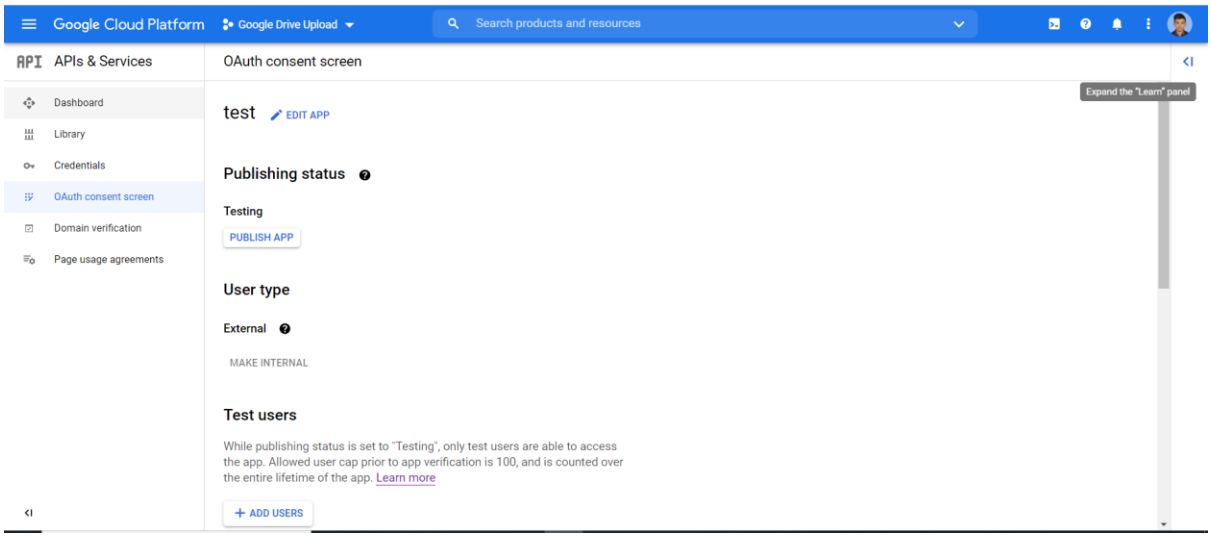

Fig 5: Add Users

Also, while testing the application the “Test Users” can get involved. So for this purpose also, can add any number of users as we like. This is shown in (Figure 6).

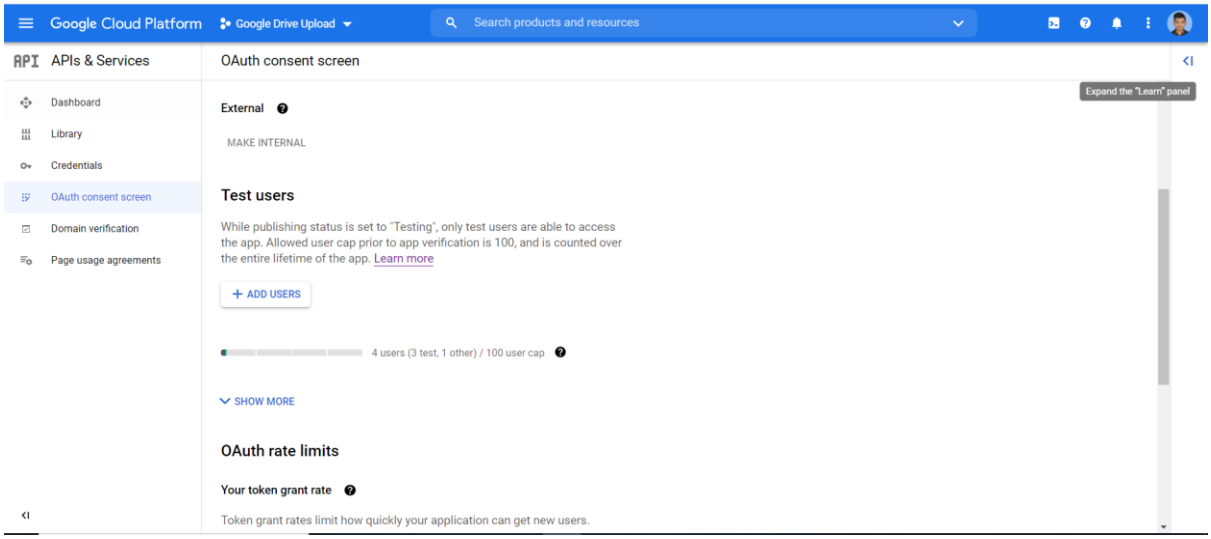

Fig 6: Add “Test users”

Demo of The Created Application

HTML and JavaScript technologies are used for the Front end development of the created application.

So, when the client/ end user run the “index.html” page in the localhost of their Personal Computer (PC), it shows a button known as “Upload Files to Drive”, in order to upload the files to the Google Drive. (Figure 7) depicts this.

Fig 7: “Upload Files to Drive” Button

Next, when the user click the button named as “Upload Files to Drive”, it will redirect to the Authentication Server in Google. This is shown in (Figure 8).

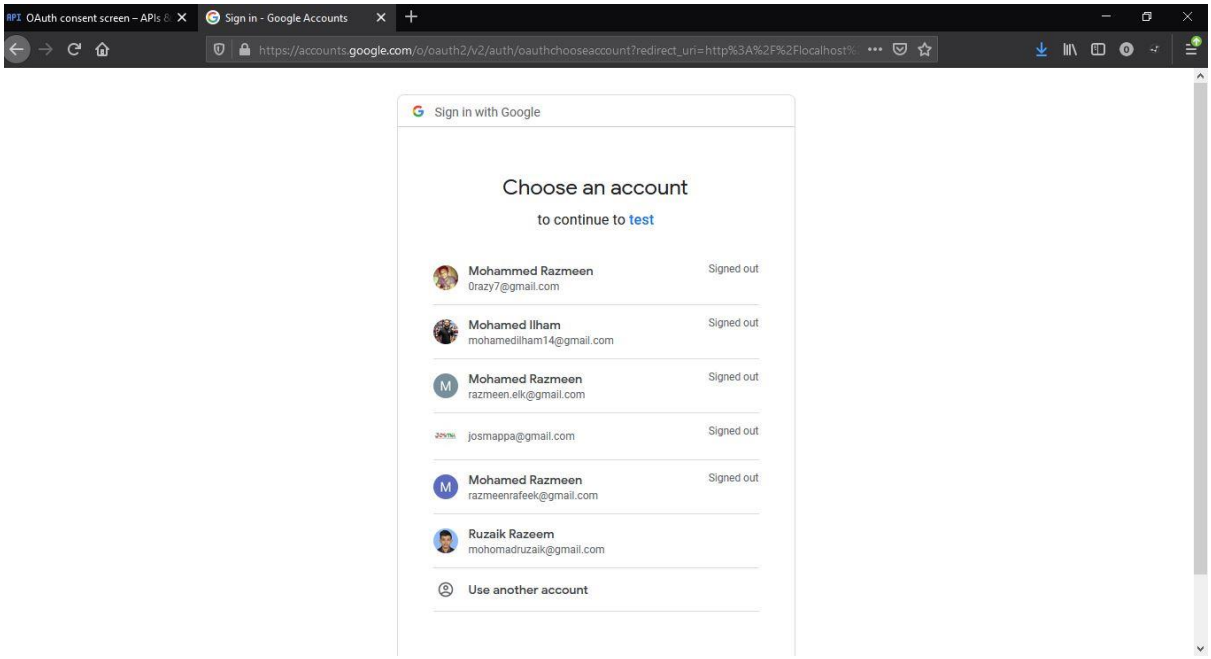

Fig 8: Redirect to Authentication Server

Then, the end user need to select any account and then it asks as follows, “Google hasn’t verified this app”. So, when the end user clicks the “Continue” button, Google grants the permission to the end user to handle this developed application. This is shown in (Figure 9).

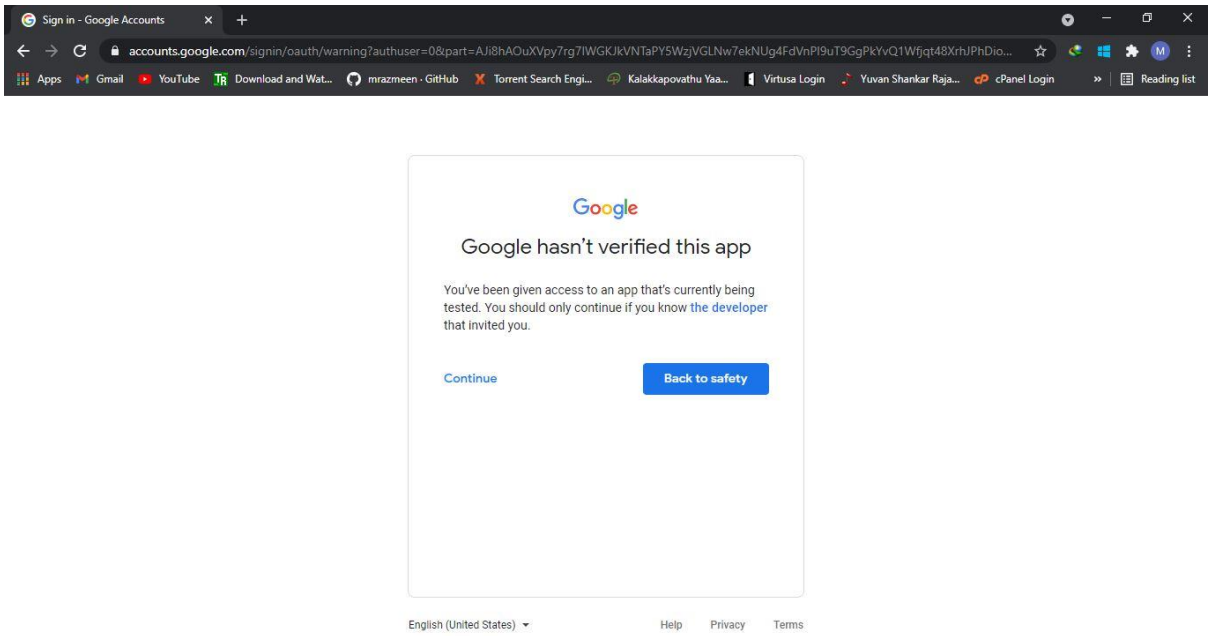

Fig 9: Google grants the permission

Next, it asks whether the end user has the rights to “edit, create and delete all the Google Drive Files”, in order to grant the permission for the developed application named as “test”. This is depicted in (Figure 10).

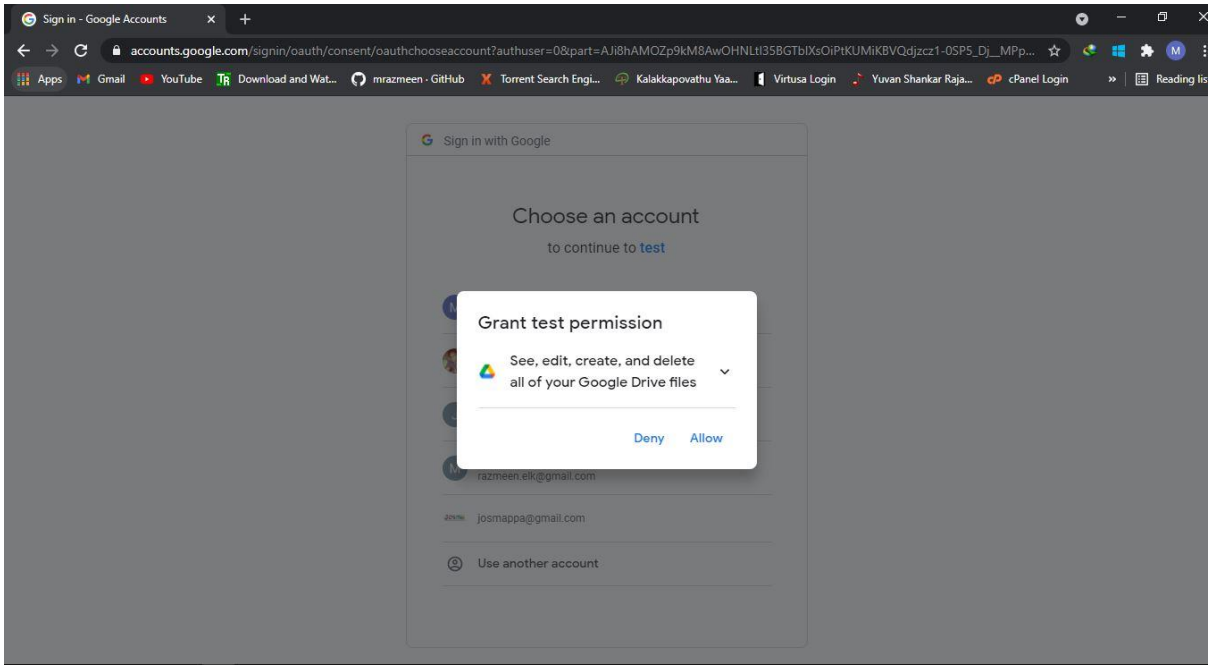

Fig 10: Grant Test Permission

Finally, when the end user select the button named as "Allow", it grants all the access. (Figure 11) depicts this.

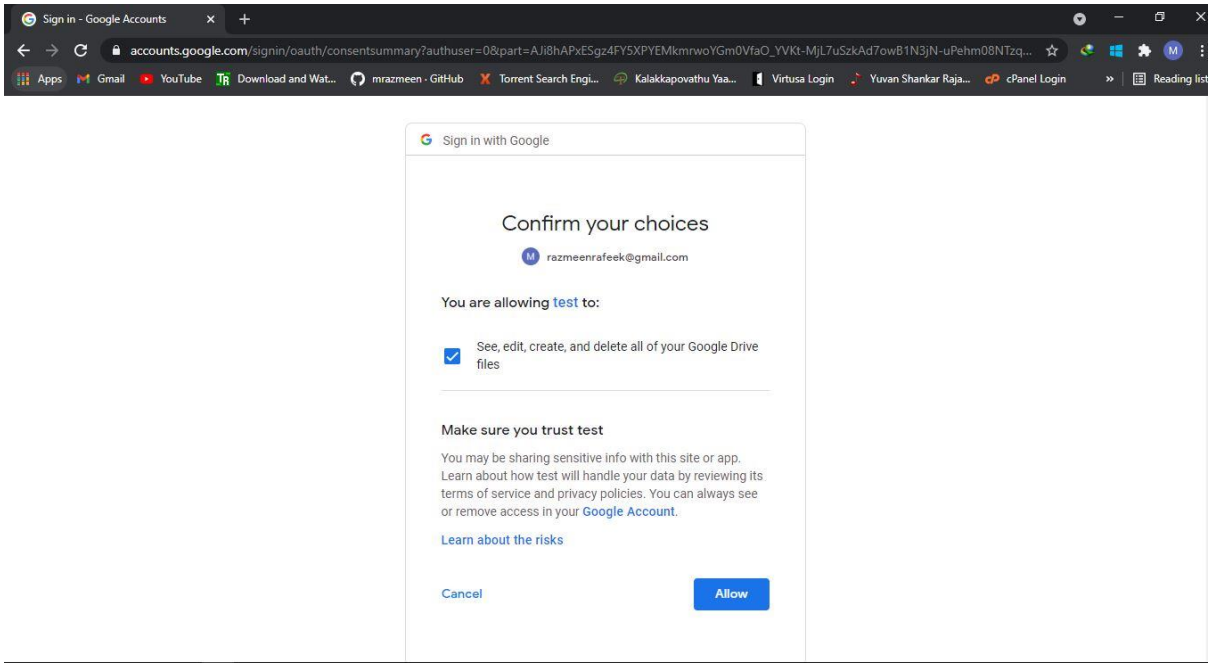

Fig 11: Click the "Allow" Button

Next, when the end user clicked the "Allow" button, it will redirect to the page where the user need to upload the files. This is shown in (Figure 12).

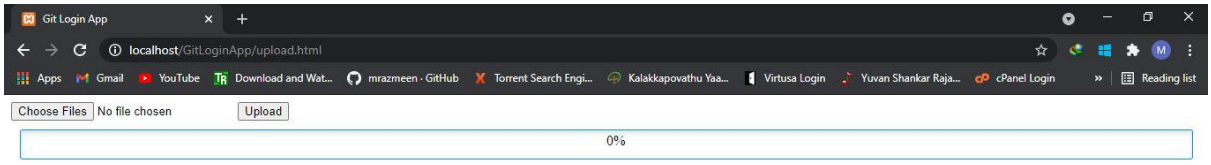

Fig 12: Redirect to the page to upload files

Then, when the end user clicked the button named as “Choose Files”, can get the screen as shown in (Figure 13), where the end user can upload whatever the files which the end user needed.

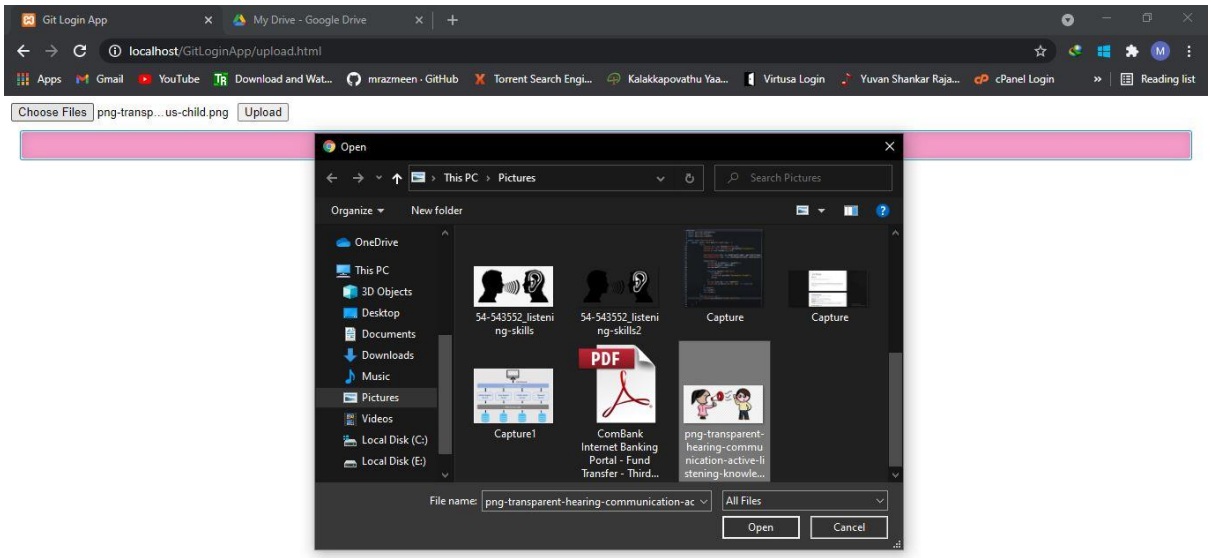

Fig 13: Choose the file

Finally, as shown in (Figure 14), selected the needed file by the end user, and clicked the button named as “Upload”. As shown in Figure 14, finally the needed file will be uploaded by the end user.

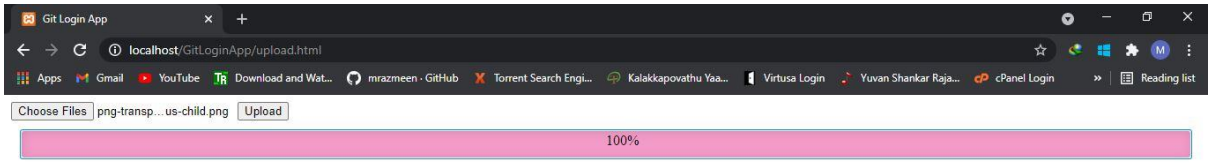

Fig 14: Uploaded the needed file by the end user

Then, in order to confirm whether the files have been uploaded or not, when the end user go to the Google Drive of the particular account (Already Signed), and refresh it, can able to see that particular file which they selected as uploaded in the Google Drive. This is shown in (Figure 15).

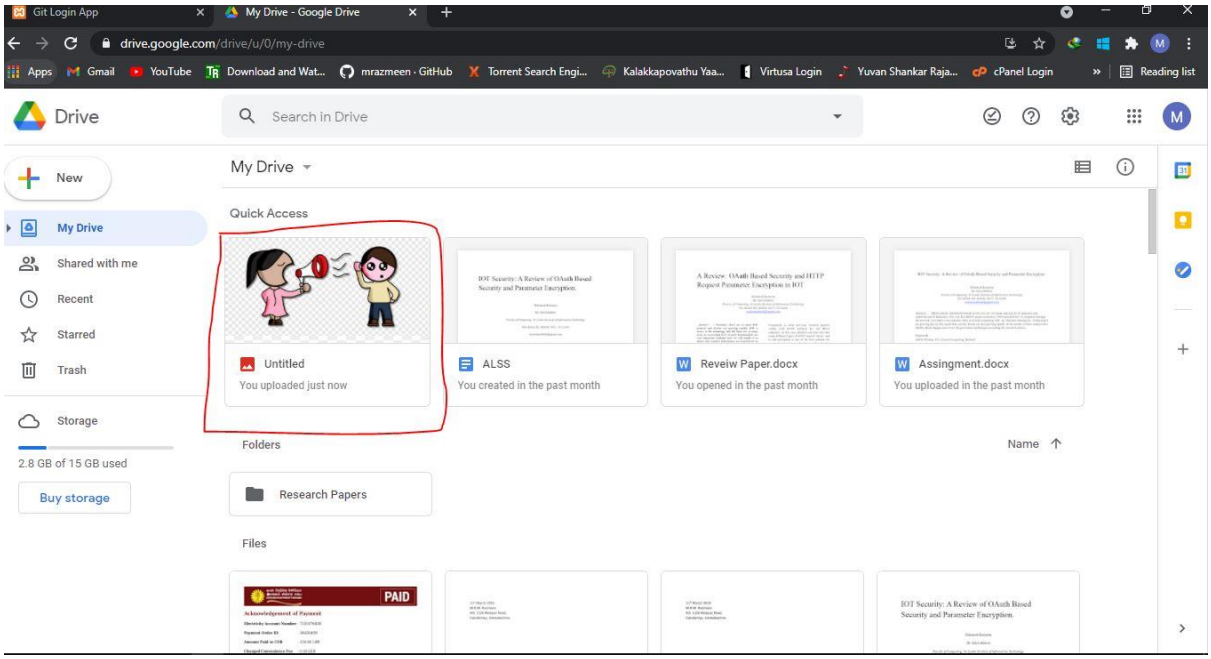

Fig 15: Uploaded file by the end user in Google Drive

Appendix

We have developed this application using JavaScript and HTML. The code snippets are bellow.

1. HTML

A. 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"></sc
ript>
    <script src="main.js"></script>
</head>
<body>
<div>
    <button id="login">
        Upload Files to Drive
    </button>
</div>
</body>
</html>
```

B. File Upload 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">
    <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"></sc
ript>
    <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">0%</div>
    </div>
</div>
<div id="result">

</div>
</body>
</html>
```

JavaScript

A. Main.JS

```
// clien id - 179483934383-
35j96hh0gv79budha40cglliij0410eo.apps.googleusercontent.com
// client secret - akhAU2Ua-YHdFb3_x7ESkFDN

$(document).ready(function(){

    // client id of the project

    var clientId = "179483934383-
35j96hh0gv79budha40cglliij0410eo.apps.googleusercontent.com";

    // redirect_uri of the project

    var redirect_uri = "http://localhost/GoogleDriveUpload/upload.html";

    // scope of the project

    var scope = "https://www.googleapis.com/auth/drive";

    // the url to which the user is redirected to

    var url = "";

    // this is event click listener for the button

    $("#login").click(function(){

        // this is the method which will be invoked it takes four
parameters

        signIn(clientId,redirect_uri,scope,url);

    });

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

        // the actual url to which the user is redirected to

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

+"&prompt=consent&response_type=code&client_id="+clientId+"&scope="+scope
            +"&access_type=offline";

        // this line makes the user redirected to the url

        window.location = url;



    }


});
```

Upload.JS

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


    const urlParams = new URLSearchParams(window.location.search);
    const code = urlParams.get('code');
    const redirect_uri = "http://localhost/GoogleDriveUpload/upload.html"
// replace with your redirect_uri;
    const client_secret = "akhAU2Ua-YHdFb3_x7ESkFDN"; // replace with your
client secret
    const scope = "https://www.googleapis.com/auth/drive";
    var access_token= "";
    var client_id = "179483934383-
35j96hh0gv79budha40cglliij0410eo.apps.googleusercontent.com"// replace it
with your client id;


    $.ajax({
        type: 'POST',
        url: "https://www.googleapis.com/oauth2/v4/token",
        data: {code:code
            ,redirect_uri:redirect_uri,
            client_secret:client_secret,
            client_id:client_id,
            scope:scope,
            grant_type:"authorization_code"},
        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.type);
        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();

        // add assoc key values, this will be posts values
        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();
    });



});
```