ECE1779: Introduction to Cloud Computing

Assignment 3: Project

Team Members:

Name	UTORID	Student Number	Email
Zhaodong Yan	yanzhaod	1001263385	zhaodong.yan@mail.utoronto.ca
Zijian Wang	wangzij7	1001644964	jarrett.wang@mail.utoronto.ca
Beibei Zhang	zhan4554	1003920876	benjamin.zhang@mail.utoronto.ca

Functions:

The main objective of the web page is to allow the user to do some text analysis with the help of AWS apis. Specifically, a user could create its account, login its account, and do the following four functions. Grammar check, translation, sentiment analysis and text extraction. This webpage has been created using python, flask, AWS Lambda, DynamoDB, S3. We use Zappa to deploy the app on AWS lambda. We fulfill the following functions:

1. **User registration**: The user is allowed to register with a proper username and password pair. The username cannot be null and must be within 100 characters. The password cannot be null and must include capital letters, small letters, and numbers. It must be longer than 8 characters and less than 255 characters. A random salt is generated and stored in the DynamoDB database, and the password that user inputs will append the salt and be hashed. The username and password pair is stored in the DynamoDB database. If user violet these rules, we will show a notification on the webpage. If user used an existed username, we will also show a notification on the webpage.

2. **User login**: The user can login using the registered username and password pair. If the user input a wrong password, we will show a notification on the webpage.

3. **Logout:** By default, the session will expire in 24 hours. After 24 hours, the user need to login again. We also provide a logout function. Once logout the session will be cleared and the user need to login again.

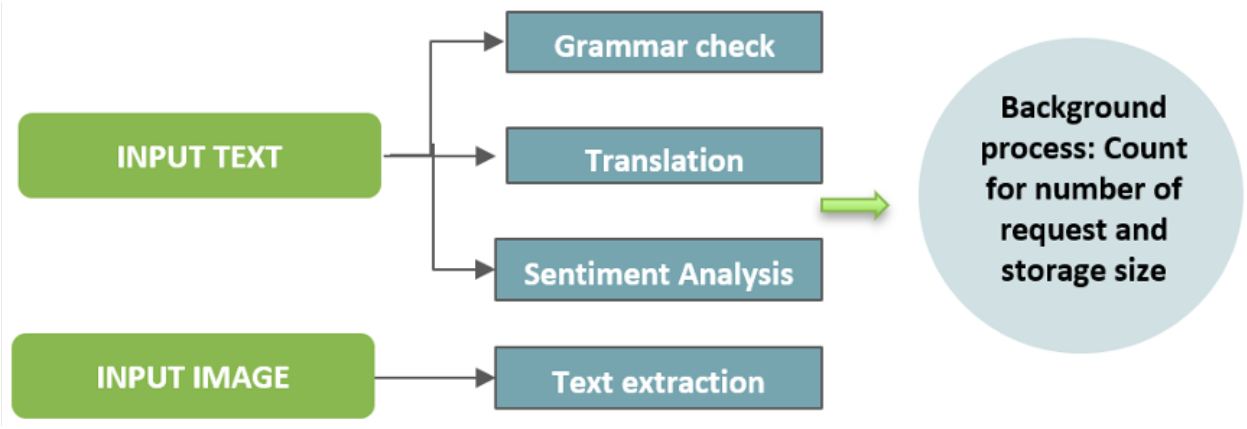
4. **Grammar Check**: Grammar check function determines the misspelled words in the input text and make corrections while label those changes in the text. This function is fulfilled by AWS api: GingerIt()

5. **Translation:** This function can translate the input text to a variety of languages selected by the user. This is fulfilled by the AWS api: translate_text()

6. **Sentiment Analysis:** This function is able to detect how positive or negative the sentiment showed in the text is. This function is fulffiled by AWS api: comprehend()

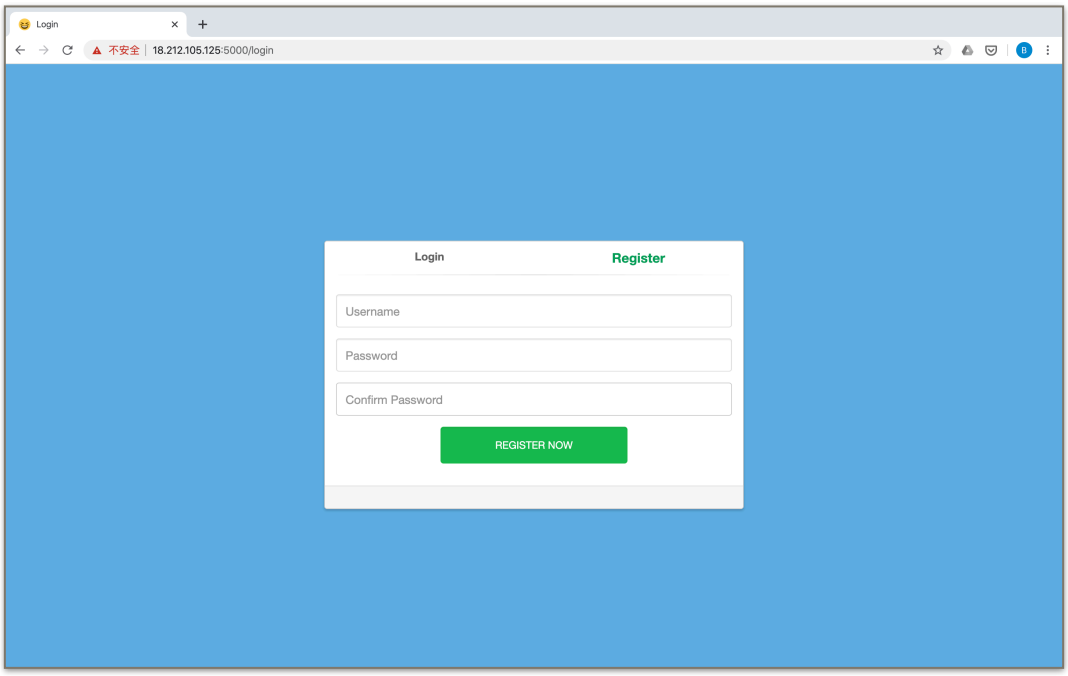
7. **Text Extraction:** This function allows the user to upload an image with printed text on it and convert it to pure text. This function is fulfilled by AWS api: textract(). The inputs texts and images from user will be uploaded to our S3 bucket.
8. **Background Process:** Our background process, which is implemented by AWS Lambda, will simultaneously update the total number of requests from that function and the size of all user-uploaded files for the function in our DynamoDB.

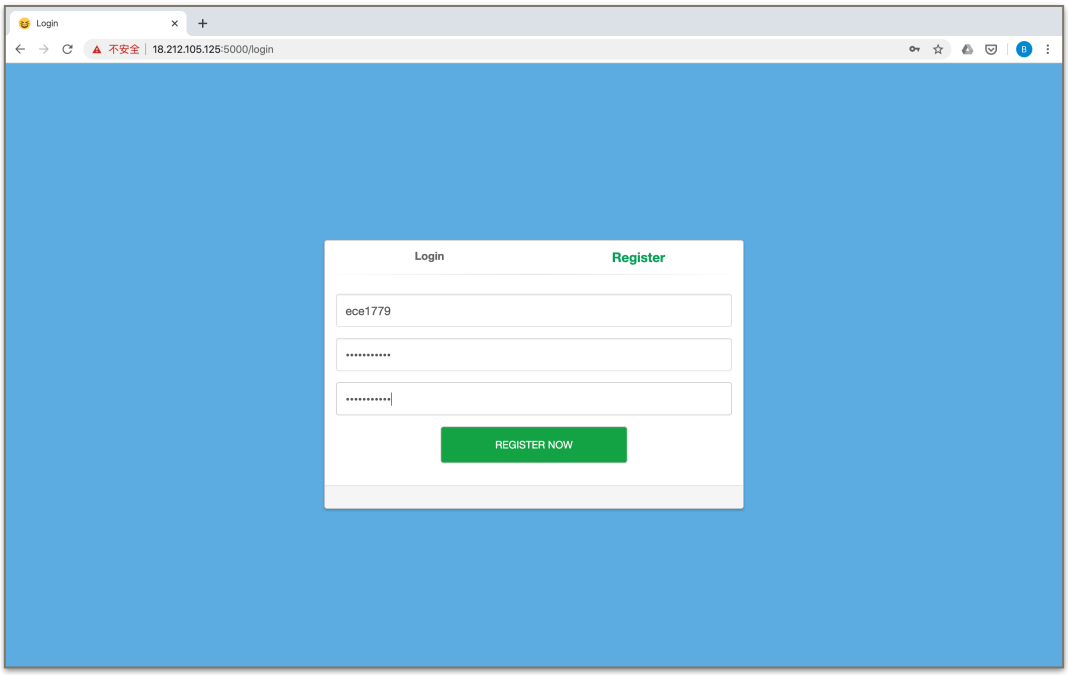
The overall functionalities can be categorized as follows:



User Guide:

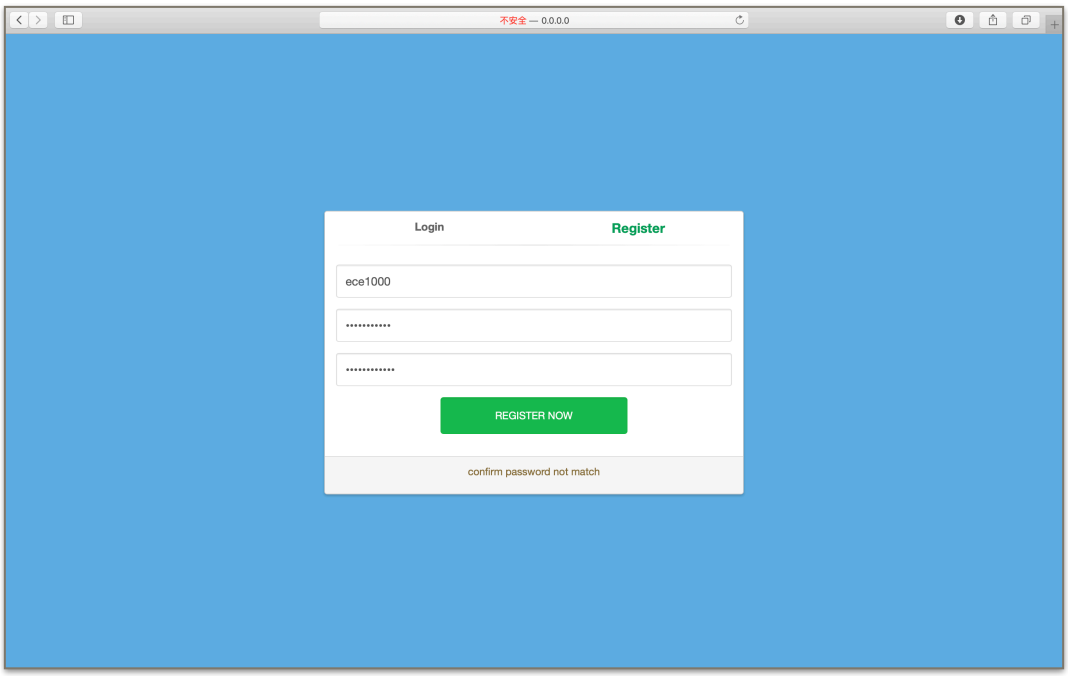
1. Access the application through the following link: https://majhvfcpm9.execute-api.us-east-1.amazonaws.com/dev/login_v2 You are able to register with your username and password pair. Or you can use the demo account: username: ece1779 password: Ece1779pass





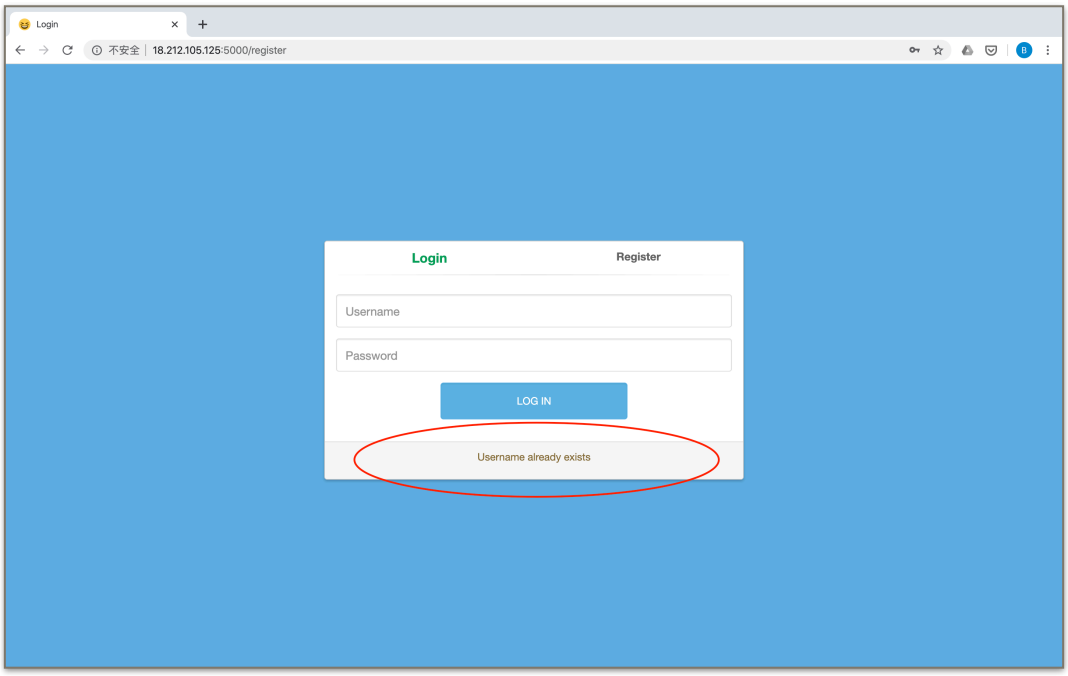
Register and Login page

2. Besides, make sure that your confirmed password is the same as your registered password, or it will show:



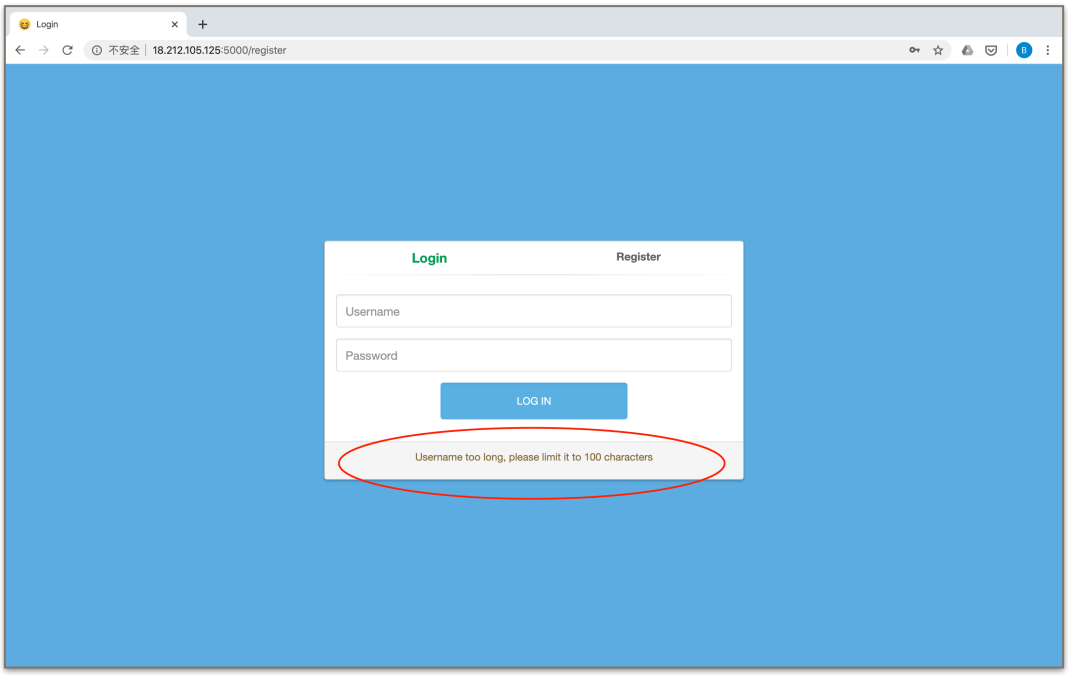
Password Confirmation Not Match Error

3. If your username exists, it will show:



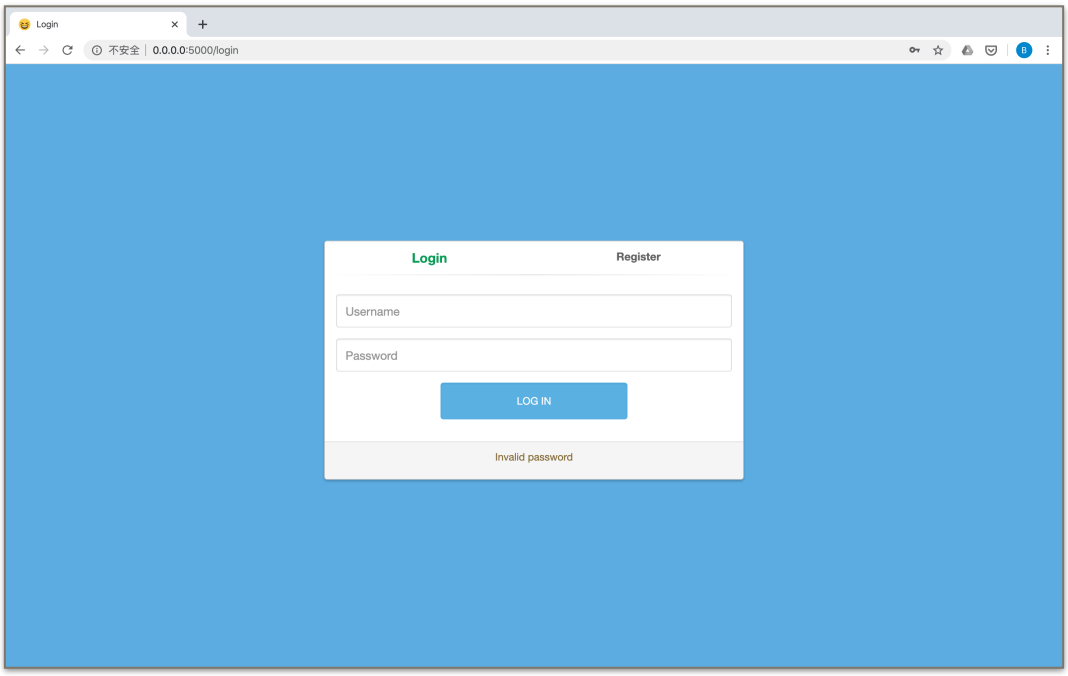
Username Exists Error

4. If your username is too long (should be within 100 characters), it will show:



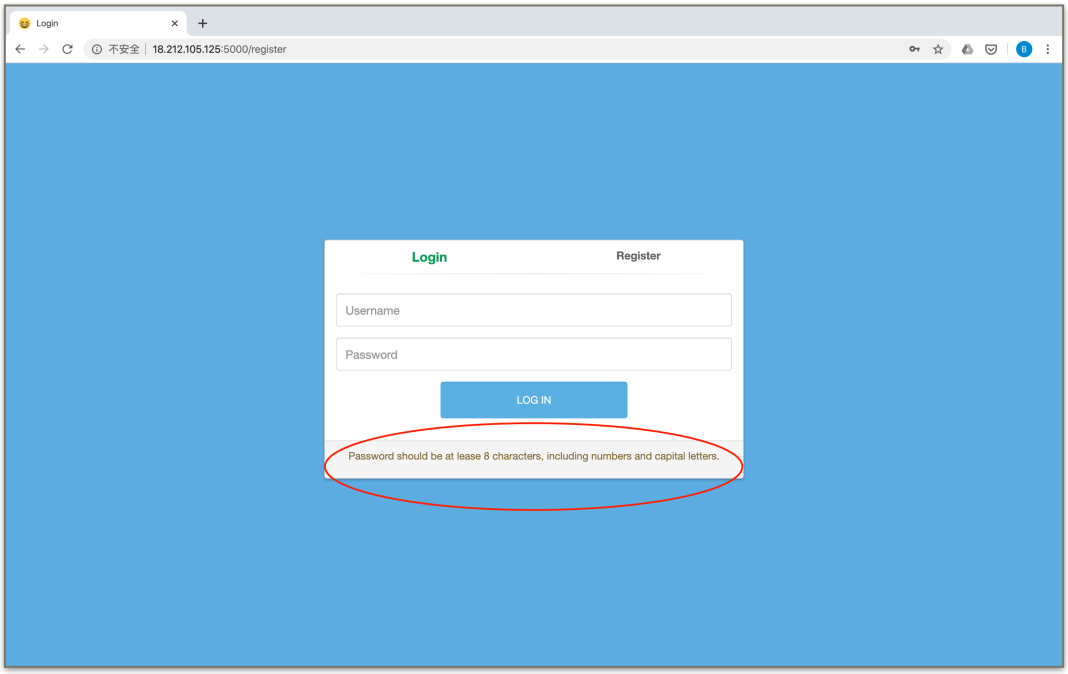
Username Too Long Error

5. If your password is wrong, it will show:



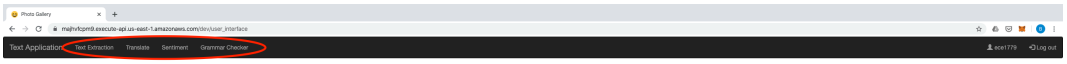
Password Wrong Error

6. If your password is longer than 255 characters or it does not include capital letters, small letters, and numbers, it will show:



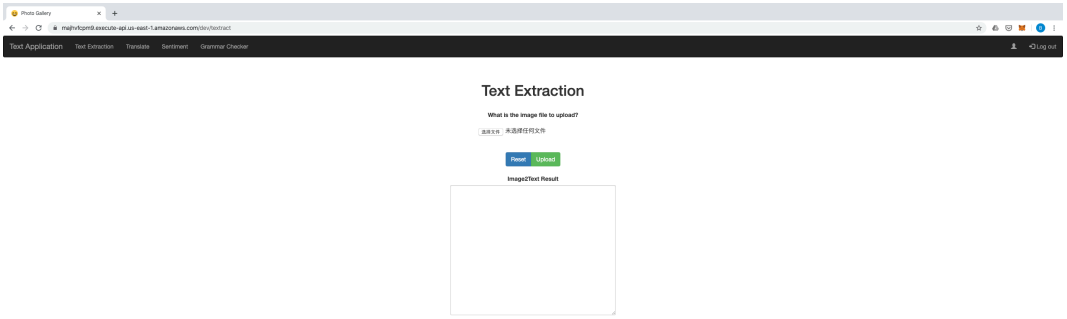
Password Error

7. If you successfully login the application, it will show the main page. All the functions are listed on the top menu:



Function Page

8.Click the Text Extraction and click the select file button to select the image that you want to upload.



Text Extraction from Image

9. You can choose an image and click upload button to upload it. After the image is uploaded, it will be shown on the page, and stored in S3. The text in the image is extracted and shown in the text area. We also designed an notification bar to notify the user that the image upload status.

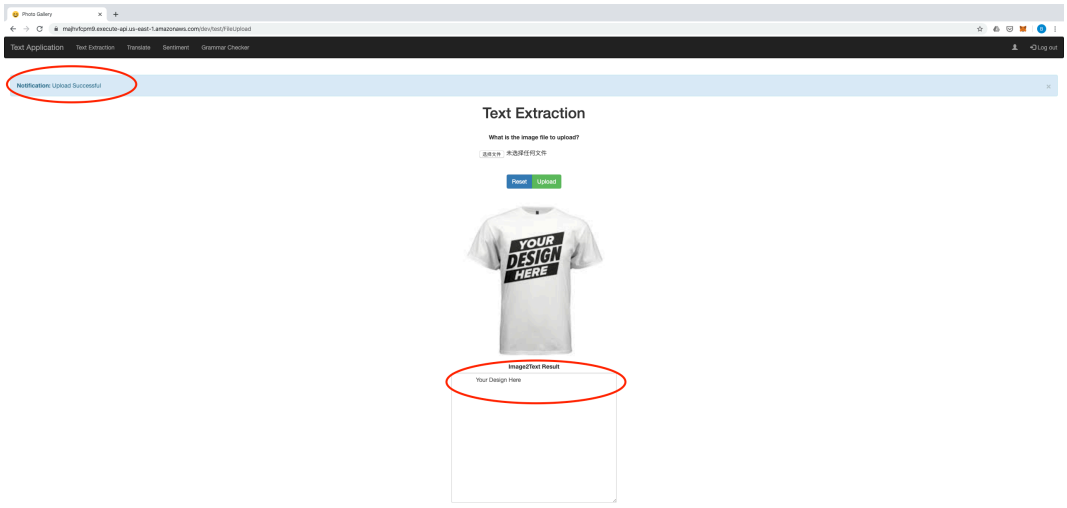
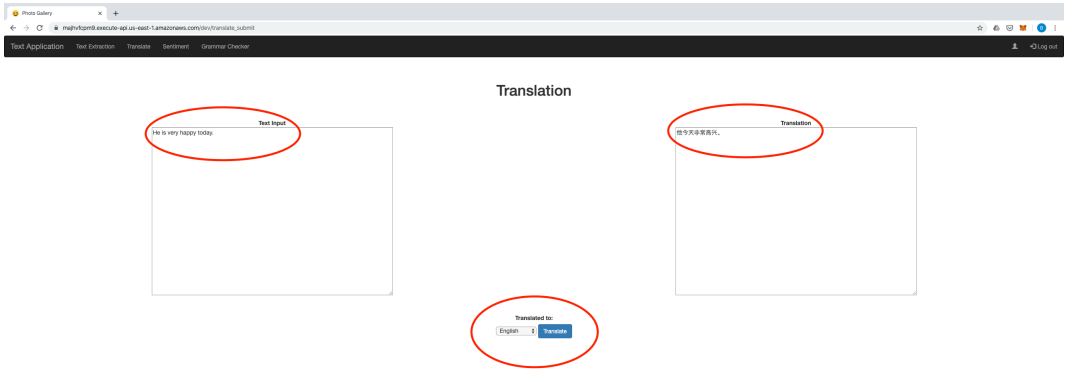


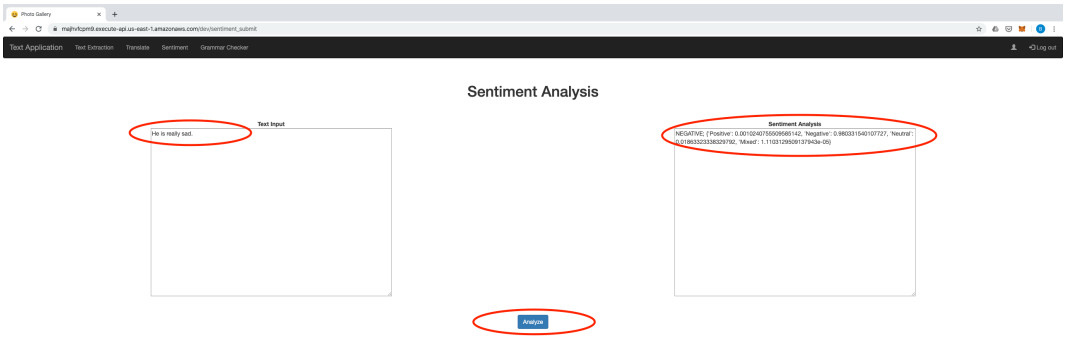
Image Uploaded

10. Click the Translate tab to use the translation function. To be more specific, input the your English text on the left text area, select the language you want to translate to, and click Translate Button. You translated text will be shown on the right text area.



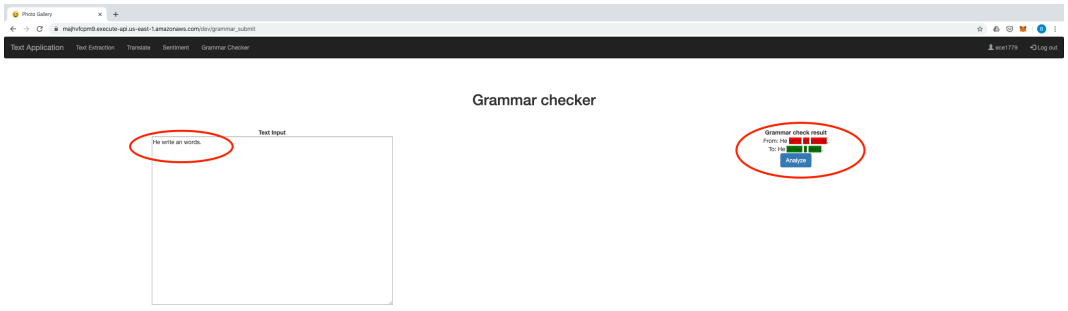
Multi-language Translation

11. For sentiment analysis, click the Sentiment tab on the top menu bar. Input your sentence on the left text area and click analyze button. The sentiment of the sentence will be shown on the right. It indicate the percentage of positive negative or neutral of this sentence.



Sentiment Analysis

12. Grammar checking function can be obtained by clicking the Grammar Checker Tab on the top menu bar. Input a sentence and click analyze button. The incorrect parts will be highlighted in red and its correction will be highlighted in green accordingly:



Grammar Checker

Design Topology:

The design to the application follows the modular thinking with the following files taking care of the related functionalities:

- __init__.py: Instantiates the application object of Flask and initializes the application of Flask.

- Login.py: Contains the register page and login page.

persist_session()	Let session persists for 24 hours
login_v2_submit()	Deal with login functionalities. If the username exists, if the password is correct
usercreate_save()	Deal with register functionalities. If the username and password align to the specification. Generate random salt and hash the password
logout()	Logout function

- fileupload.py: This file contains the upload function and image processing function.

file_upload()	Upload the image for text extraction

- comprehend.py: This file contains the text analysis functions.

getSentiment(text)	Get sentiment result from aws api
detectLanguage(text)	Return the language type code from aws api
translate(text, targetLang='en')	Translate the English text to target text
comprehend()	Return sentiment analysis result.
grammar_submit()	Do grammar checking and return the corrected result.

Database Design:

Our database contains three tables: '1779user', 'a3Analytics' and 'a3RequestLog. We store user credentials in the '1779user' table:

- **id**: primary key, it is automatically generated, and it is unique.
- **username**: exact username. The length of the username cannot be longer than 100 characters.
- **password**: stores hashed password. Password should contain numbers, small letters, and capital letters. The length of a password should be at least 8 characters.
- **salt**: salt is used to enhance the security of a password. Each user has their own unique salt. The salt has a length of 16 and it is generated by a random generator.

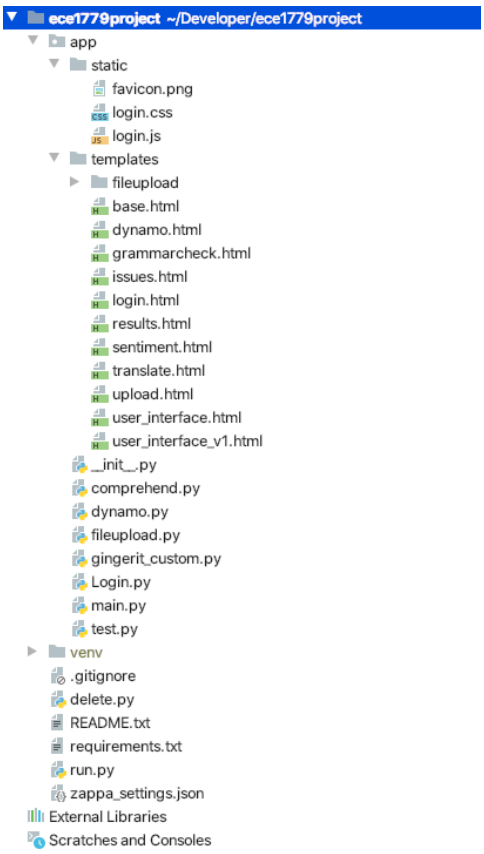
We store image information in the ‘images’ table:

- **requestID**: he corresponding username for the requests and files. There is an additional key called ‘total’ which stores the sum of requests and file sizes for all users.
- **grammar_req**: the number of requests from grammar check function.
- **grammar_size**: the sizes (in bytes) of the txt files of the input texts uploaded to the grammar check function.
- **sentiment_req**: the number of requests from sentiment analysis function
- **sentiment_size**: the sizes (in bytes) of the txt files of the input texts uploaded to the sentiment analysis function
- **textract_req**: the number of requests from text extract function
- **textract_size**: the sizes (in bytes) of the input images uploaded to the text extract function
- **translation_req**: the number of requests from translation function
- **translation_size**: he sizes (in bytes) of the txt files of the input texts uploaded to the translation function

We store logs in the table ‘a3RequestLog’:

- **requestID**: A universal unique id generated for each requests using uuid()
- **author**: The corresponding user of the log
- **event_time**: The creation time for the log
- **file**: the path to the uploaded file in our s3 bucket ‘ece1779project’
- **requestType**: the function that is used at the request
- **size**: The size of the uploaded document

App file structure:



Cost Model:

The cost model for AWS costs. Also, predict these costs after six months for 10, 0000 and 1000,000 users using your model. Discuss your assumptions about the user behaviour, such as requests per day, etc.

From the AWS website, rates of functions utilized in this project are obtained. Rates are different after free-tier expires and the changes in rates are included in the cost model as well. We can reasonably assume that our text analysis app is used in the work days and each function is requested once a week per user. Based on the above assumptions, ~15 requests are made per user per month. Therefore, weâ€™ll calculate the costs that are generated in AWS Lambda, AWS Translate, AWS Comprehend and AWS S3 respectively according to the above assumptions.

Calculation Example for 100000 users per month:
Lambda Calculations-Include Free Tier
Assumptions:
Number of requests: 1500000
Duration of each request (in ms) : 2000msÂ
Amount of memory allocated: 2049 MB
Induced costs: 93.43 USD

Lambda Calculations-Exclude Free Tier
Assumptions:
Number of requests: 8000000
Duration of each request (in ms) : 2000msÂ
Amount of memory allocated: 2049 MB
Induced costs: 100.03 USD

Translate Calculations-Include Free Tier
Assumptions:
Usage: 4 requests per user per month
# of Characters: 300 per request
Induced costs: 1770.00 USD

Translate Calculations-Exclude Free Tier
Assumptions:
Usage: 4 requests per user per month
# of Characters: 300 per request
Induced costs: 1800.00 USD

Comprehend-Include Free Tier:
Assumptions:
Usage: 4 requests per user per month
# of Characters: 300 per request
Induced costs: 0.00 USD

Comprehend-Exclude Free Tier:
Assumptions:

Usage: 4 requests per user per month
# of Characters: 300 per request

Induced costs: 120.00 USD

Textract-Include Free Tier:
Assumptions:
Usage: 4 requests per user per month
Induced costs: 598.5 USD

Textract-Exclude Free Tier:
Assumptions:
Usage: 4 requests per user per month
Induced costs: 600.00 USD

S3 and DynamoDB- Include Free Tier:
Assumptions:
Only Textract files are stored and one write-in per request
Induced costs: 9.09 USD

S3 and DynamoDB- Exclude Free Tier:
Assumptions:
Only Textract files are stored and one write-in per request
Induced costs: 9.2 USD

In conclusion, the total cost for six month and 100000 users with appropriate assumptions with a free tier account is about 14826.12 USD and 15777 USD without a free tier account.

Calculation Example for 1000000 users per month:
Lambda Calculations-Include Free Tier
Number of requests: 15,000,000
Duration of each request (in ms) : 2000msÂ
Amount of memory allocated: 2049 MB
Induced costs: 996.14 USD
Lambda Calculations-Exclude Free Tier
Number of requests: 15,000,000
Duration of each request (in ms) : 2000msÂ
Amount of memory allocated: 2049 MB
Induced costs: 1003.00 USD
Translate Calculations-Include Free Tier
Assumptions:
Usage: 4 requests per user per month
# of Characters: 300 per request
Induced costs: 17970.00 USD
Translate Calculations-Include Free Tier
Assumptions:
Usage: 4 requests per user per month
# of Characters: 300 per request
Induced costs: 18000.00 USD

Comprehend-Include Free Tier:
Assumptions:
Usage: 4 requests per user per month
# of Characters: 300 per request
Induced costs: 350.00 USD

Comprehend-Exclude Free Tier:
Assumptions:
Usage: 4 requests per user per month
# of Characters: 300 per request
Induced costs: 600.00 USD

Textract-Include Free Tier:
Assumptions:
Usage: 4 requests per user per month
Induced costs: 3299.4 USD

Textract-Exclude Free Tier:
Assumptions:
Usage: 4 requests per user per month
Induced costs:3300.00 USD

S3 and DynamoDB- Include Free Tier:
Assumptions:
Only Textract files are stored and one write-in per request
Induced costs: 94.14 USD

S3 and DynamoDB- Exclude Free Tier:
Assumptions:
Only Textract files are stored and one write-in per request
Induced costs: 98.00 USD

In conclusion, the total cost for six month and 1 million users with appropriate assumptions with a free tier account is about 136258.08 USD and 138066 USD without a free tier account. From observations, AWS Translate is a very expensive service to use. Though it has a flat rate, the add-up amount for large user groups is large. Therefore, for web apps that target large user groups, including aws translate may not be a good choice in the budget perspective.