KURKULASOORIYA KVR **200332X**

WEEK 07: ONE RING TO BRING THEM ALL

REPORT

Since the management is now concerned about integrating their applications with a strong authentication mechanism, OAuth2 can be used to provide a secure and convenient solution which is a widely adopted standard that allows users to access their data and resources without sharing their passwords.

This report emphasizes the procedure of how the implementation of OAuth2 authentication of a simple login application is carried out and the challenges faced during the process along with how they have been overcome. The generated PHP code files are stored in the submission package.

Implementation of OAuth2 Authentication

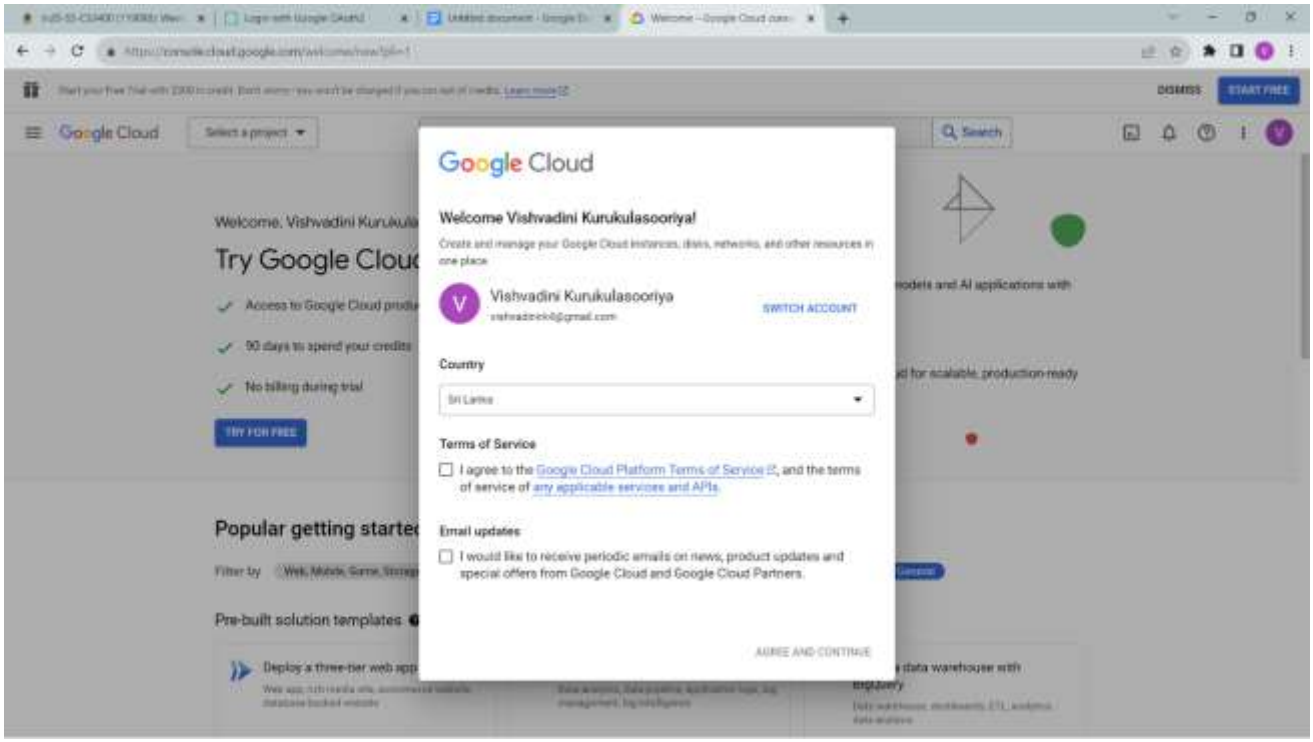
When creating a PHP-based login application with "Login with Google" functionality using OAuth2, the following steps were followed:

1. Set up a Google Cloud Platform project and configure OAuth2 credentials.

Google Cloud Platform (GCP) is a suite of cloud computing services offered by Google and supports OAuth 2.0, which is an open standard for authorization by providing OAuth 2.0 client libraries, Identity-Aware proxies, and OAuth 2.0 APIs.

During the Setting up and configuring process, it was needed to;

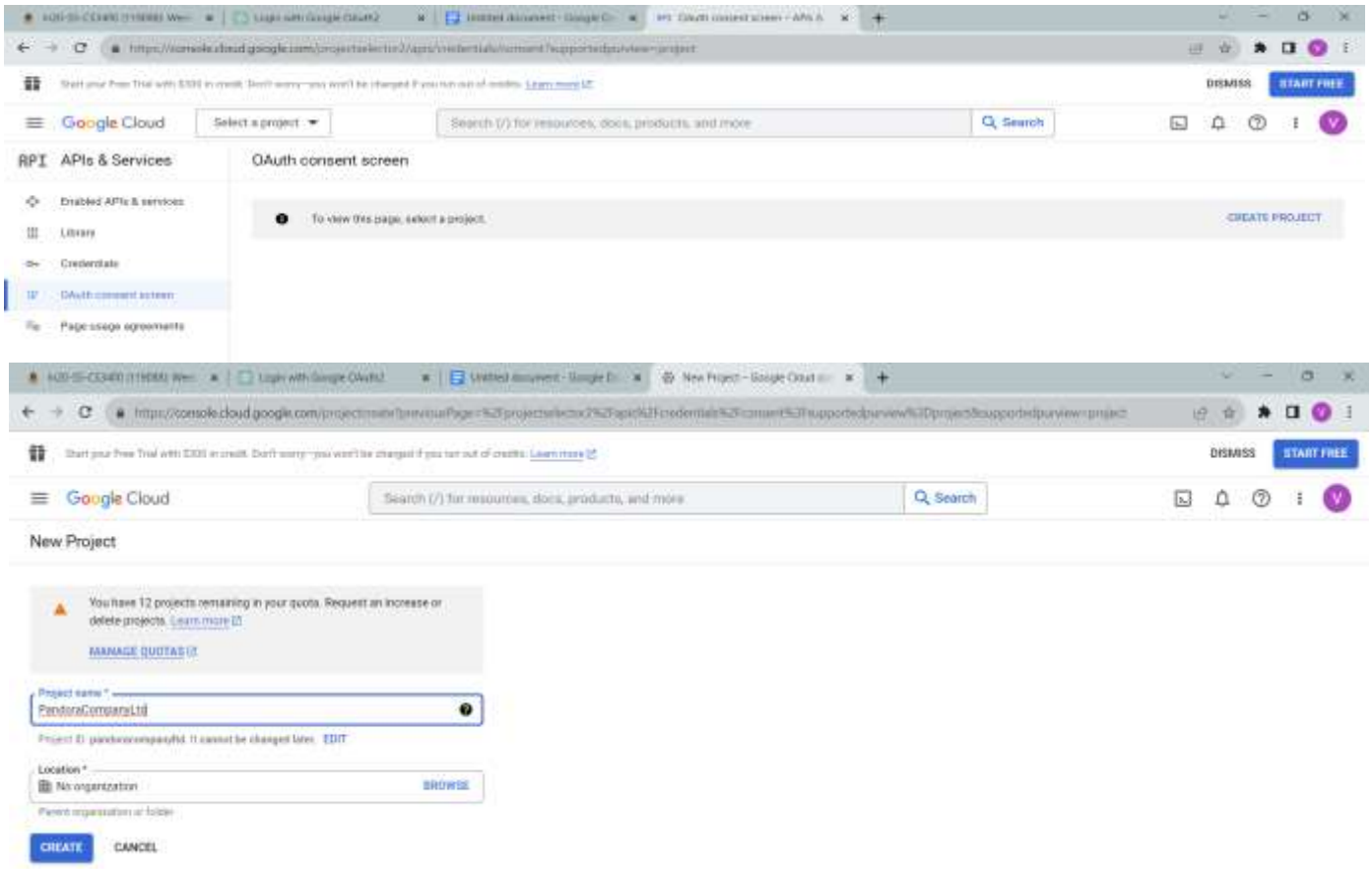
1.1 Go to the Google Cloud Console (https://console.cloud.google.com/).



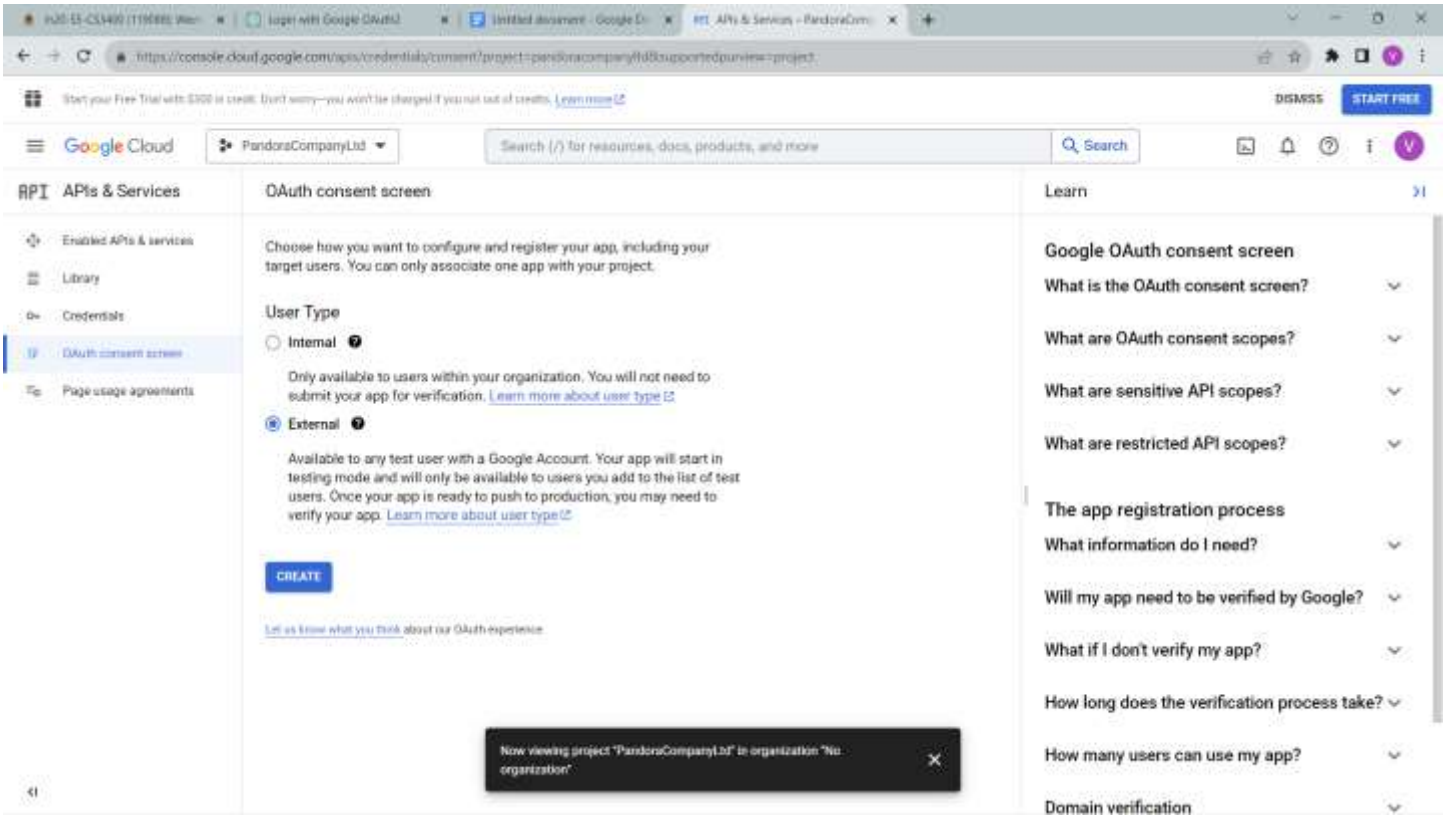
1.2

1.3

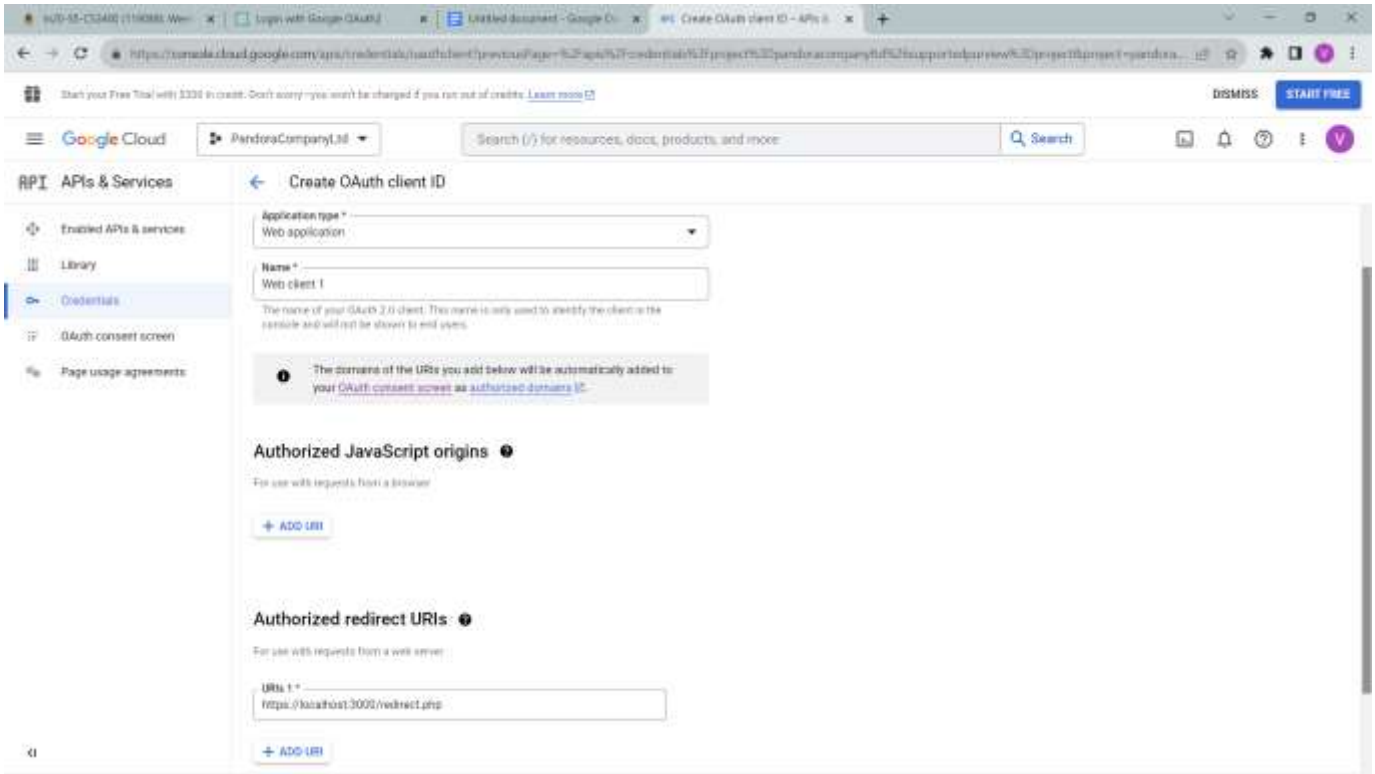
1.4 Create a new project by filling out the necessary details on project identification.



1.5 Select the created project from the menu and configure the consent screen.



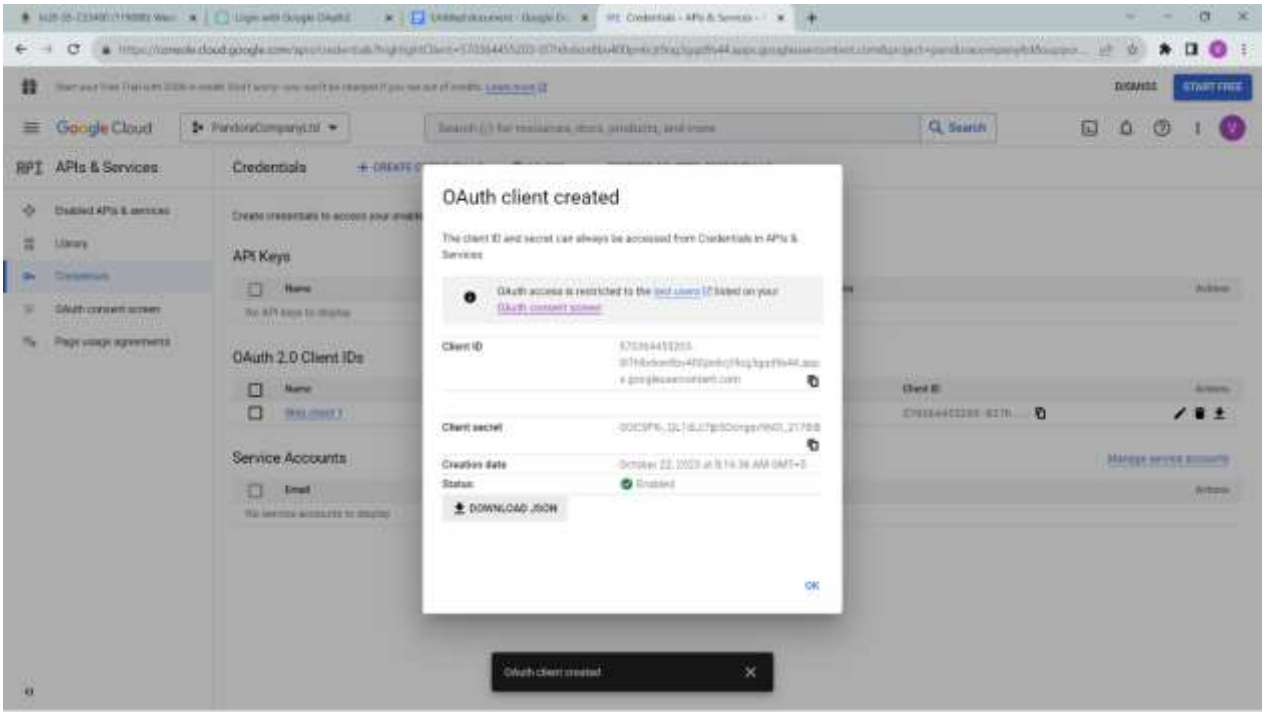
1.6 Navigate to the "APIs & Services" > "Credentials" section. Create OAuth client credentials and set the authorized redirect URIs to point to the redirect.php file as http://localhost:3000/redirect.php).



1.7 Save details and proceed.

2. Create the PHP files: login.php, redirect.php, and dashboard.php, and install dependencies.

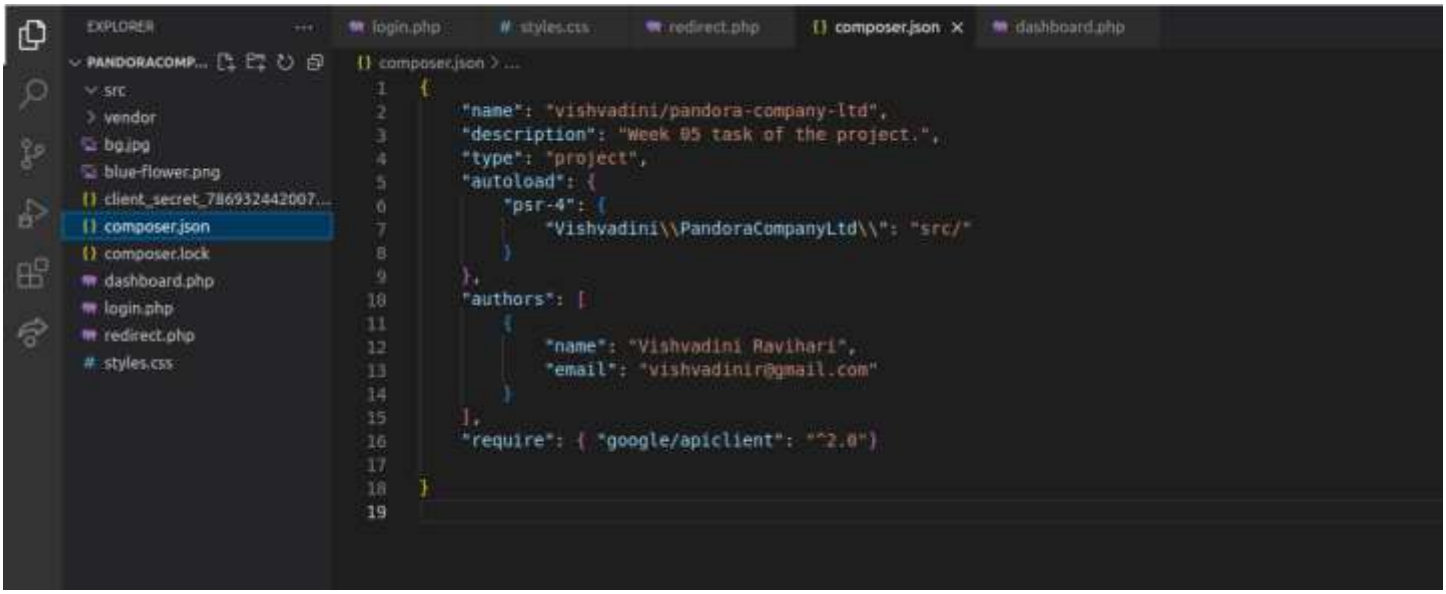
Once the initial configuration process is done, download the JSON object file that contains the Client ID info that includes client_id, client_secret, redirect_url, auth_url etc.



Then the development phase could be started in the Vscode IDE environment by performing the following necessary steps on developing the source code, installing PHP environment support dependency managers to run the PHP codes in the local machine.

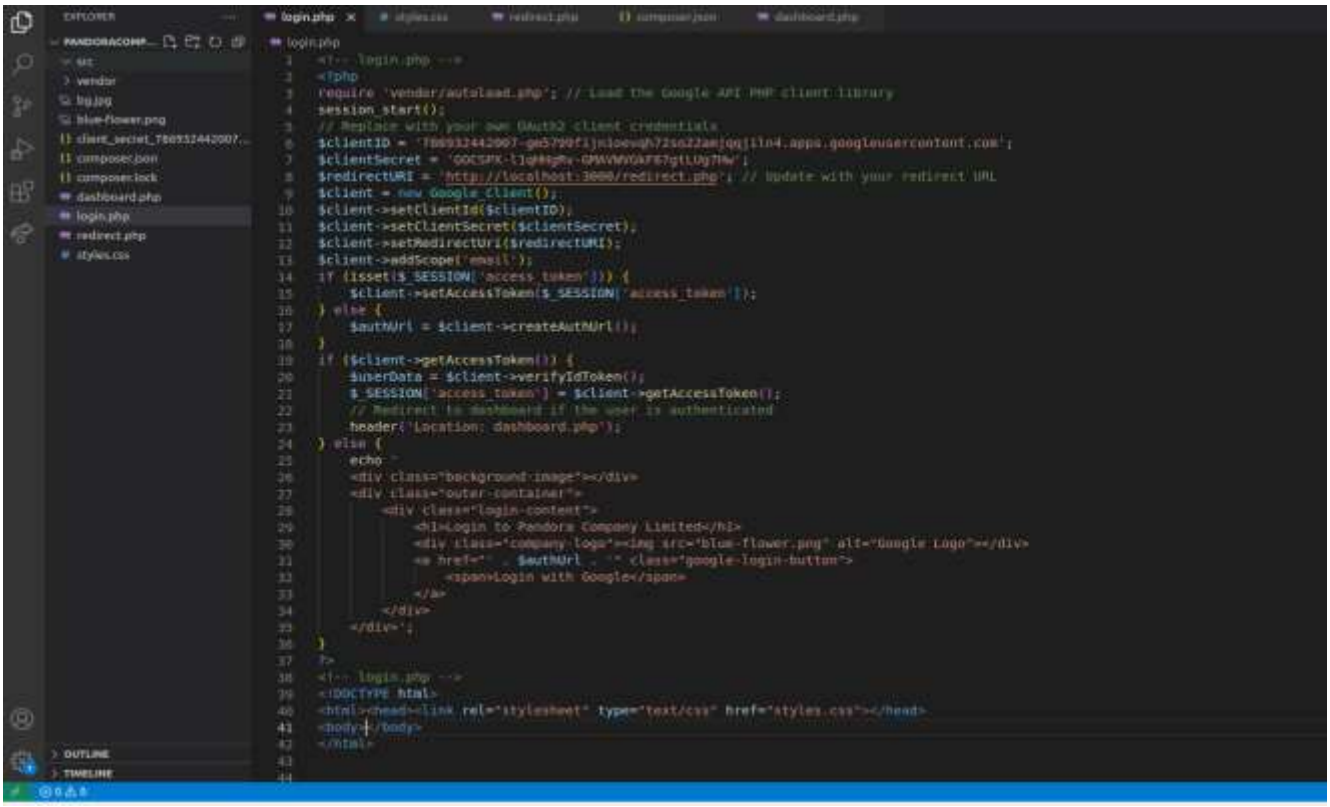
2.1 Open VsCode, set up a new workplace folder for the project, and include the downloaded client's secret in there.

2.2 Install, configure, and run composer, the dependency manager by including Google/apiclient library to be installed along with it.



3. Create 3 .php files for the application and implement their functionalities.

login.php; : A page displaying the "Login with Google" button



redirect.php; : A page handling the redirect from Google after successful authentication

```php
<?php
require 'vendor/autoload.php'; // Load the Google API PHP client library
session_start();

$clientID = '786932442087-gm5799fijn1oevqh72so22amjqqjiln4.apps.googleusercontent.com'; // Replace with your own client ID
$clientSecret = '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'; // Replace with your own client secret
$redirectURI = 'http://localhost:3000/redirect.php'; // Update with your redirect URL

$client = new Google_Client();
$client->setClientId($clientID);
$client->setClientSecret($clientSecret);
$client->setRedirectUri($redirectURI);

if (isset($_GET['code'])) {
    $token = $client->fetchAccessTokenWithAuthCode($_GET['code']);
    $_SESSION['access_token'] = $token;
    header('Location: dashboard.php');
}
?>
```

dashboard.php; A landing page after successful login, displaying a simple greeting message to the authenticated user.

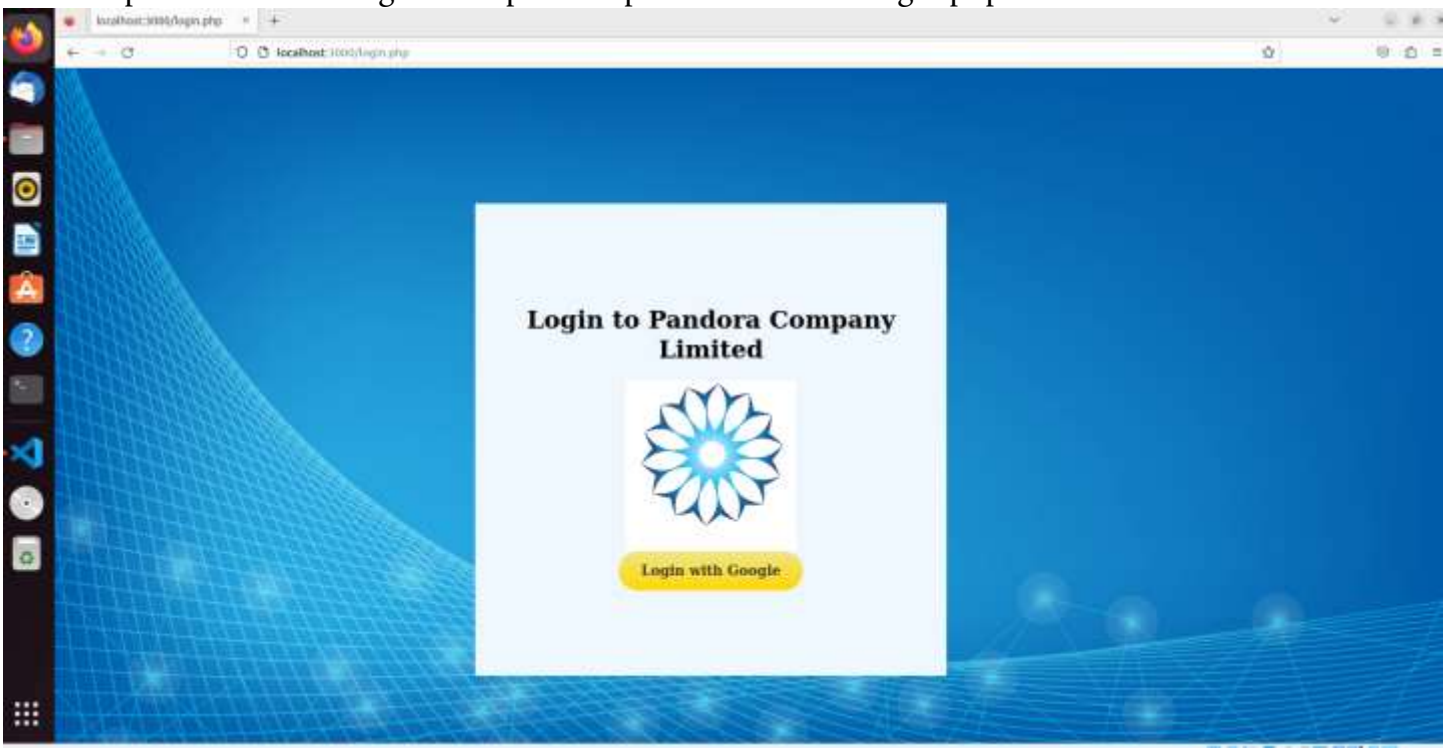
```php
<!-- dashboard.php -->
<?php
require 'vendor/autoload.php'; // Load the Google API PHP client library
session_start();
if (isset($_SESSION['access_token'])) {
    $clientID = '786932442087-gm5799fijn1oevqh72so22amjqqjiln4.apps.googleusercontent.com'; // Replace with your own client ID
    $clientSecret = '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'; // Replace with your own client secret
    $redirectURI = 'http://localhost:3000/redirect.php'; // Update with your redirect URL
    $client = new Google_Client();
    $client->setClientId($clientID);
    $client->setClientSecret($clientSecret);
    $client->setRedirectUri($redirectURI);
    $client->setAccessToken($_SESSION['access_token']);

    $oauth2 = new Google_Service_Oauth2($client);
    $userData = $oauth2->userinfo->get();
    if (isset($_POST['logout')) {
    // Logout button was clicked
        session_unset();
        session_destroy();
        header('Location: login.php');
        exit();
        }
    echo
    <div class="background-image"></div>
        <div class="outer-container">
            <div class="logout-container">
                <div class="login-content">
            <h1>Hello, $userName! </h1>
            <h1>Welcome to Pandora Company Limited !</h1>
            <div class="company-logo"><img src="blue-flower.png" alt="Google Logo"></div>
            <div class="logout-form"><form method="post" action="dashboard.php"><input type="submit" name="logout" value="Logout" class="logout-button">
            </form>
            </div></div></div></div>';
} else {
    header('Location: login.php');
}
?>
<!-- dashboard.php -->
<!DOCTYPE html>
<html><head><link rel="stylesheet" type="text/css" href="styles.css"></head><body></body>
</html>
```

The above code generation was done with the support of ChatGPT, and necessary changes were made to replace 'YOUR_CLIENT_ID' and 'YOUR_CLIENT_SECRET' with the generated Google OAuth2 credentials at step 1. HTML tags were included to set the correct visibility of components at the front end.

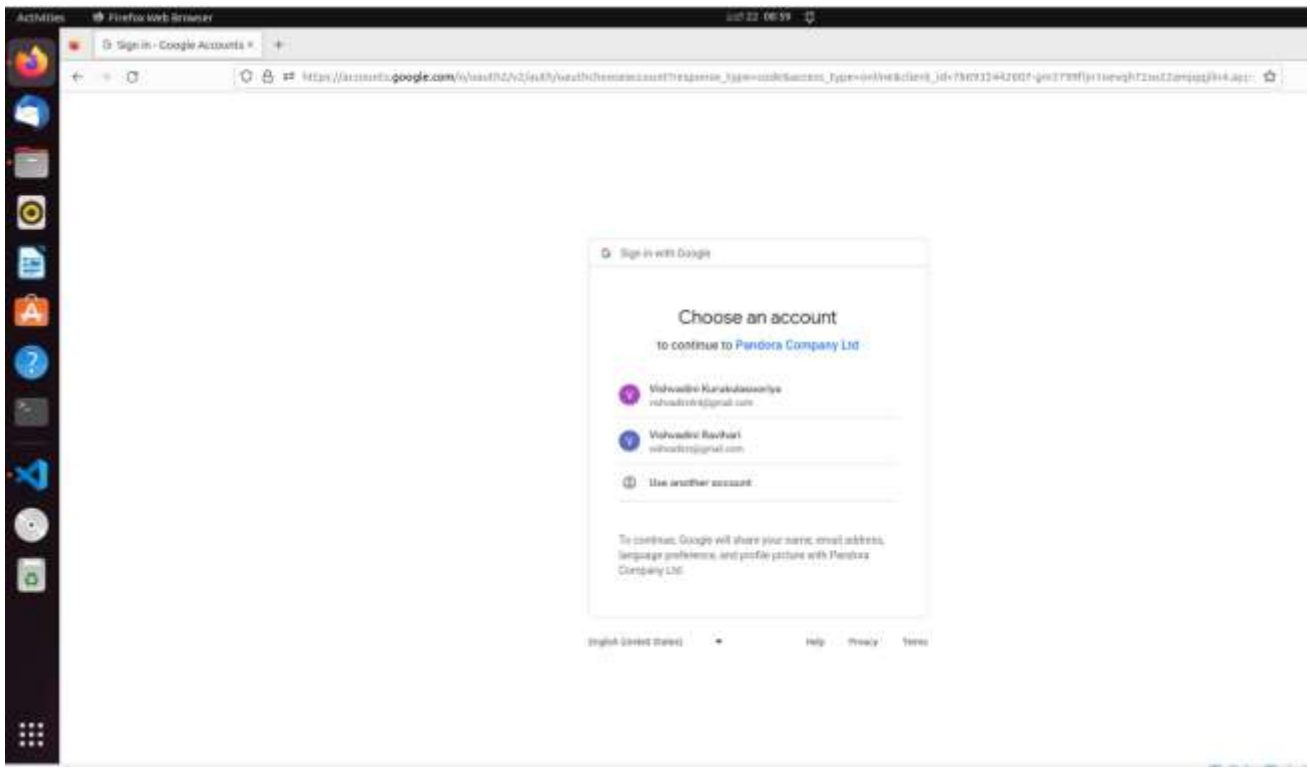
A styles.css file was to decorate the pages and the path was configured to a .css file in each .php file within <html><head> section of each of the above files.[1]

4. Run the login.php file by giving the command **php -S localhost:3000 -t** in the terminal.

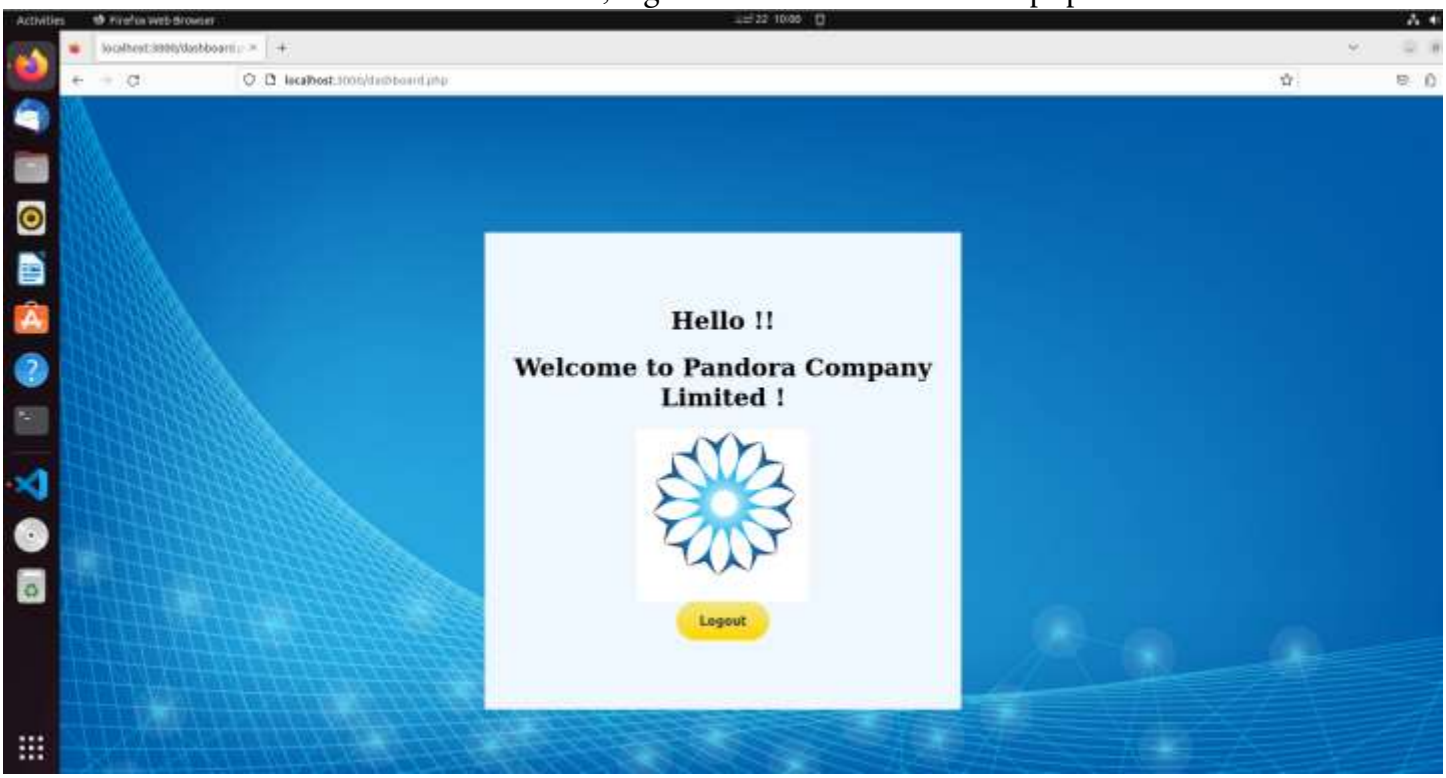
5. Open a browser and go to the path "http://localhost:3000/login.php



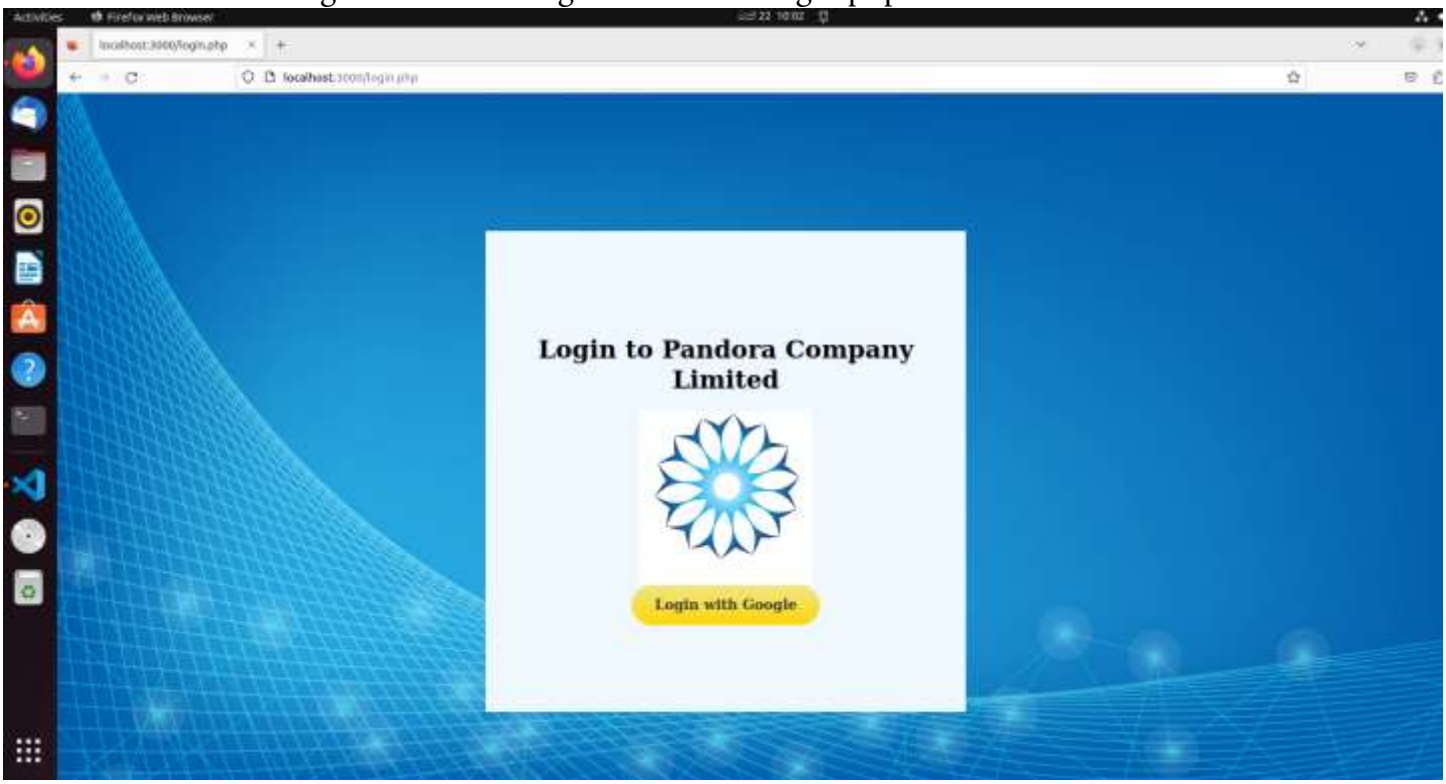
6. Click the "Login with Google" button.
7. Select a suitable Google account to log in and give the password.



8. When the authentication is successful, it gets redirected to dashboard.php.



9. Click on the "Log out" button to again redirect to login.php



Challenges Faced & How They Were Overcome

1. **There was an issue with installing the composer, the dependency manager. Thus, the guidance from ChatGPT was obtained with a step-by-step procedure.**

Solution:

First to run the following command to install the composer.

```
curl -sS https://getcomposer.org/installer | php sudo mv composer.phar /usr/local/bin/composer
```

Then, to create a composer.json file in the same directory and add the following dependency.

```
{"require": {"google/apiclient": "^2.0"}}
```

Here, "google/apiclient" package contains the Google Client

Run the composer with `composer install` and import the necessary classes from the library to the PHP scripts.

2. **Once successfully authenticated and redirected to the dashboard.php file, there was no option visible in the frontend window to return to login.php by logging out from the already signed-in account.**

Solution:

The following HTML code was implemented in dashboard.php to include the functionality of a log-out button.

```
<div class="logout-form"><form method="post" action="dashboard.php"><input type="submit" name="logout" value="Logout" class="logout-button">
</form>
</div>
```

3. **The "E: could not get lock/var/lib/dpkg/lock " and "E: Unable to acquire dpkg frontend lock" error message was generated when it was trying to execute the code for the first time.**

Solution:

The composer was reinstalled after rechecking the dependencies and then the system was rebooted and the code was executed. [2]

4. **For the web development process, PHP was not a very familiar language and thus, it was a bit difficult when debug the code.**

Solution:

Assistance from the copilot and ChatGPT was obtained for debugging purposes.

References:

[1] Colorlib, "HTML5 and CSS login Forms", Available at: https://colorlib.com/wp/html5-and-css3-login-forms/

[2] Abhishek Prakash on IT'S FOSS, "How to Fix 'E: Could not get lock/var/lib/dpkg/lock' Error in Ubuntu", 2022 December 30, Available at: https://itsfoss.com/could-not-get-lock-error/