5225 report

4.1 Team Report - 1000 words

Note that the tables, architecture diagram and references are excluded from the word count limit.

蓝色字部分记得要删掉

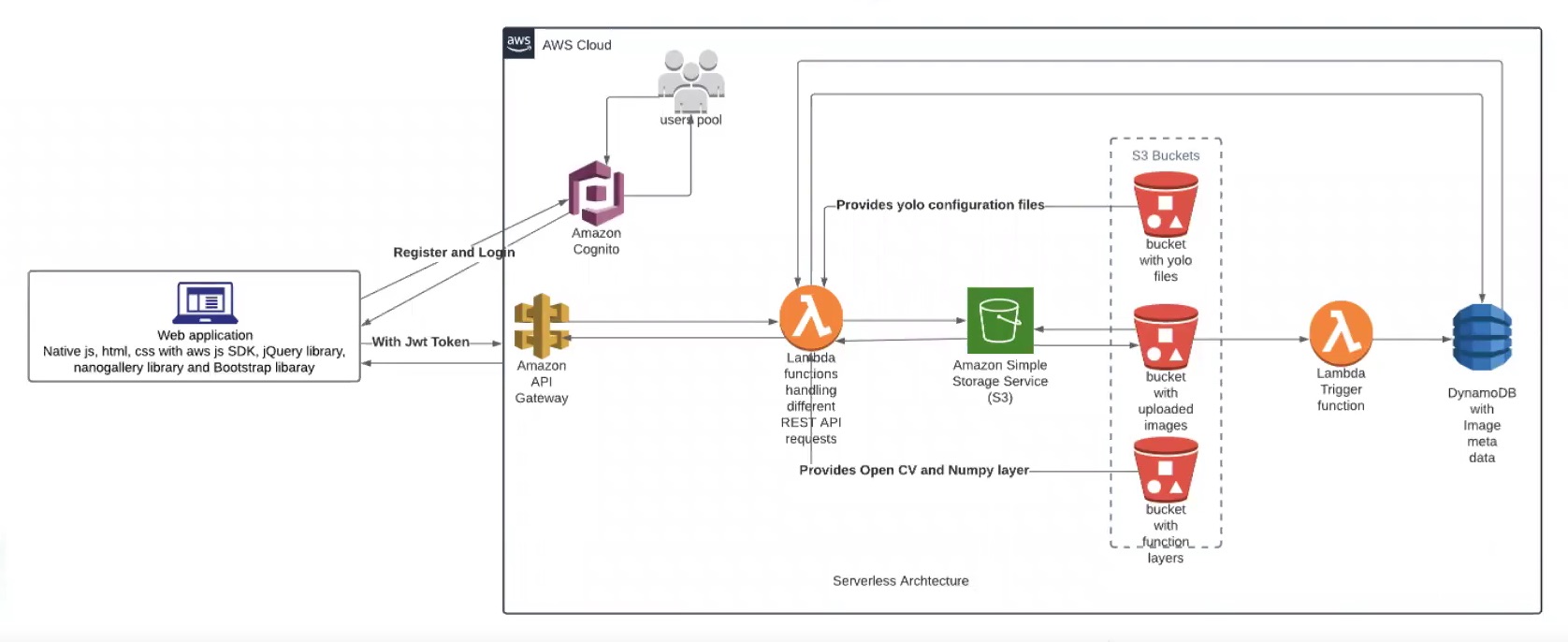
高亮的也

PPT:

<https://docs.google.com/presentation/d/1yqoAyTh2crS77fVEHIVWujWn94JFXrhEyHhWvLhLvnc/edit?usp=sharing>

1. Architecture Diagram

• Include an architecture diagram in the team report. For the architecture diagram, you must use AWS Architecture Icons (more info can be found here: https://aws.amazon.com/architecture/icons/).



*Figure 1.1 - Architecture*

1. Explanation:

• Explain your diagram by focusing on how your application works and how various components of the architecture interacts with each other. You should justify your design decisions and implementations, for example why specific AWS service was used, or why your database is designed in certain way.

The application will be separated into two parts: frontend and backend. The frontend includes the Web page UI implementation and design and makes the application UI interact with the backend. For the backend, the user will be authorized with an access token once gets registered and gets authority over the application via the Amazon congnito. The data will be stored in the S3 buckets, the lambda functions will handle different REST API requests, these two components provide OpenCV and Numpy layer to the application. The lambda functions will be achieved by the triggers. DynamoDB as a persistent NoSQL database, it has built-in security, backup and restore capabilities, which will allow the data to be stored in the form of a scaled-up hash table. Thus, data such as image can be stored as:

{

"url": "https://imagetodetect.s3.amazonaws.com/1.jpg",

"type": 1,

"tags": [

"person",

"ax"

]

}, which acutely improves the efficiency and simplicity of the lambda function code. The interaction between frontend and backend relies on the URL, access token, and id token. The access token will let users get permission to the application, the id token will be stored and specified by the cloud database, and URLs will link the UI with the AWS system to realize the interaction.

Functions include 1. Image upload function, which will allow users to upload images into the database, with URL, name and tags. 2. Delete image function will allow users to delete the appointed image from the database by inputting the URL. 3. Find images based on the tags function, the function will first let the user appoint a tag, then return images with the same tag to the user in the form of Figure 4.8. 3. Find images based on the tags of an image function, similarly to the former function, yet it will get the appointed first then detect the tag and return the same dormant result, yet the uploaded image will not be stored in the database. 4. Manual addition or removal of tags function will let the user appoint an image by a specific tag or URL, then search and find the image in the database, and finally delete the image from the database.

1. Task Allocation Table

• Include a table to describe the role of each team member in your team report, You can provide a three-column table in your report that shows student name and id, percentage of contribution and elements of the project the member contributed (maximum of 50 words per member).

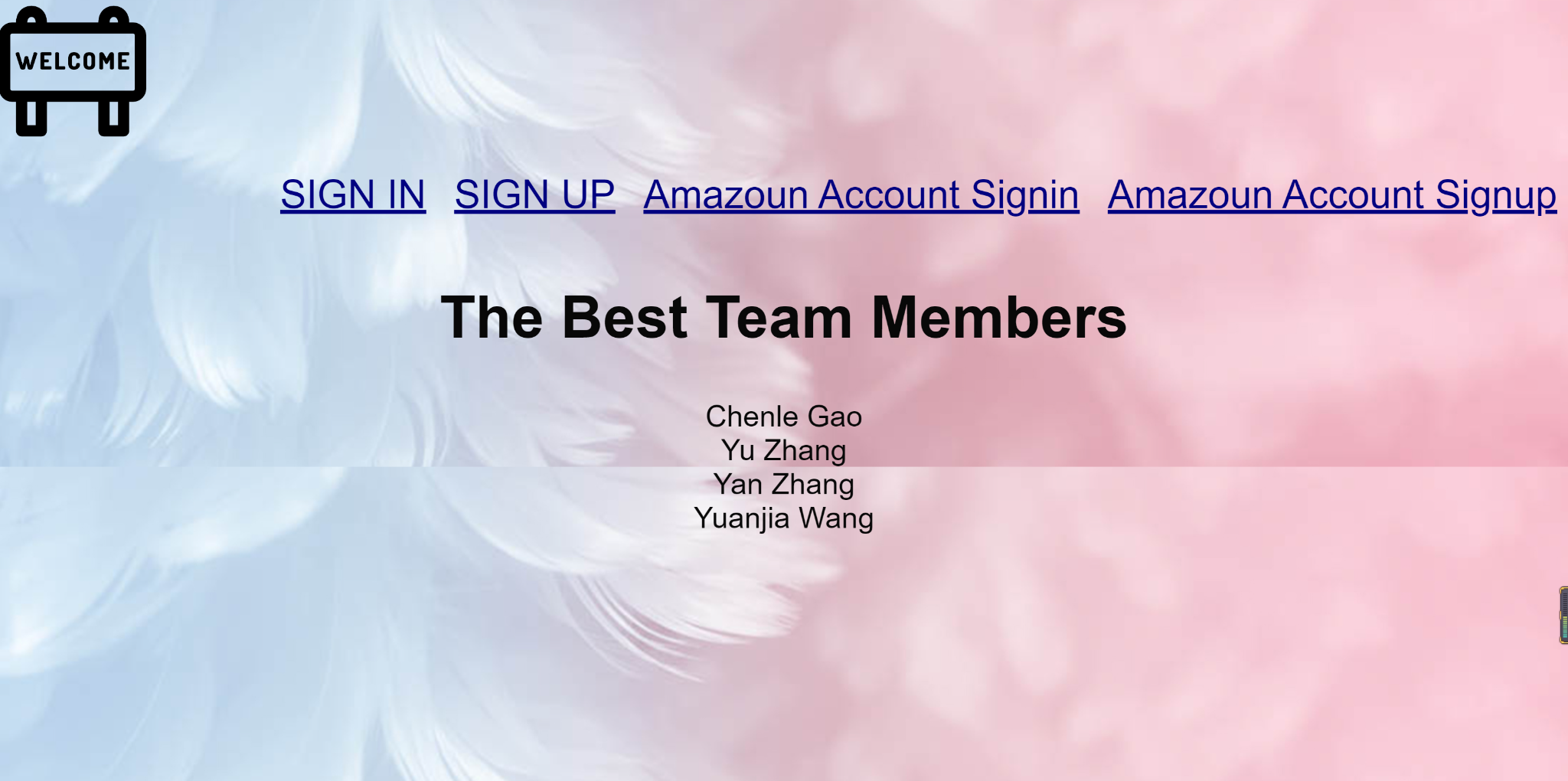
| Task: | Team:  Chenle Gao, Yu Zhang, Yan Zhang, Yuanjia Wang |
| --- | --- |
| Authentication and Authorisation | Yu Zhang 80%, Yuanjia Wang 40% |
| Image Upload | Chenle Gao 80%, Yu Zhang 20% |
| Queries | Yu Zhang 50%, Chenle Gao 50% |
| Delete an image | Chenle Gao 80%, Yan Zhang20% |
| User Interface | Yan Zhang 80%, Yuanjia Wang 20% |
| Report | Yuanjia Wang 60%, Yan Zhang 40% |

1. User Guide

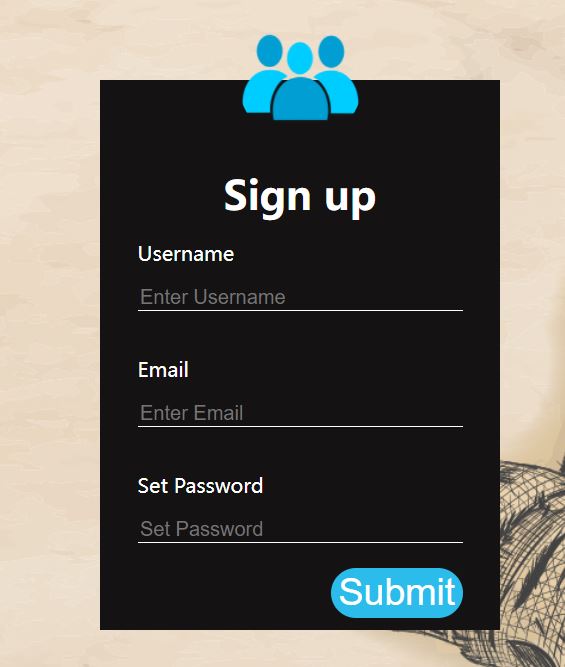
• Report includes a simple user guide for testing your application (you should keep your application up and running for two weeks after the submission deadline).

On the dashboard page, there declines information: welcome message, and team members. There are four links here: 1. Sign in, 2. Sign up, 3. Amazon account Signin and 4, Amazon Account SignUp.

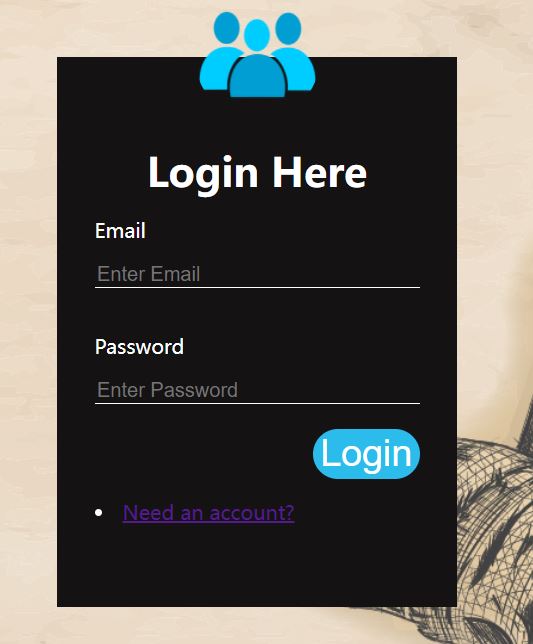
For the Signin page, the user needs to log in to the account with the email address and password. If the user does not have an account yet, the “need an account?” link leads to the Signup page for the user to get registered. If the user wants to log in with the AWS account or register a new account, Amazon Account Sign-in and Amazon Account Sign Up will help.



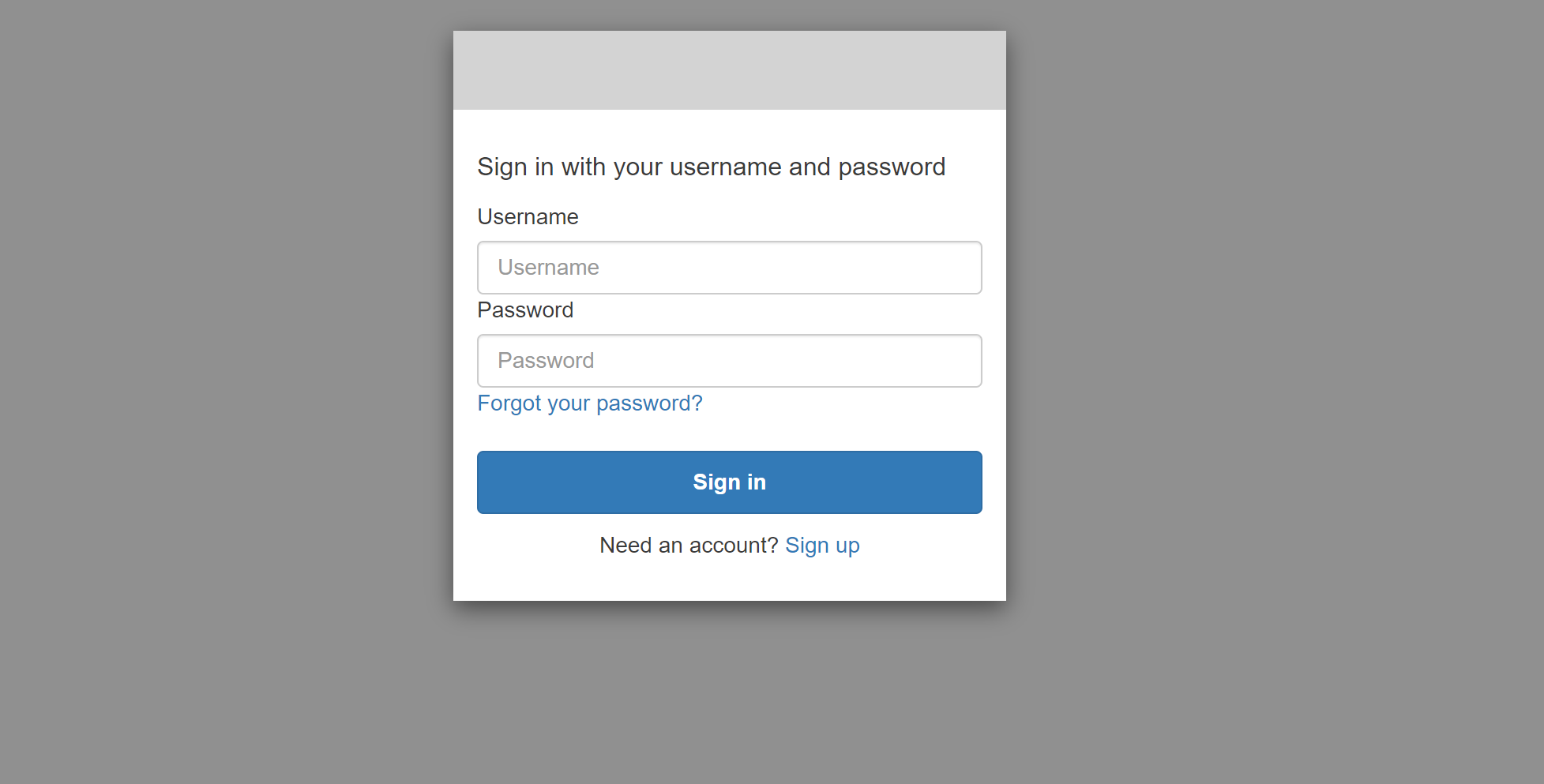
*Figure 4.1 - Dashboard*



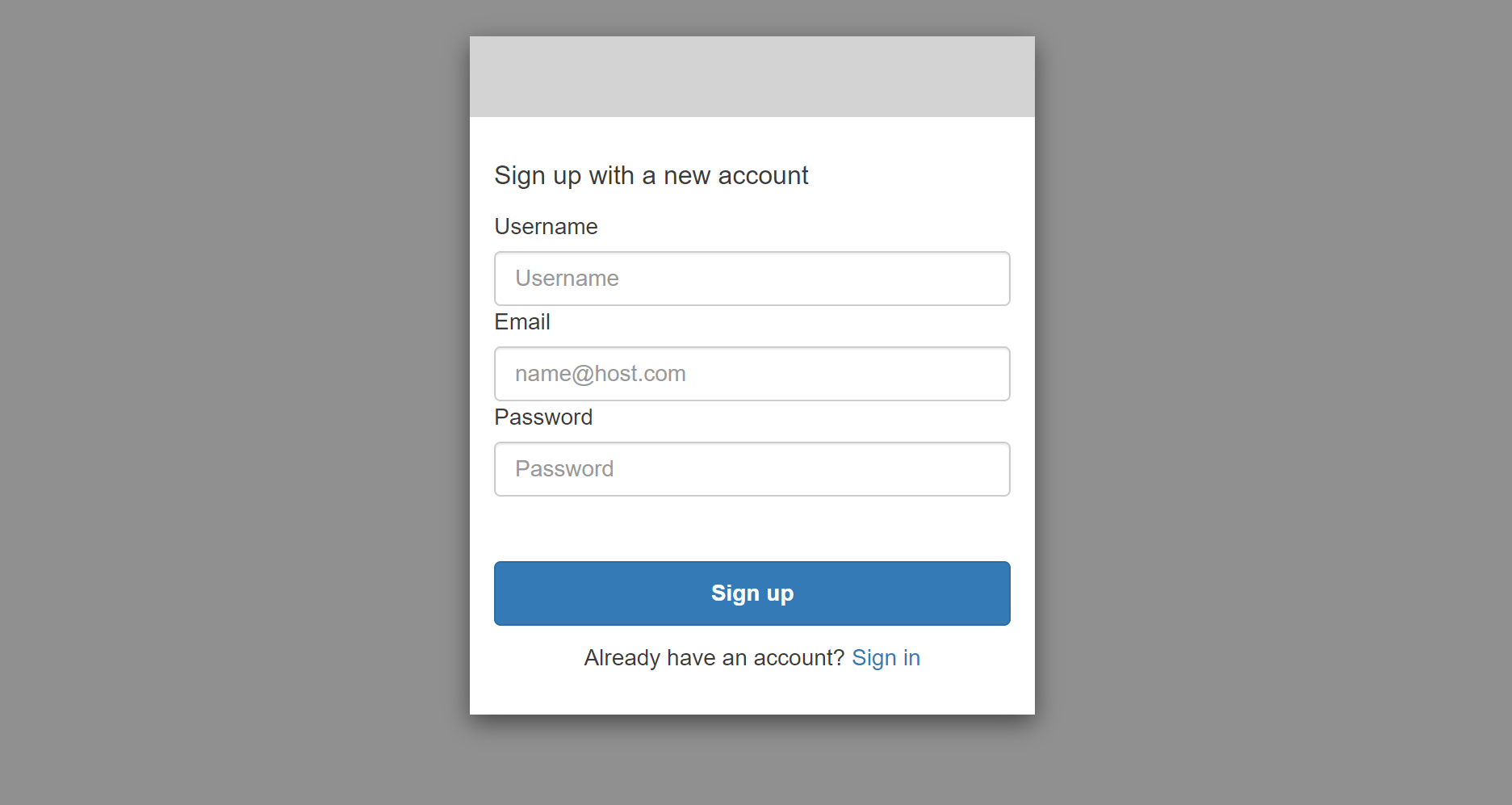
*Figure 4.2 - SignIn*

**

*Figure 4.2 - SignUp*

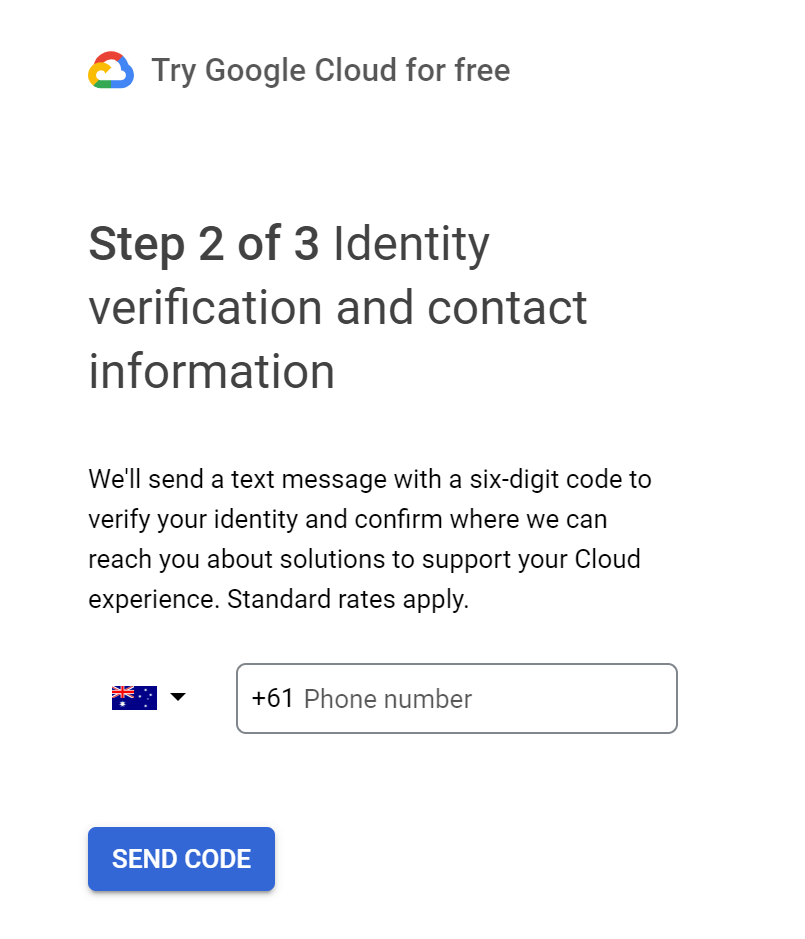
**

*Figure 4.3 - AmazonSignIn*

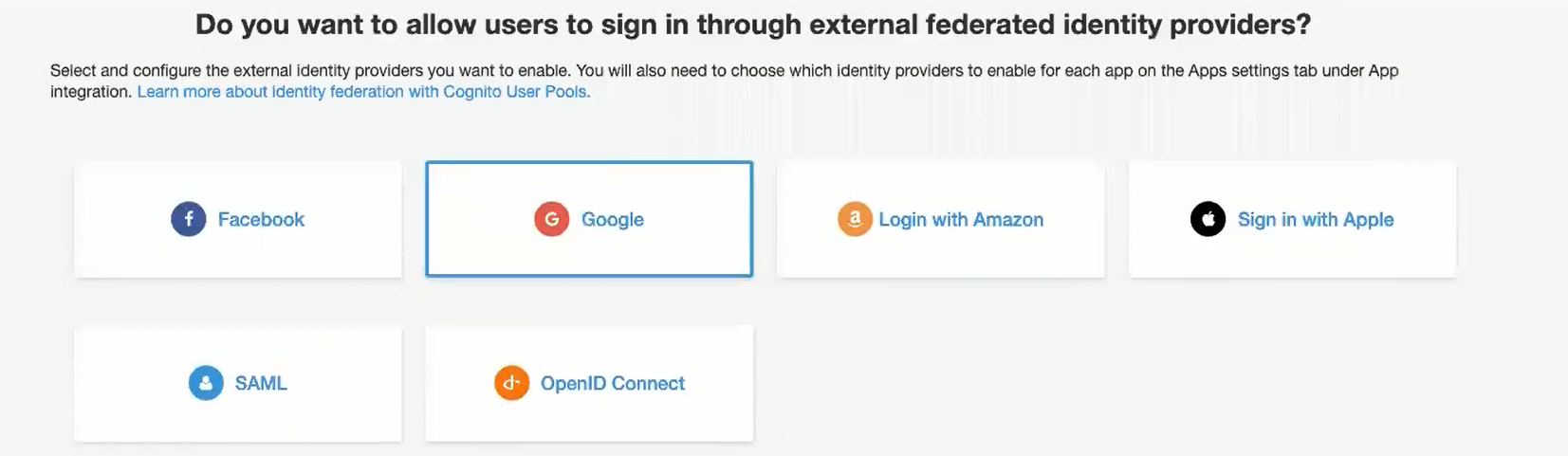
**

*Figure 4.4 - AmazonSignUp*

For the optional feature for login with a Google account, during the implementation of the application, there occurs a problem so, for now, it can only be practised theoretically. When we tried to add a Google app which serves an external identity provider and authenticates users on behalf of the application, there was one problem: we needed to have a Google Cloud Platform Account, which requires us to receive an SMS code to verify. On the grounds that all team members are based not in Australia, it is hard for us to get the code and there seems to be no option for the +86 phone number. Thus the implementation of the additional feature can only be conducted in theory.



*Figure 4.5 - Google Cloud Platform Registration*



*Figure 4.5 - AWS Federation*

For the additional feature, first, we need to register a google cloud account. Then on the AWS platform, go to the User Pools, under the Federation, there is the Identity providers, we can choose google. The pop-up window will suggest we have three attributes: Google app ID, App secret, and Authorize scope.

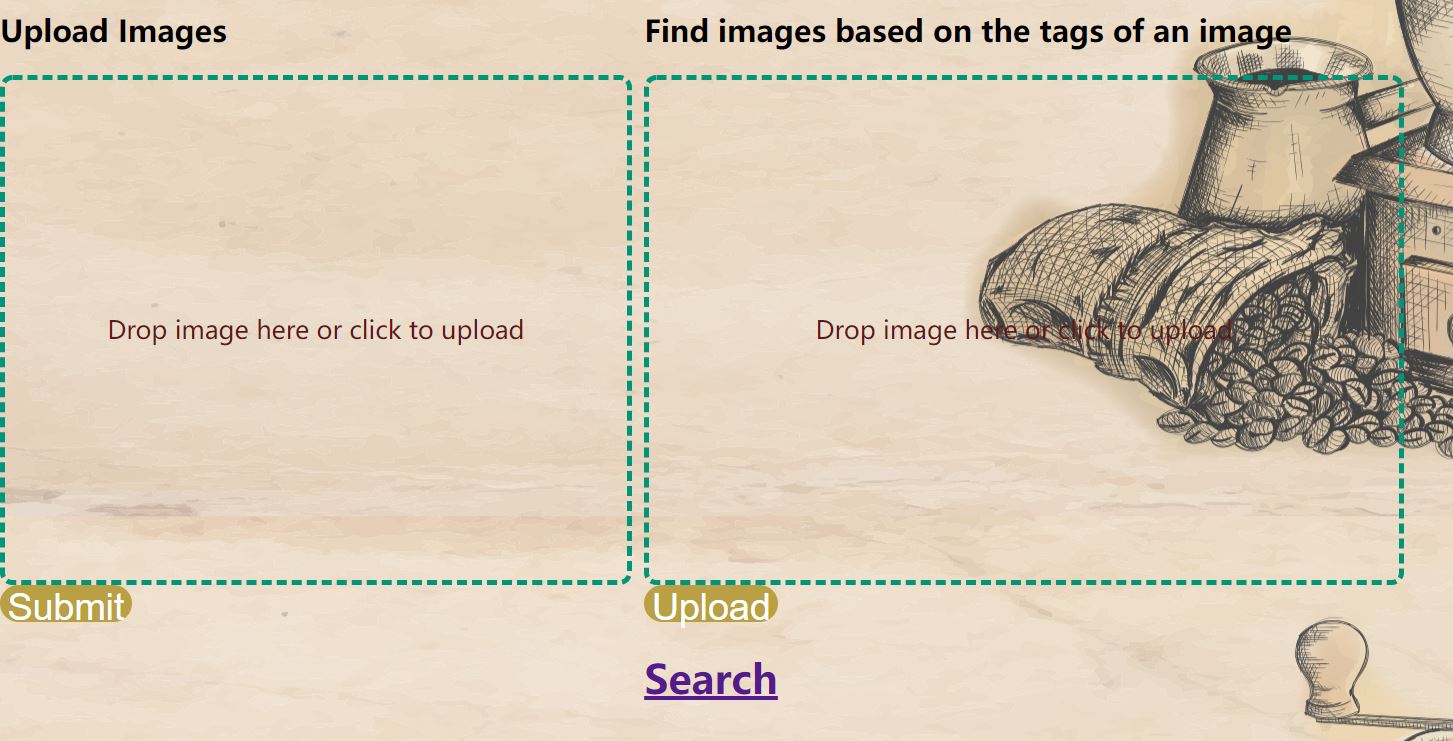
For the Google app id, once we finished with the Google cloud account, we can get the id from the credentials. The client ID should be the Google app ID and the client secret should be the App secret.

Second, go to Google Cloud Platform, under the Credentials, there needs to deploy some settings. Under OAuth 2.0 Client IDs, click the Web client(auto-created by Google Service) link, copy “https://yourDomainPrefix.auth.region.amazoncognito.com” from the AWS guideline and past it to Authorised JavaScript origins URLs. Then copy “[https://yourDomainPrefix.auth.region.amazoncognito.com/oauth2/idpresponse](https://yourdomainprefix.auth.region.amazoncognito.com/oauth2/idpresponse)” from the AWS guideline, and paste it to the Authorised redirect URLs. please note the “yourDomainPrefix” shall change to the specific domain.

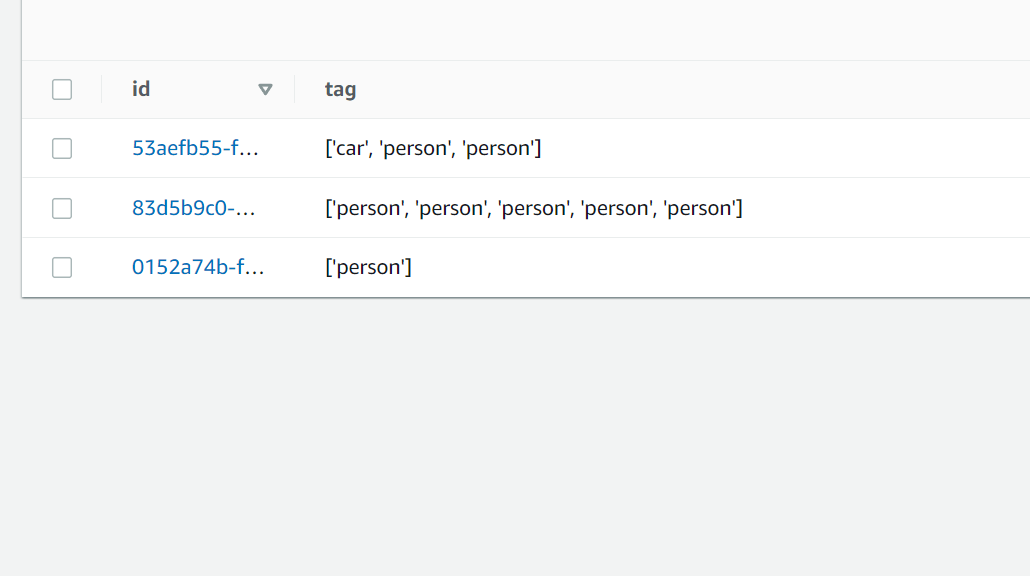
Third, for the Authorize scope, enter “profile email openid”. And Enable Google.

Lastly, go to the AWS page, under App integration, app client setting, enable Google and save changes.

On the main page, there are two separate boxes: one is for uploading images, and another one is for finding images based on the tags of an image. Once the image has been uploaded, click submit button can upload it to the cloud service, the image will list as the form in Figure 4.7. Or click the upload button to detect the tags of such a picture and return the result, the image should first be detected its tags with the lambda function which should find an image based on the tag of an image and then find those pictures and return to the user, the result should be in the form of Figure 4.8.



*Figure 4.6 - MainPage*



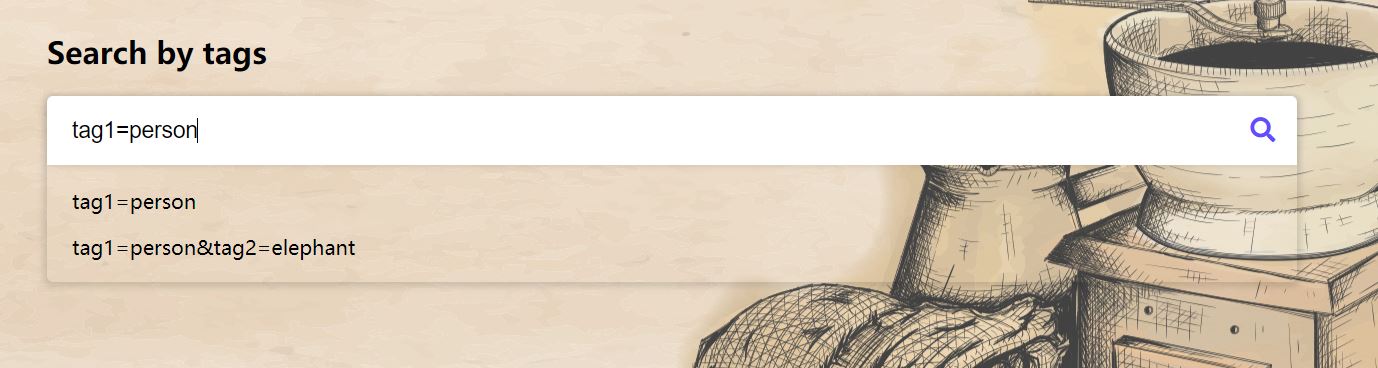
*Figure 4.7 - ImageUpload*

**

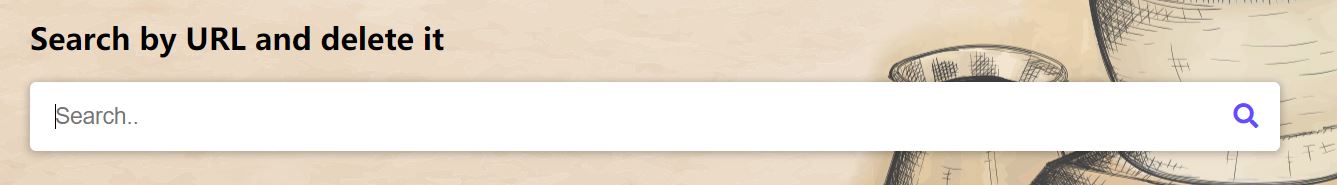
*Figure 4.8 - ImageTagSearch*

If the user wants to delete an image, the user needs to use the “search” link below two input fields. The user needs to appoint a specific image by its URL. The result should be the images with the appointed URL. The functionality will interact with the lambda function in AWS. The user needs to input three indexes as shown in Figure 4.11, then the application will pull out the image from the bucket and delete it.

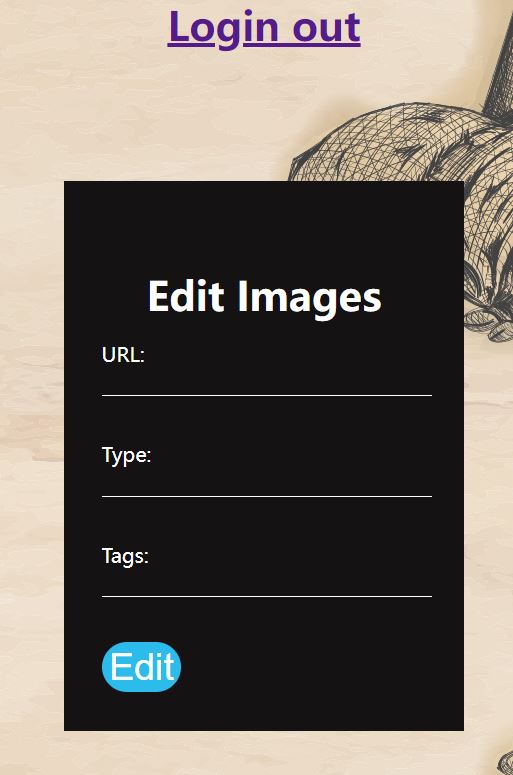
Once the user needs to log out of the account, click the sign-out link, a message will be shown as Figure 4.12 and return to the Dashboard page.

**

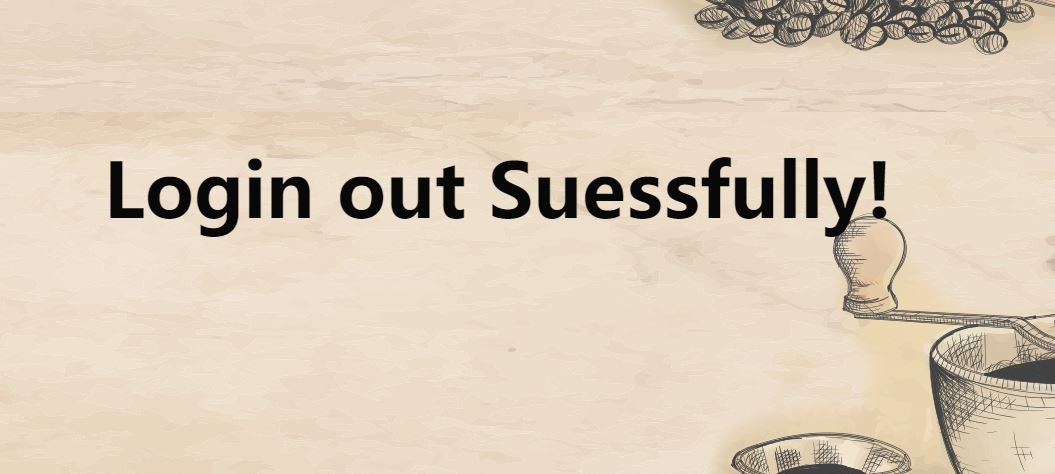
*Figure 4.9 - SearchPage*

**

*Figure 4.10 - SearchPage and delete it*

**

*Figure 4.11 - Edit pages*

**

*Figure 4.12 - Login out*

1. Link

• Your report should include a link to the source code (GitHub or Bitbucket). It is recommended that all students commit their code to a private code repository rather than delegate this to a single team member. This can provide an evidence base if teams run into “issues”. You should share your private repository with the whole teaching team. Please do not use a public repository to avoid any plagiarism.

—LINK—

1. Reference List:

How to Add Google Social Sign On To Amazon Cognito | Step by Step Tutorial(2022). Retrieved 23 May 2022, from https://www.youtube.com/watch?v=r1P\_glQGvfo

Set up Google as a social identity provider in an Amazon Cognito user pool. (2022). Retrieved 24 May 2022, from https://aws.amazon.com/tw/premiumsupport/knowledge-center/cognito-google-social-identity-provider/