SENIOR PROJECT
ANALYSIS REPORT

Dinçer İnce
Halil Mert Güler
Süleyman Samed Çalışkan

Date: 05.12.2022

1. Introduction

EasySearch is a document storage, management and analysis solution. Using this solution users can store documents, see statistics about the documents, perform detailed search on the documents, categorize the documents into a folder structure and categorize further using machine learning algorithms.

The appeal of the EasySearch comes from its simplicity in its usage. Searching through millions of words requires a lot of setup and configuration efficient architecture and so forth. Even then the solution is sometimes not satisfactory, as proper implementation requires someone knowledgeable about the specifics of natural language processing. EasySearch provides that functionality with virtually no work and time. By uploading the documents to the solution API does all the work for the user and user can immediately start searching through their documents. It also doubles as a document storage solution as well.

EasySearch is a solution consisting of two parts, a web application and an API. The API is the main part of the application, and all the logic of the application will be present in. The web application is a way to both manage the API settings, configurations etc., and a way to use the solution completely free of a third party application.

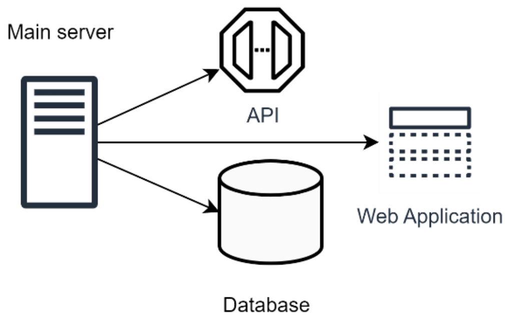
Using the web application users, can utilize the solution as an easy way to store and manage documents. Users can, create API keys, create dictionaries and configure their dictionaries, fetch edit or upload documents to their dictionaries, perform detailed search on the dictionaries, perform machine learning categorization on the dictionaries to categorize the documents. The web application provides all the capabilities the solution offers, however the main use case of the application is to be used with its API.

Using the API, users can, delete edit or upload documents to their dictionaries, and perform detailed search on the documents that covers all of the content of the document like a search engine. Users can utilise this API in their desktop or web application to effectively and simply solve every problem related to document storage and detailed search.

2. Proposed System

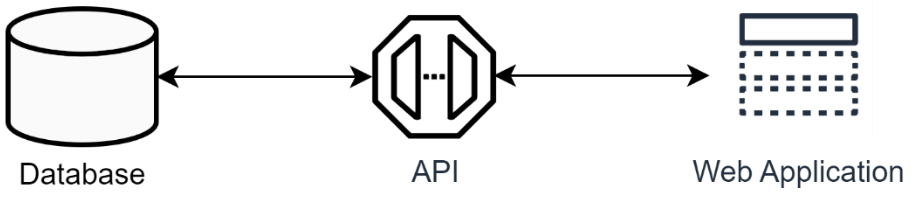
2.1 Overview

As the system is a storage heavy solution system can be very modular except for the web application. There will be different configurations depending on the usage. The initial configuration of the system will be as such;

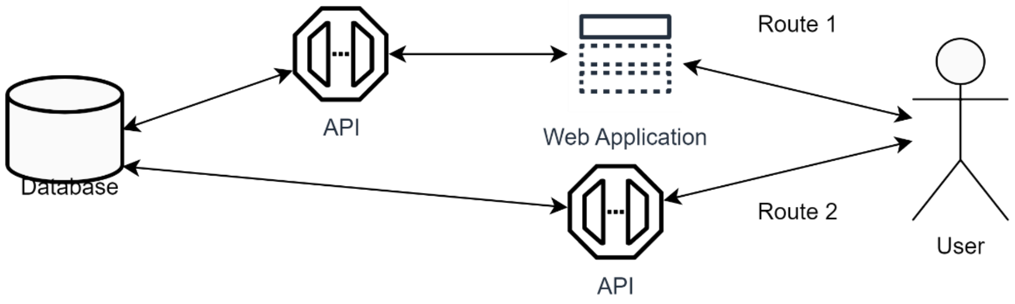


Main server will host the API, the web application and the database. As everything is in one location the delay between the components will be minimized, and would provide high performance at low user count.

The module interaction will be as such;



All of the interactions to the database will be conducted through the API. The main logic and the core of the solution is our API (or back end), the users can utilize the solution through the API itself or through web application where it visualizes the API and provides some configurations that can only be done through the web application.



User can interact with the solution using these two routes defined in the figure above. If the user wants to use the solution as a standalone document storage option, user should use route 1, if the user would like to use the solution as a helper for their application user would use the route 2.

2.2 Functional Requirements

-Users shall be able to sign up through the website of the API in order to access the API.

- Users shall be able to sign in through the website in order to access and interact with the settings of their configuration, and their data.

-Users shall be able to create new API keys and can select specific permissions to be given to the key.

- Users shall be able to add new dictionaries in order to perform search only on the documents registered for that dictionary.

- Users shall be able to refresh the dictionaries, in order to update the dictionary to be equal to the documents contained inside it.

- The system shall be able to refresh the dictionaries at periods designated by the user.
- Users shall be able to delete dictionaries.

- Users shall be able to add documents to their dictionaries, in order to perform search and store their documents.

- Users shall be able to get their individual documents.

- Users shall be able to get their documents multiple at a time with a specified number of documents and the page number.

- Users shall be able to sort their documents in specified ways.

- Users shall be able to edit their documents.

-Users shall be able to delete their documents.

- Users shall be able to perform search on their documents using the algorithms provided by the API as they prefer.

- The search function shall take into account every word of each document in the dictionary to thoroughly evaluate the result to be sent back.

- Users shall be able to change their preferred searching algorithm through the website.

- Users shall be ablet to change the number of search results to be sent back in order to limit the results received.

- Users shall be able to view various analytics gathered from the usage of the API in order to analyze their document storage and search solution.

2.3 Non-functional Requirements

1. Performance Requirements

Because of this project is an API system, it will require a stable and strong network connection to use it in best performance.

2. Safety Requirements

Users must use reliable network connections while they use this API for security reasons. According to Privacy Policy of this application, users are responsible for their data sharing and own device software, connections etc.

3. Security Requirements

This project will hold the information of registered users and because of that, database information could be hacked, user information could steal. To avoid circumstances like this, thrusted hosting services should use for this project. Also, this project includes an authorizing system for tracking users and event on system for security reasons.

4. Software Quality Attributes

This project’s dashboard is adaptable for other platforms like mobile. Because database and API system can be remained same, but frontend must be redeveloped in case of adapt. Application is available for all users who has network connection and browser on their device. Project can maintainable as long it has a domain and hosting service. This application is reliable because it uses many security systems and protocols to keep user data safe. Project is robust because it developed, designed by lates long time supported frameworks, database, software and platforms.

2.4.1 Scenarios

1. New user registration
   - The user opens the web site of application
   - Information page and login-register buttons are displayed.
   - The user chooses the register button
   - Registration screen appears with the input fields for email, username and password.
   - The user fills the input fields with their information and clicks register button.
   - System sends register confirmation email to user.
   - The user clicks the link in email to confirm his/her email and registration and link redirect to continue page.
   - The user chooses to continue
   - Dashboard page appears with options view dictionaries, manage account, get product key, search box and general statistics.
   - The user chooses to view dictionaries.
   - Dictionary view page will appear with the options create new dictionary, list of created dictionaries, dictionary detail, delete dictionary and configure dictionary.

2. View dictionaries
    - Dictionary view page will appear with the options create new dictionary, list of created dictionaries, dictionary detail, delete dictionary and configure dictionary.
    - The user chooses to create new dictionary.

3. Generating new dictionary
    - Create new dictionary page will appear with the options dictionary name and a create button.
    - The user fills the input field with desired dictionary name and chooses the create button.
    - System will redirect to dictionary configuration page.

4. Configure dictionary
    - Configure dictionary page will appear with the options select dictionary algorithm and delete the dictionary.
    - The user chooses to select dictionary algorithm
    - Available dictionary algorithms will be listed o select
    - The user selects one of the available algorithms.

5. Remove dictionary
    - Warning message will appear with the message "The dictionary will be deleted permanently!", input field for dictionary name to confirm dictionary name, and a delete button which will be available if the dictionary name filled correctly.
    - The user fills the input field with correct dictionary name and chooses to delete.
    - Message will appear with the text "dictionary has deleted successfully!".
    - System redirects to dictionary list page.

6. Dictionary detail
    - Dictionary detail page will appear according to the selected dictionary with the fields dictionary name, selected algorithm, dictionary's document count, and option to create new document.

7. Generating new document
    - Create new document page will appear with the fields document name, file import option, textbox for manual import, dictionary category, and send button.
    - The user fills the fields according to the document and chooses to send button
    - Massage will appear with the text "document created and added to the dictionary ["dictionary name"]!".
    - System will redirect to document list

8. View document list
    - Document list will appear with the fields list of all documents in selected dictionary, delete document and edit document.
    - The user chooses to delete or edit document.

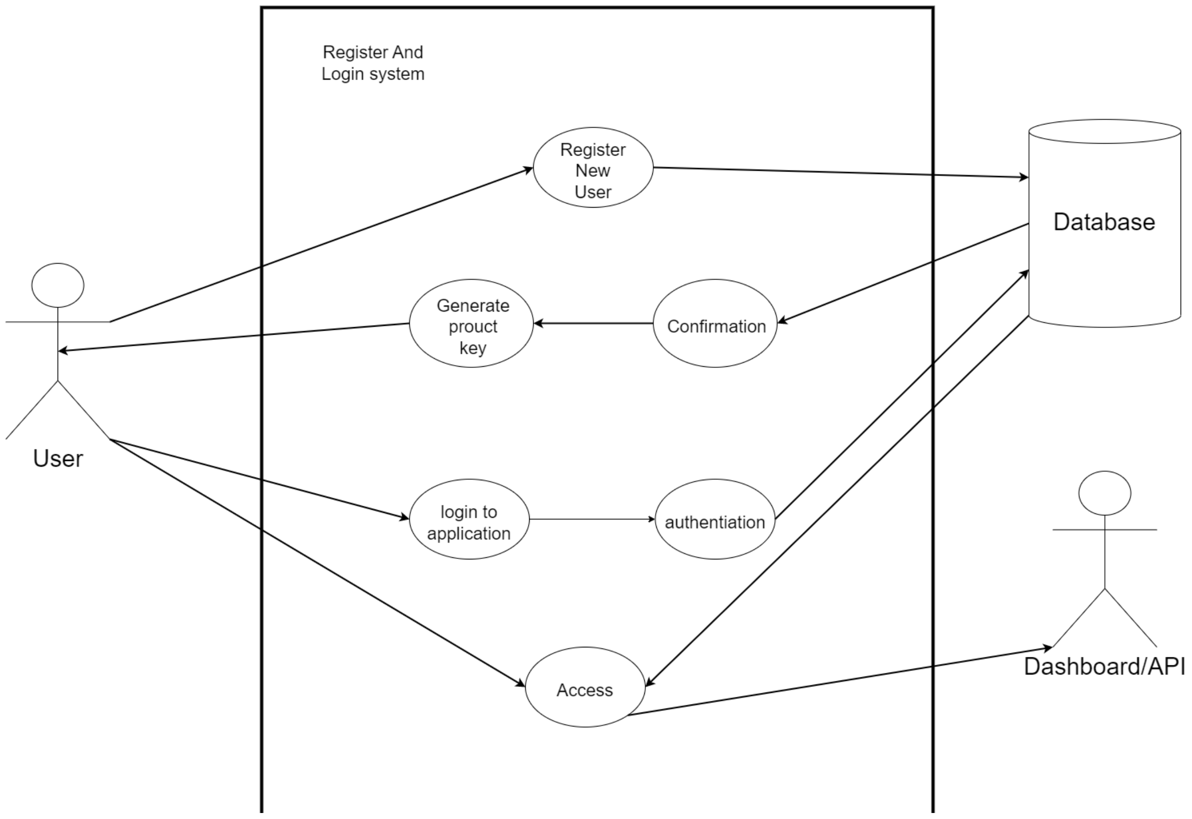
9. Delete document
    - The user chooses to delete document option.
    - Message will appear with the text “selected dictionary will be deleted from dictionary!”.
    - The user confirms.
    - System redirects to document list.

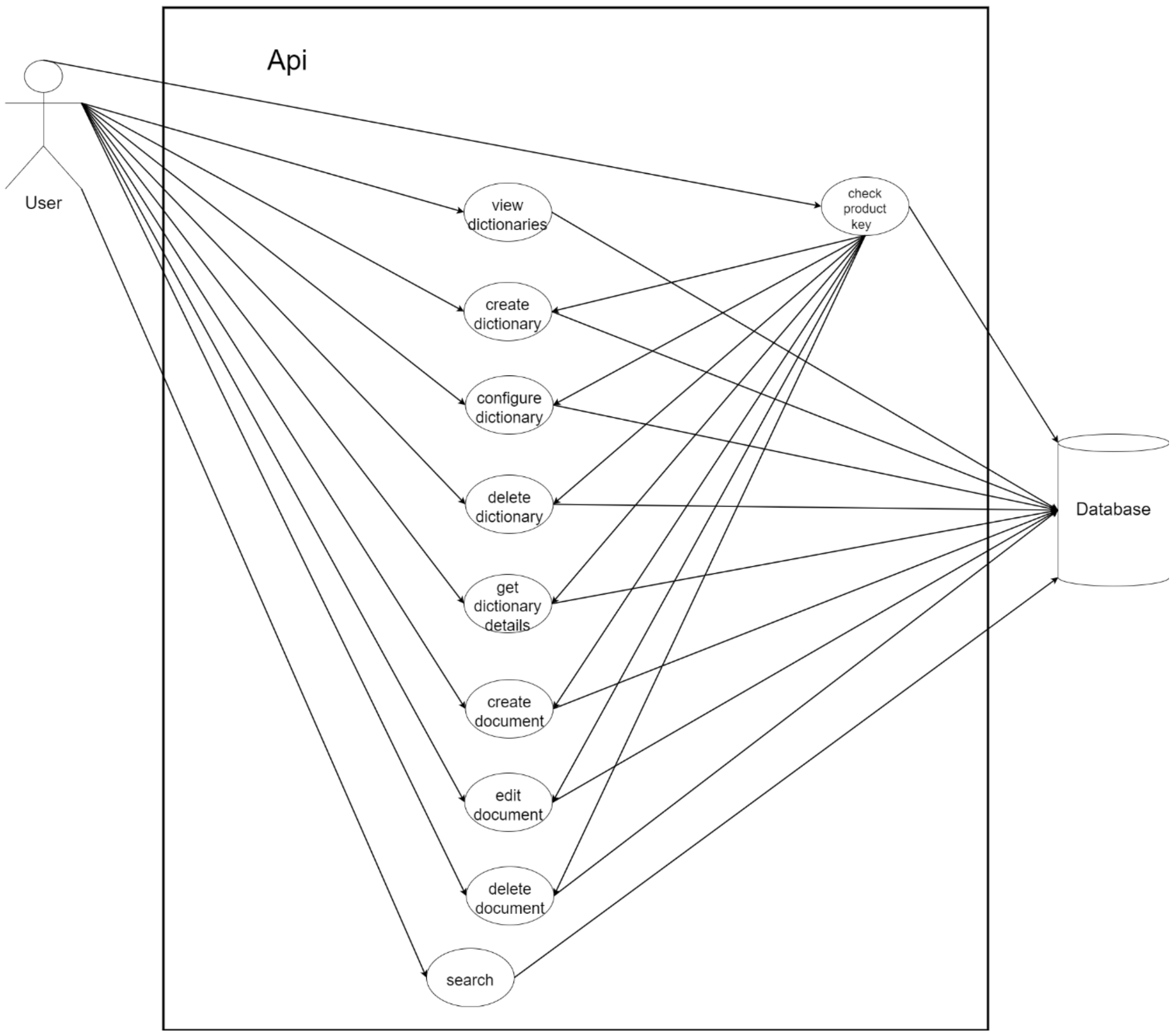
10. Edit document
    - The user chooses to edit document option.
    - Selected documents edit page will be appear with fields document name, file import option, textbox for manual import, dictionary category, and save button.
    - The user fills the information and chooses to save.
    - Message with text “document saved successfully!” will appear.
    - System redirects to document list.

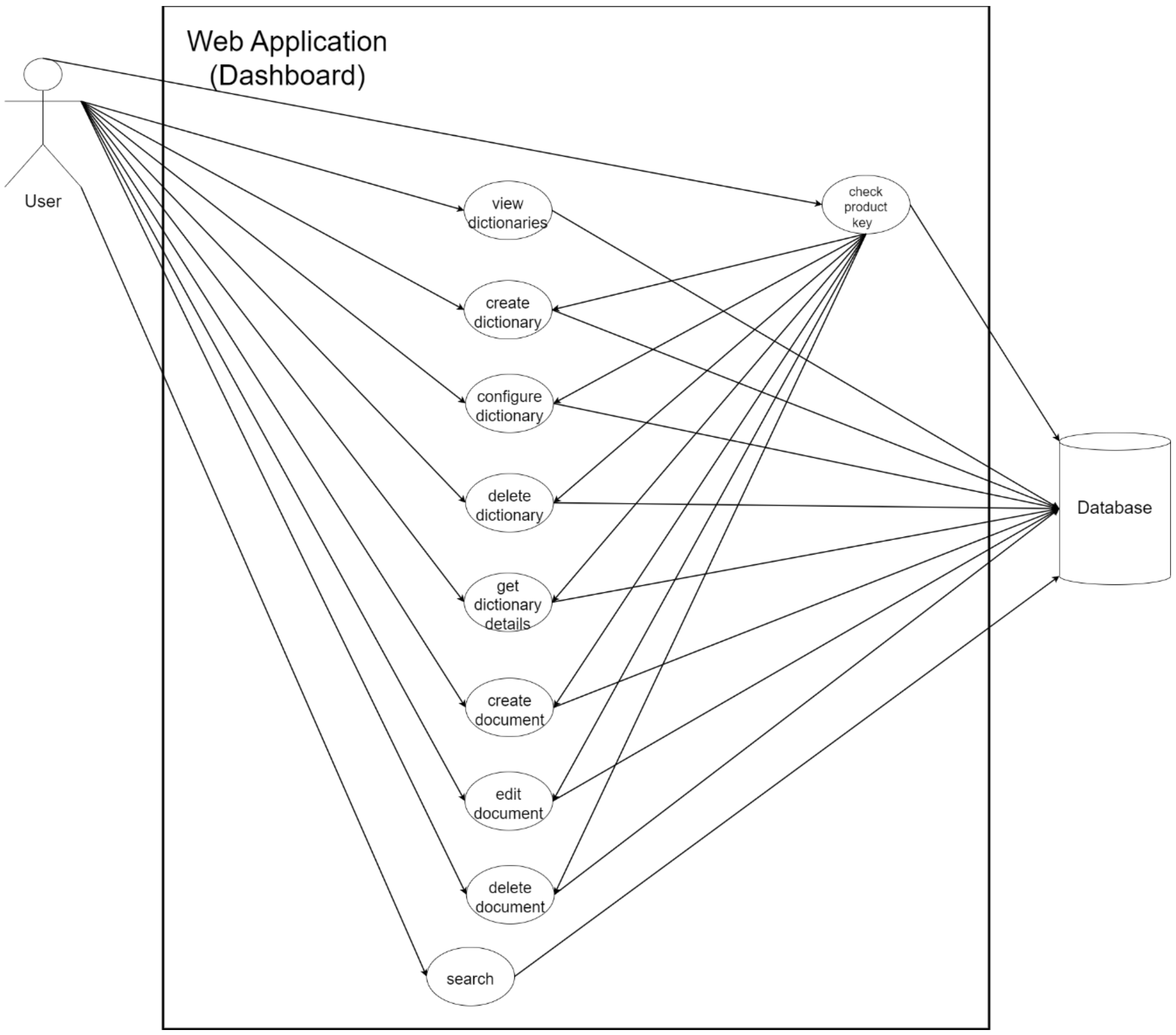
11. Implementation of API
    - API implementation page will appear with the information section and product key.
    - The user will read the document and implement the API to their own system.

12. Perform search
    - The user selected the search.
    - Main search algorithms will work, and result will be listed.

2.4.2 Use case model



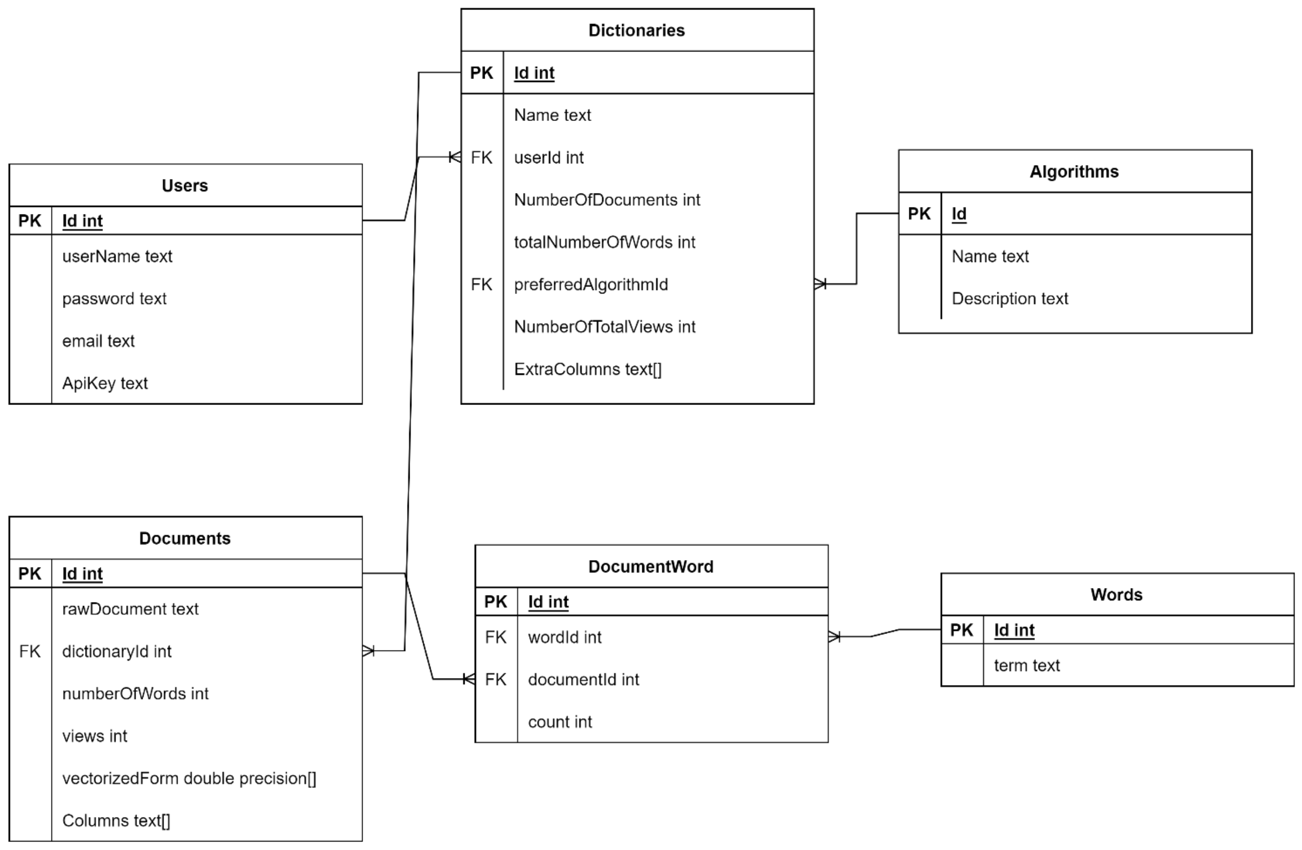

Api
User
view dictionaries
create dictionary
configure dictionary
delete dictionary
get dictionary details
create document
edit document
delete document
search
check product key
Database


Web Application (Dashboard)
User
view dictionaries
create dictionary
configure dictionary
delete dictionary
get dictionary details
create document
edit document
delete document
search
check product key
Database

2.4.3 Object and class model

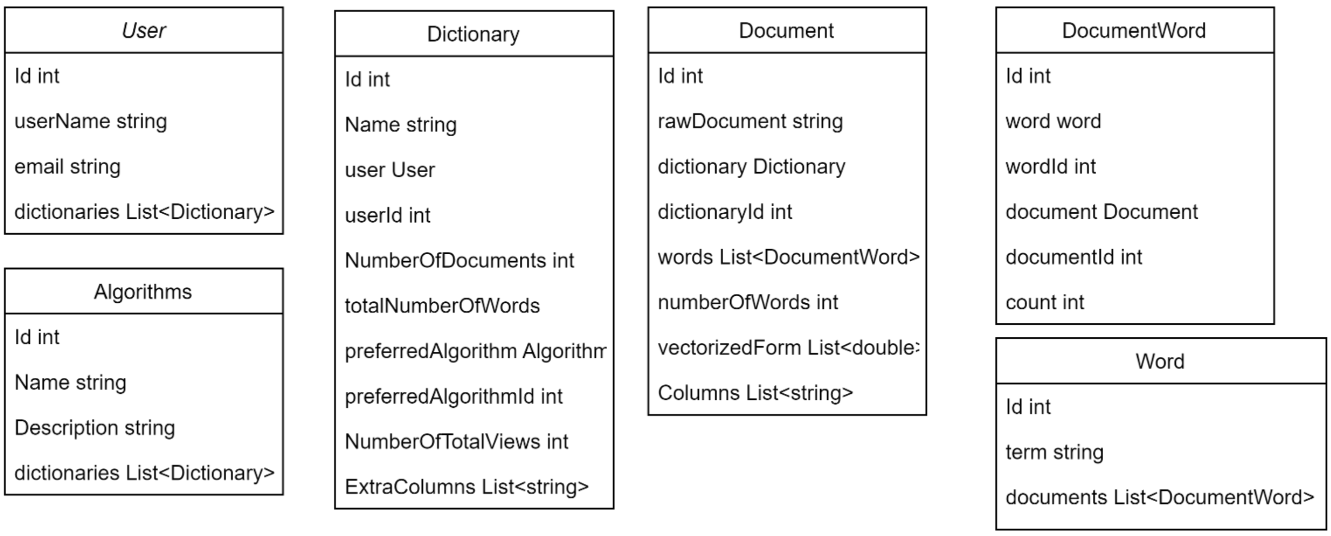
As it is unpractical to list all the classes in the API, there is the divided version that contains only important classes.

Our solutions database design will be as such;



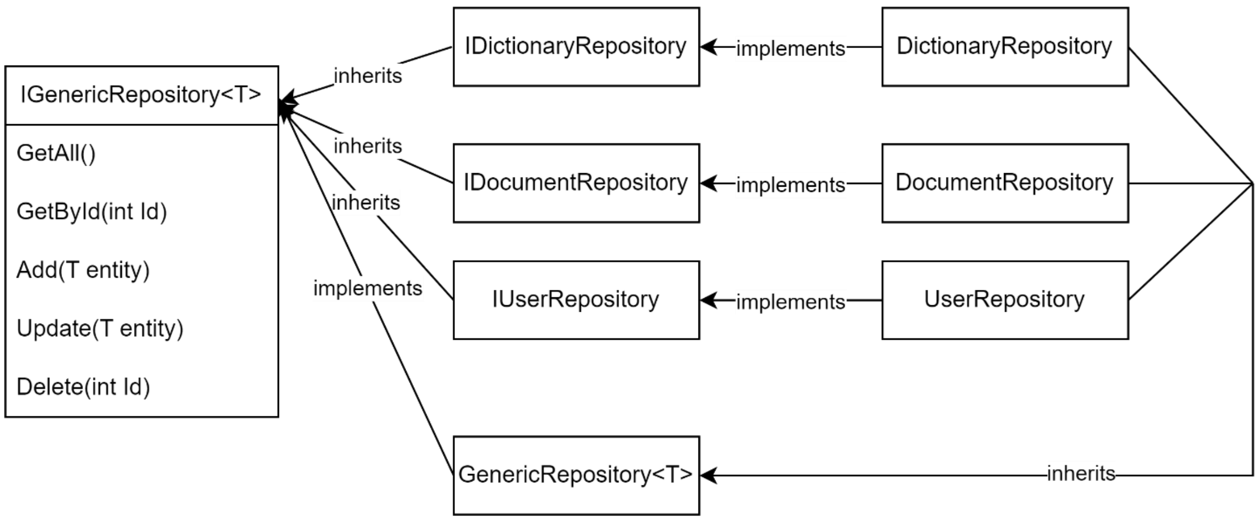
As you can see from the design we have our users that can have multiple dictionaries. Our dictionaries serve a purpose like a folder, it holds our documents and some information about them. The extra columns field defines the virtual columns the user would like to store in their documents, important to note that all the documents will have the same extra columns defined in their dictionary. When a document is added if a word that is not present in the words table is found a new word is created in the words table, and a new entry is created in the documentWord table that holds the word's Id and the document's Id so that the word document pair is linked.

To manage this database system we will use a ORM, object relational mapping system. The advantage of such a system provides is that the API can easily find what its looking for without us needing to create complex SQL queries, the system resolves that for us. The objects related to our database tables are as follows.

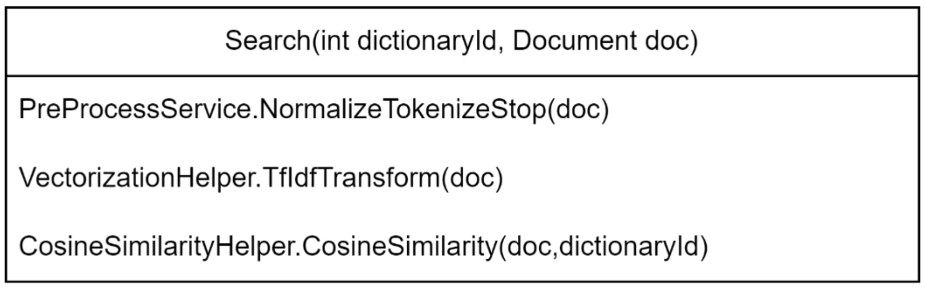


These objects are the models that is used in the object relational mapping. These objects contain the information of one to many and one to one relationships, such as the dictionary belongs to a user, to fetch the user we can just call the dictionary's user property and it will give the user object.

Taking a look at where the logic happens, we have a repository pattern in our API, so all the data fetching and the requested functions to be used are present on the repositories. This pattern sperate our concerns and frees controllers from code duplication and complexity as much as possible.



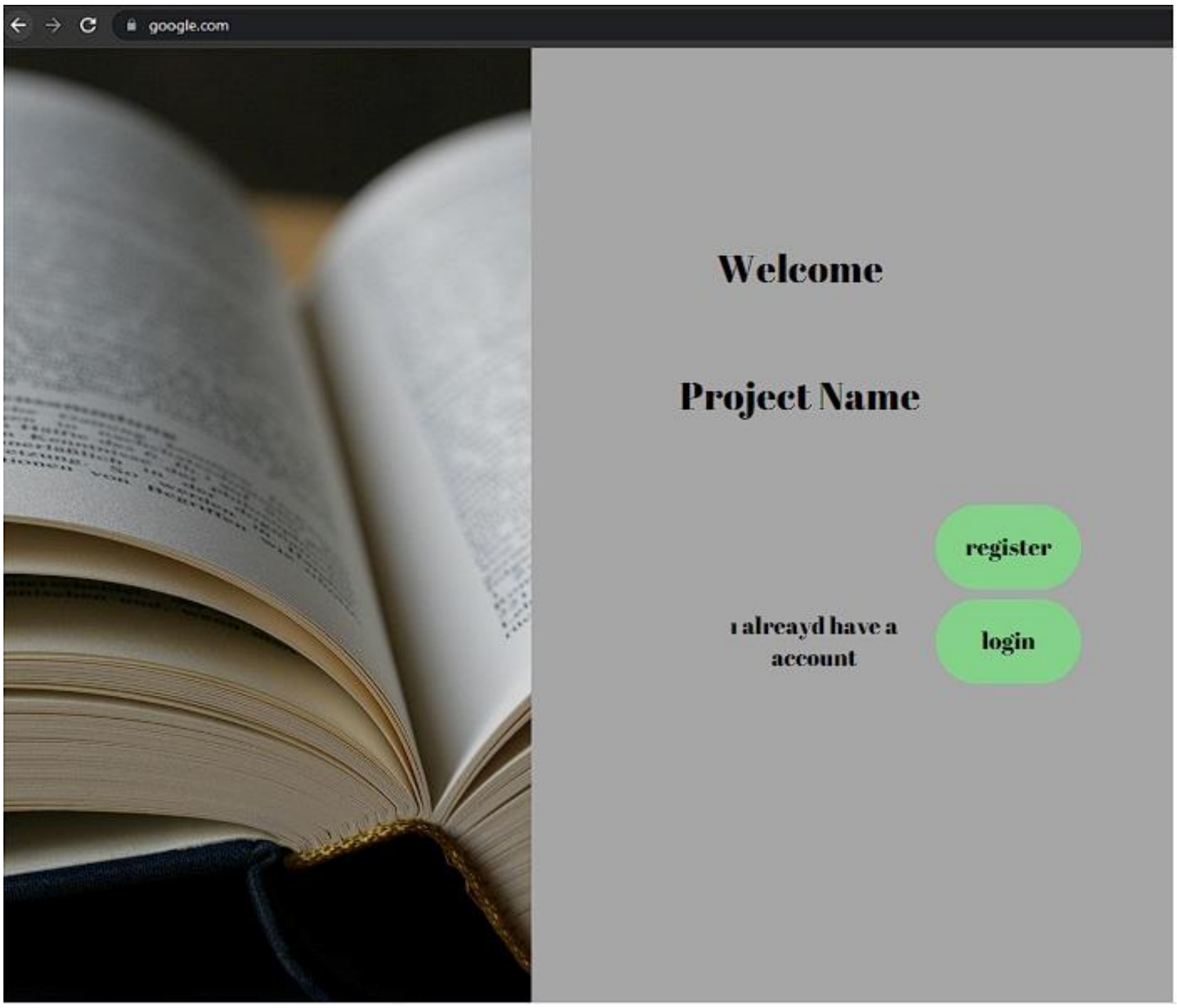
Using this we implement our repositories as scoped services to be used by the endpoints of the controllers. As the project requires more methods we can easily extend and be flexible with our implementations with as little code duplication as possible. Taking a look at a search function in the Document repository.

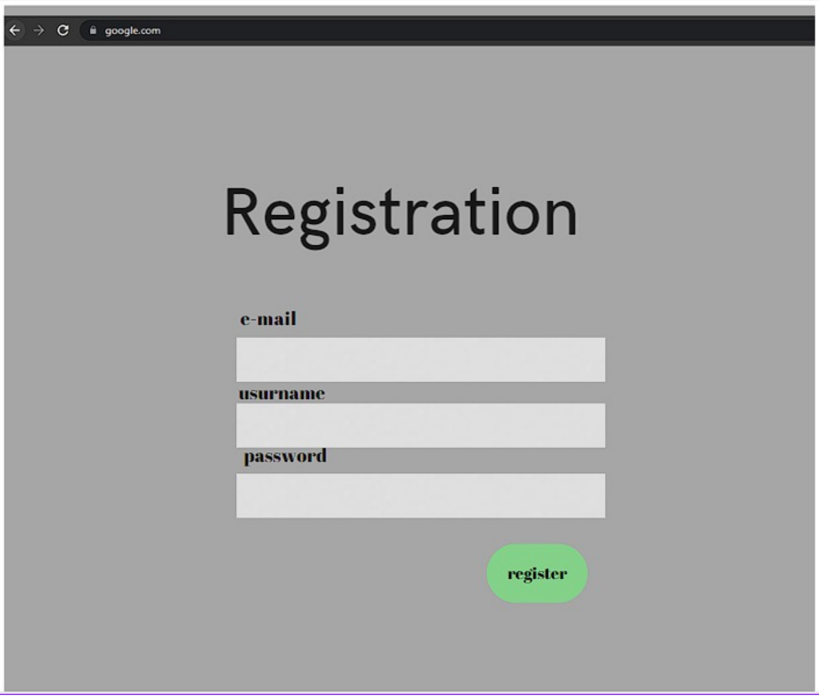


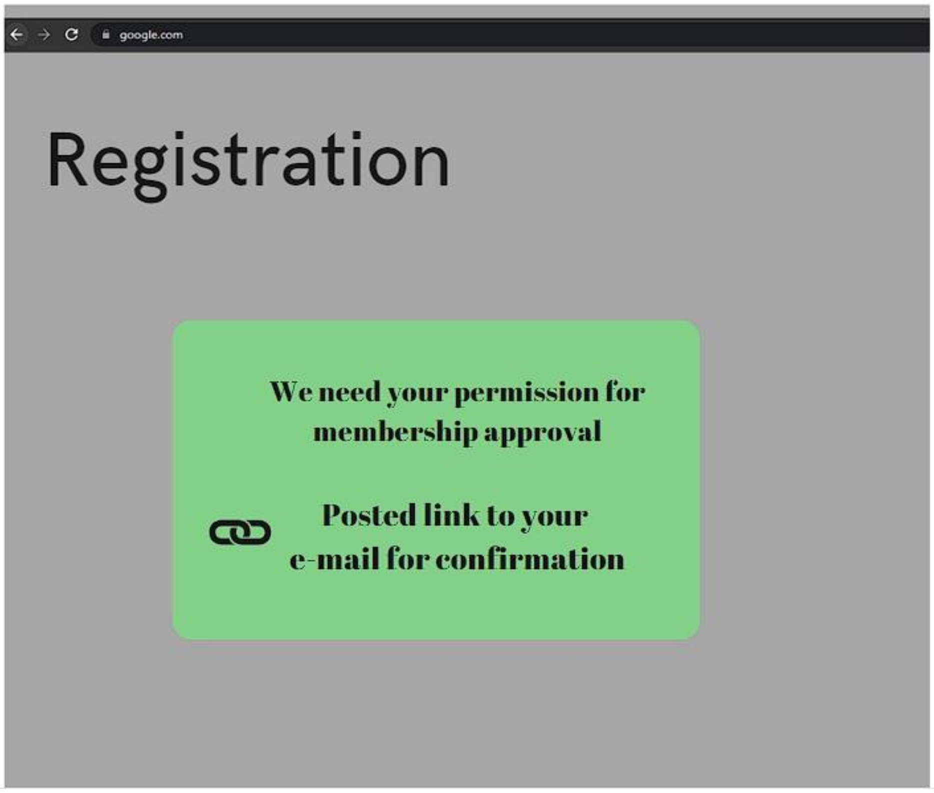
```
Search(int dictionaryId, Document doc)

PreProcessService.NormalizeTokenizeStop(doc)

VectorizationHelper.TfIdfTransform(doc)

CosineSimilarityHelper.CosineSimilarity(doc,dictionaryId)
```

The project structure revolves around controller calling a repository, repository utilizes the services and helper classes in the project.

2.4.5 User interface - navigational paths and screen mock-ups

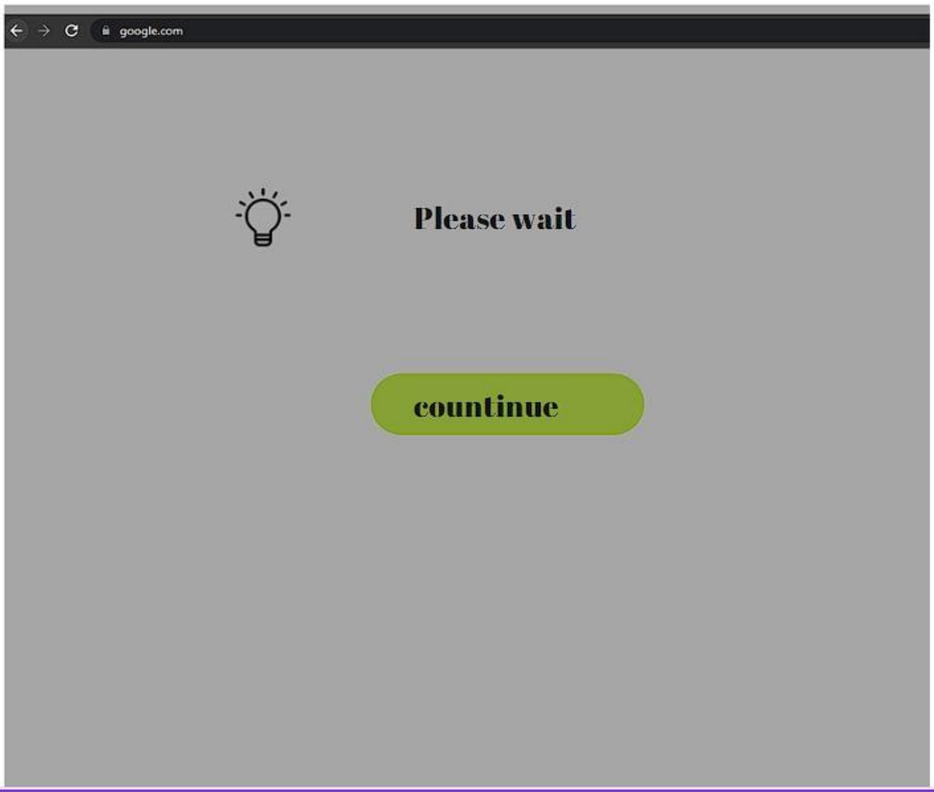


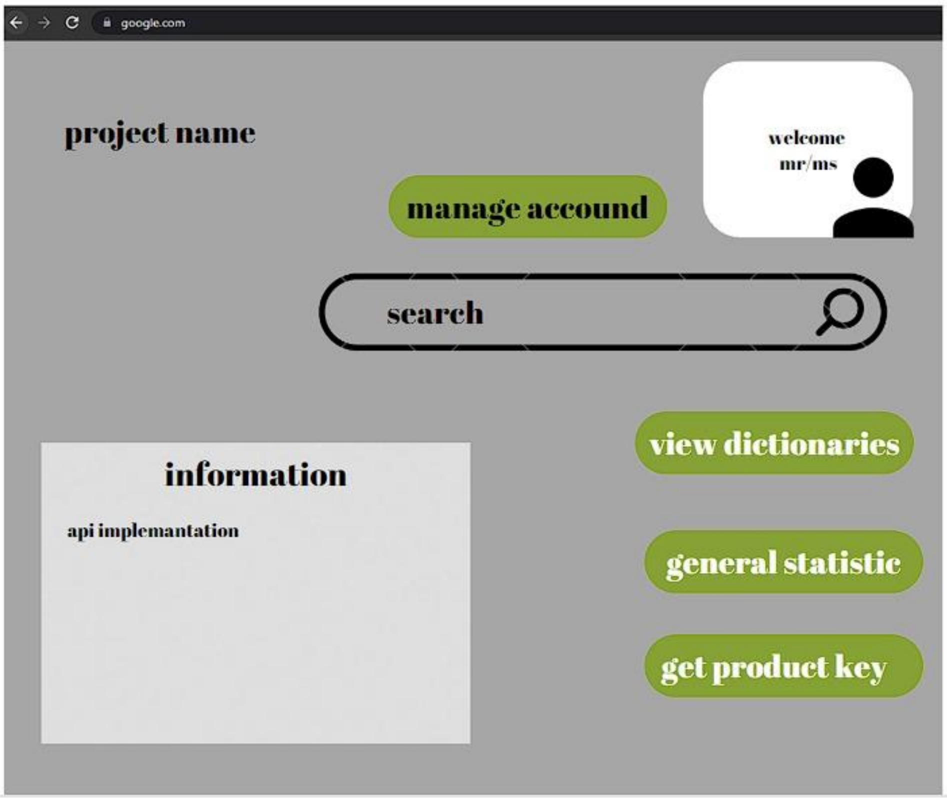


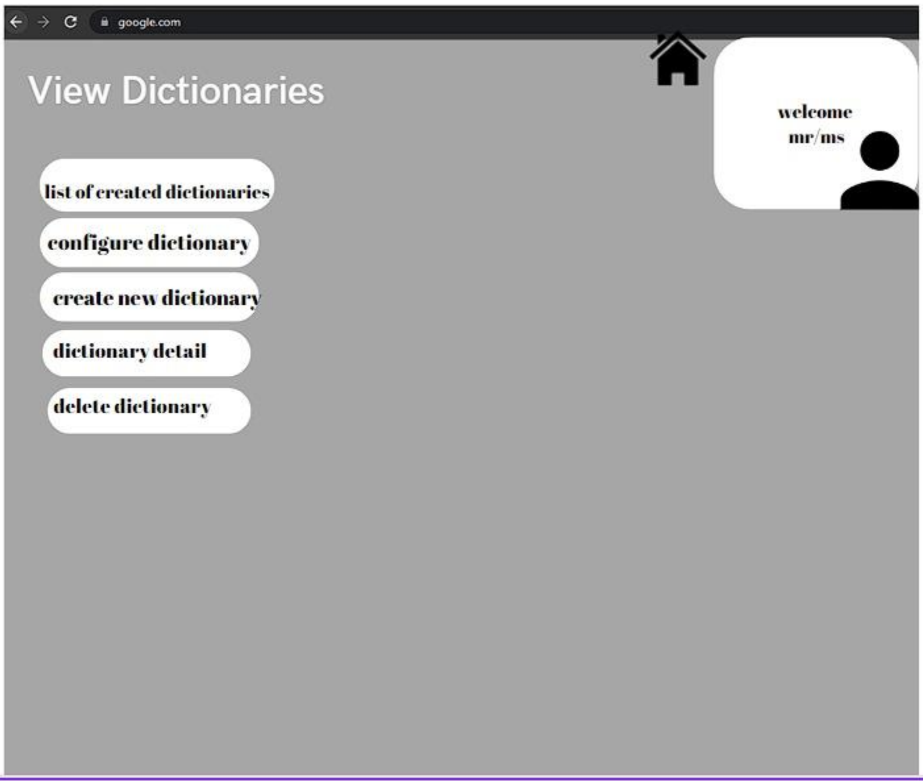


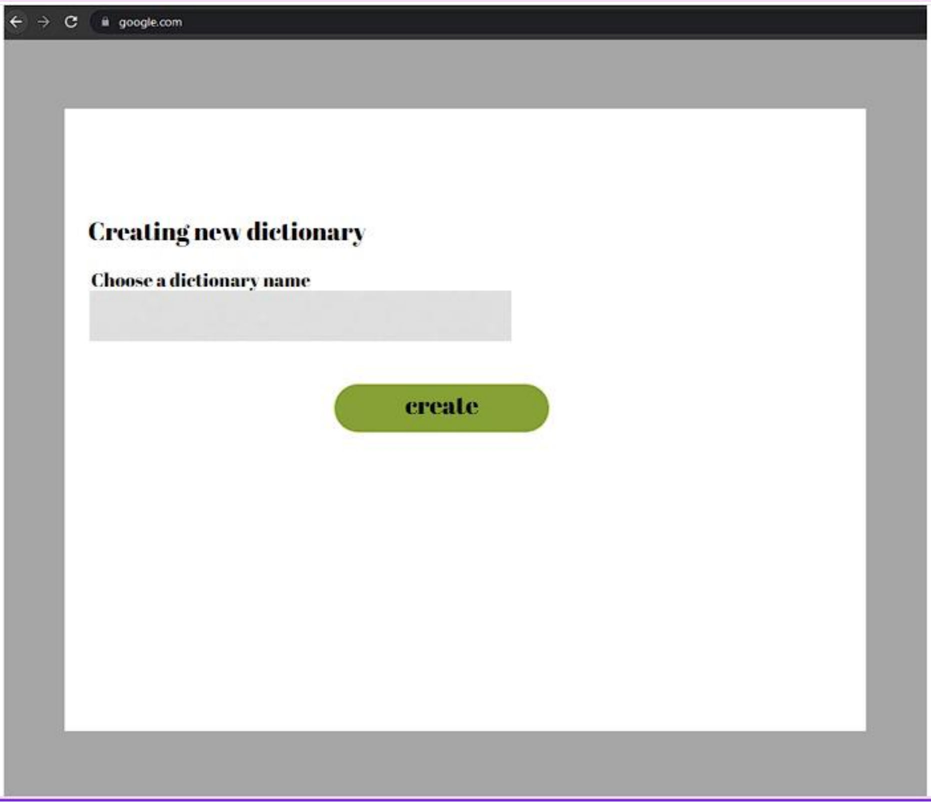
We need your permission for membership approval

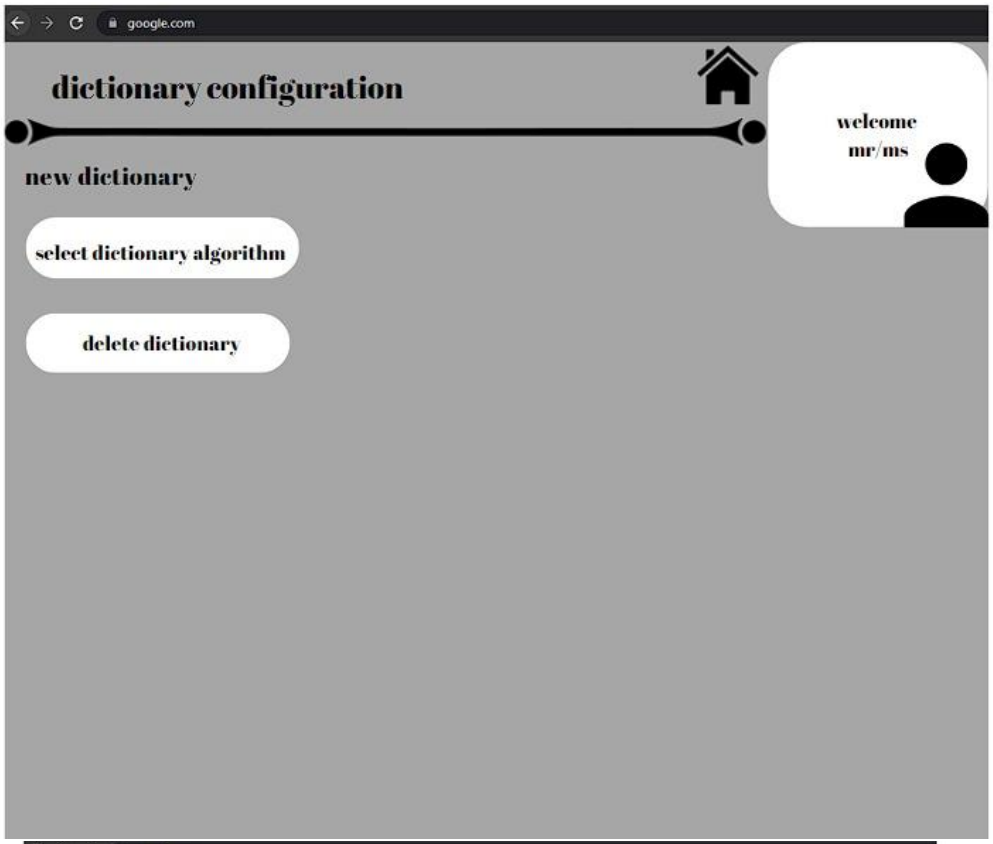
Posted link to your e-mail for confirmation

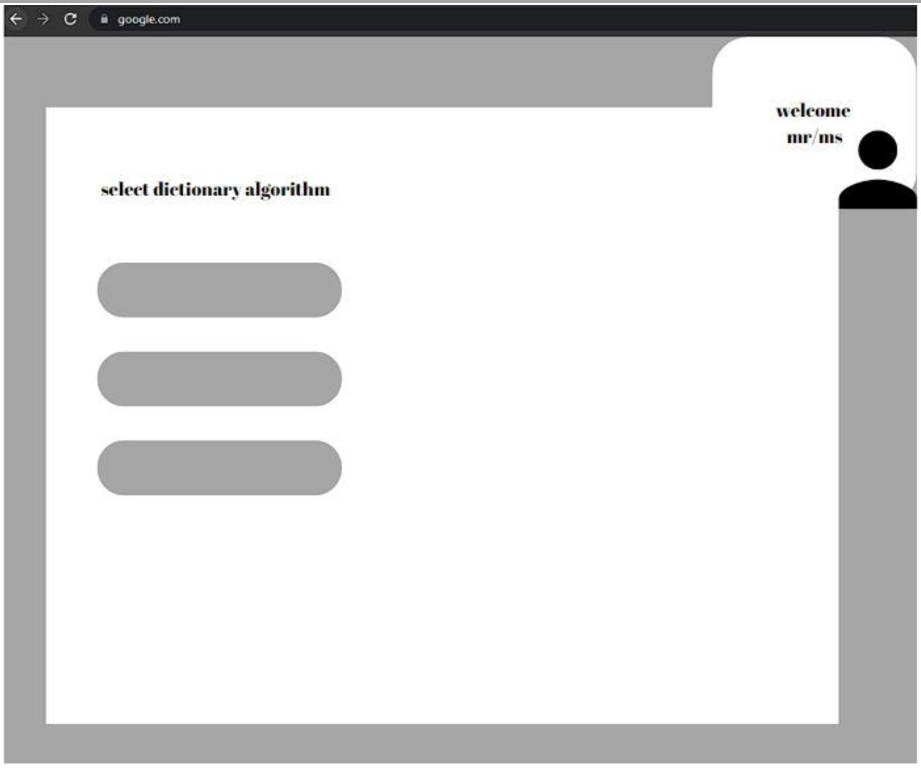

google.com
Please wait
countinue

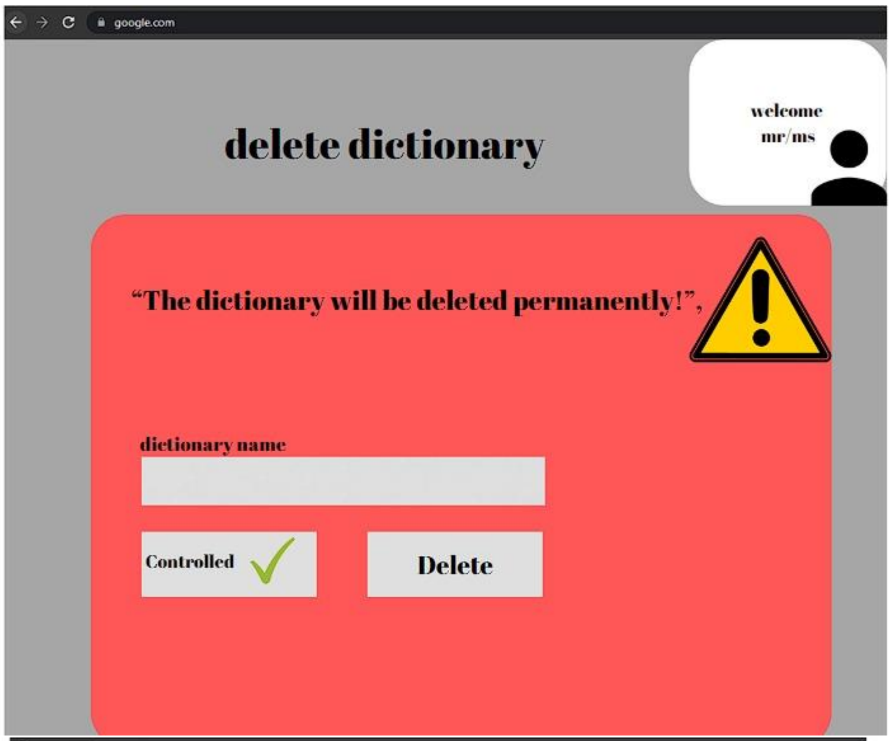

google.com
project name
welcome
mr/ms
manage accound
search
information
api implemantation
view dictionaries
general statistic
get product key

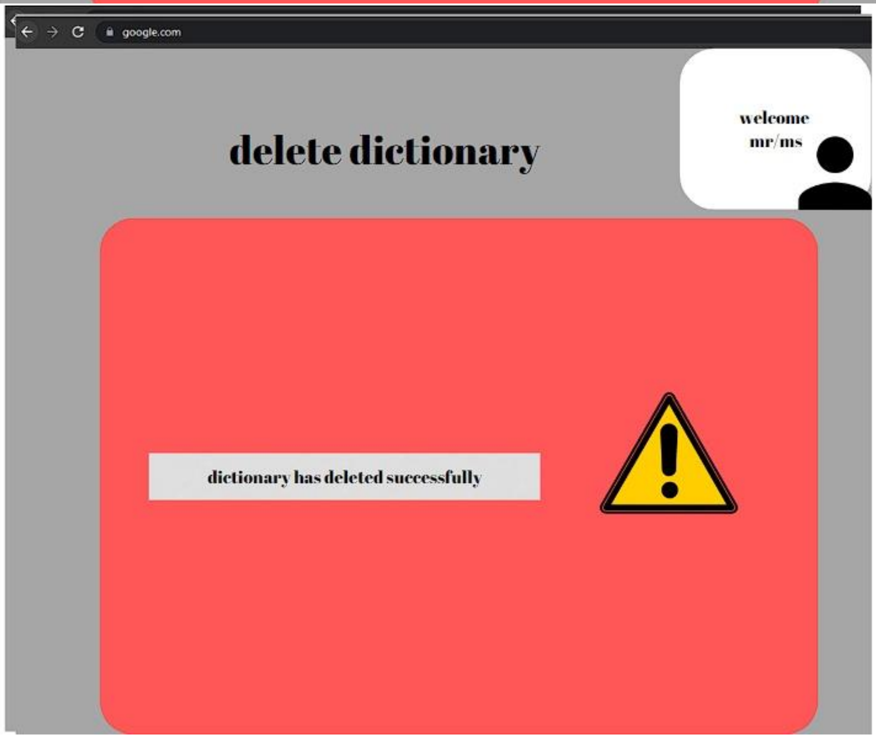

google.com
View Dictionaries
welcome mr/ms
list of created dictionaries
configure dictionary
create new dictionary
dictionary detail
delete dictionary

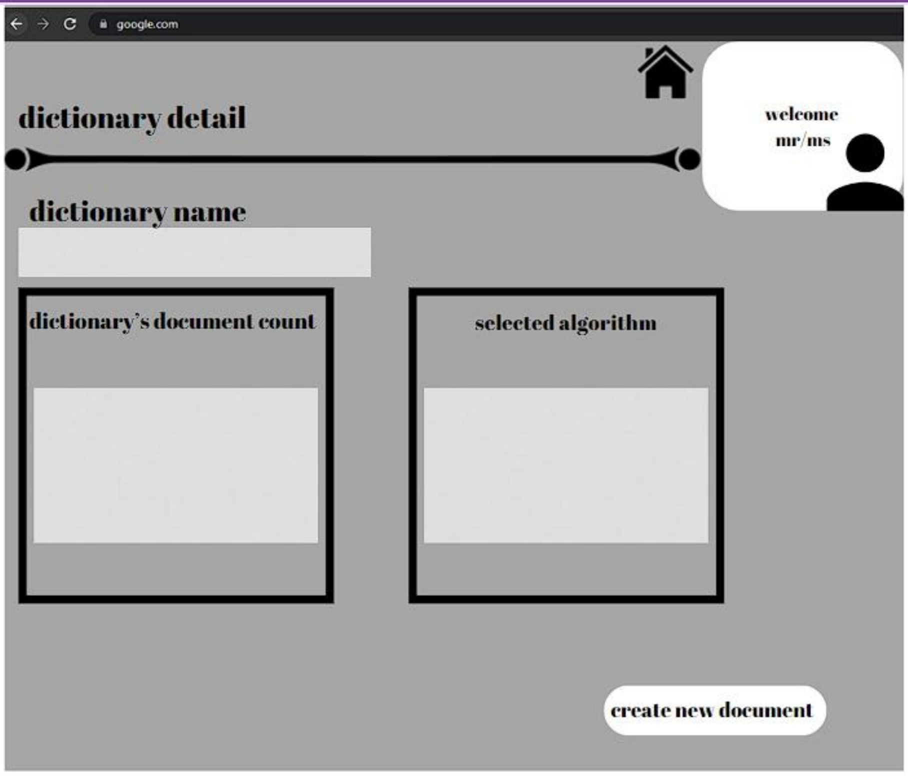

google.com
Creating new dictionary
Choose a dictionary name
create

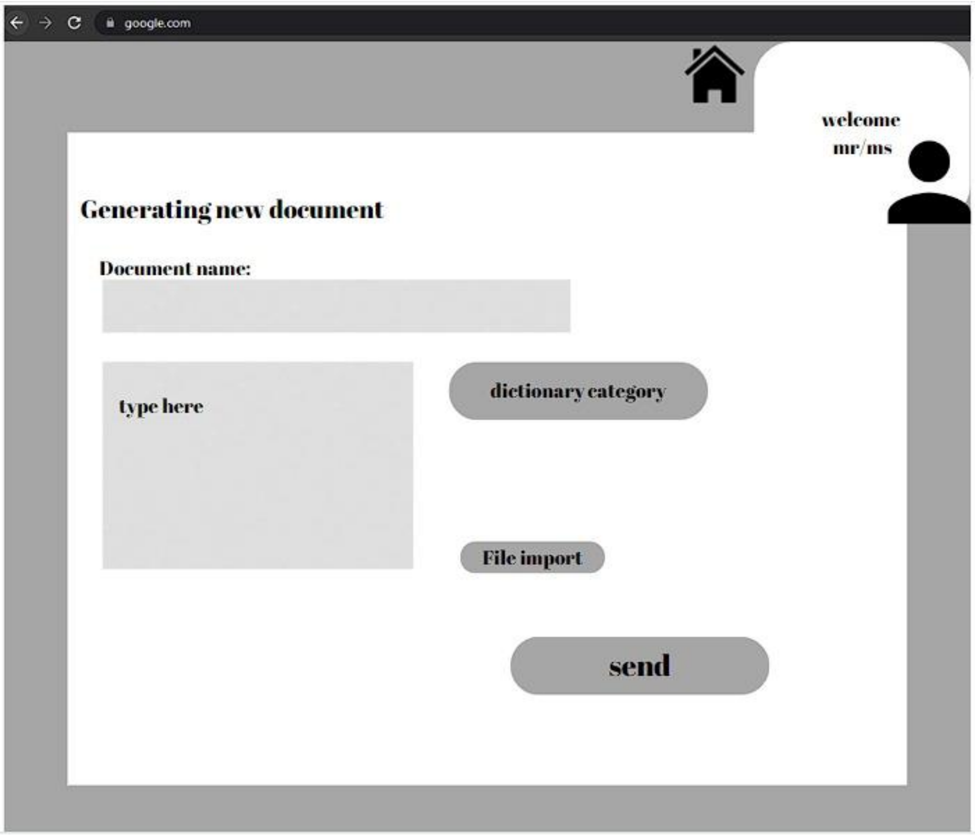

google.com
dictionary configuration
welcome
mr/ms
new dictionary
select dictionary algorithm
delete dictionary

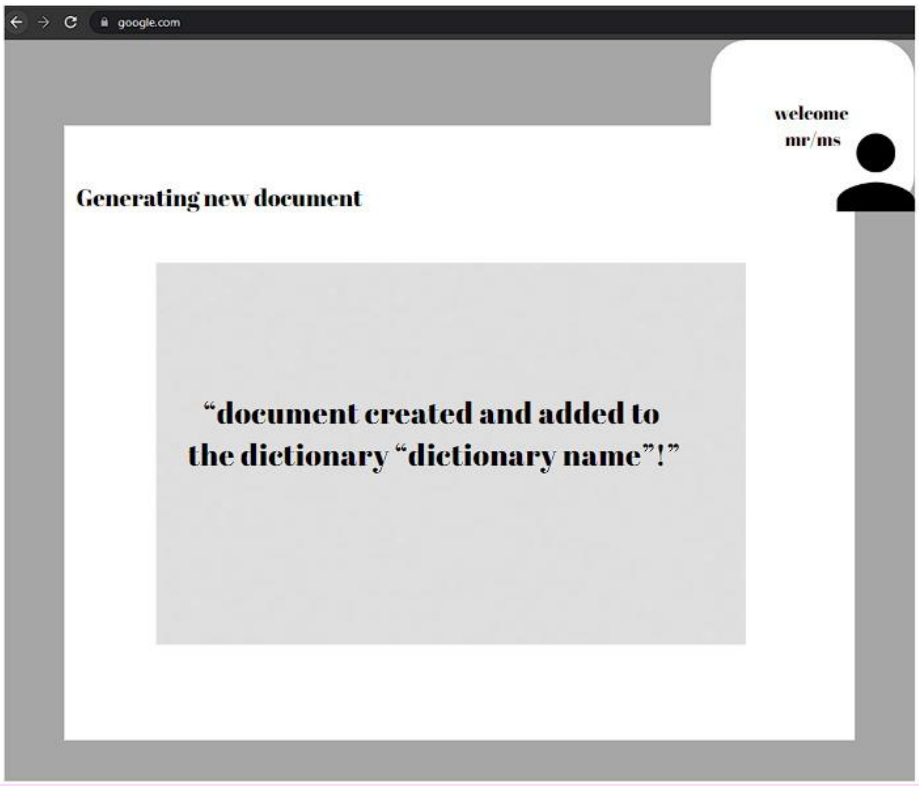

google.com
welcome
mr/ms
select dictionary algorithm


google.com
delete dictionary
welcome
mr/ms
"The dictionary will be deleted permanently!",
dictionary name
Controlled
Delete


google.com
delete dictionary
welcome
mr/ms
dictionary has deleted successfully


google.com
dictionary detail
welcome
mr/ms
dictionary name
dictionary's document count
selected algorithm
create new document


google.com
welcome
mr/ms
Generating new document
Document name:
type here
dictionary category
File import
send

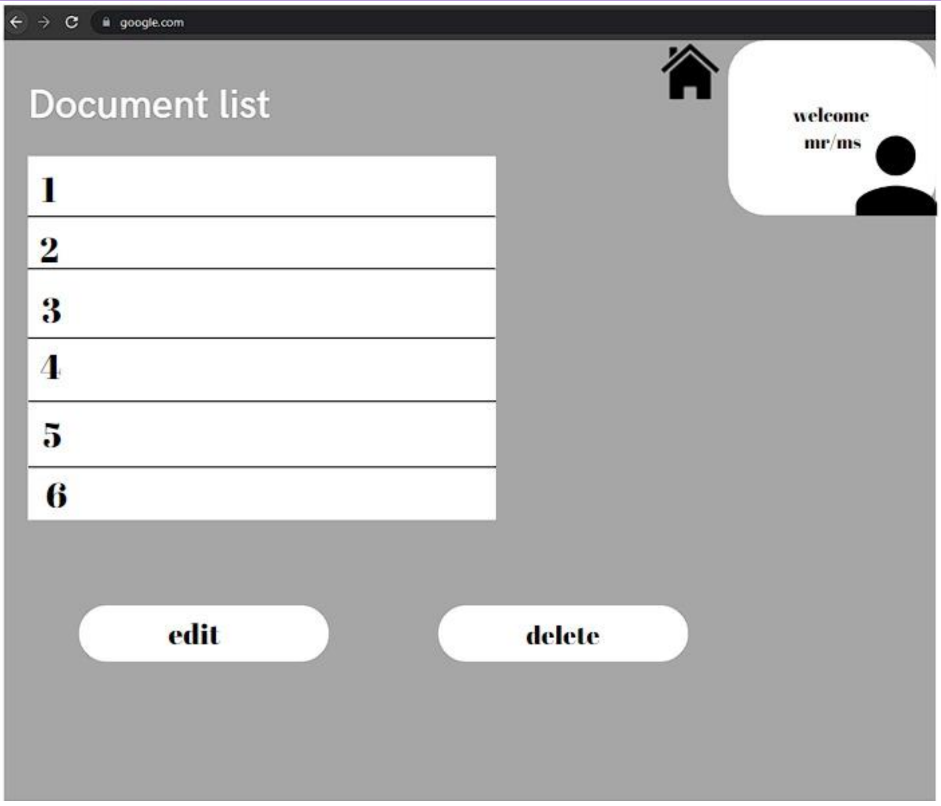


google.com

welcome mr/ms

Generating new document

"document created and added to the dictionary "dictionary name"!"



google.com

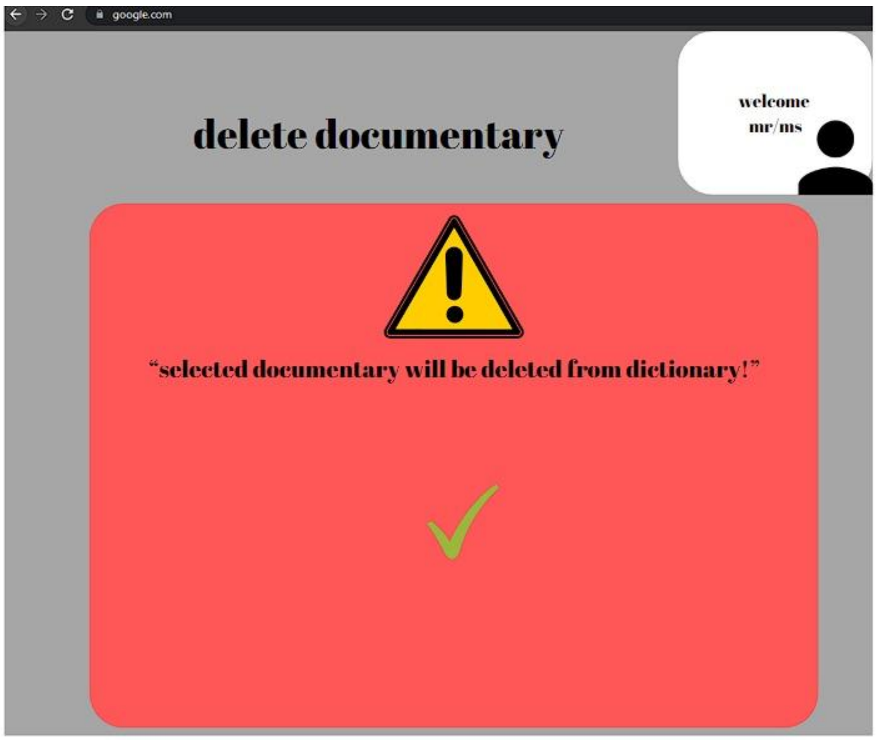
Document list

welcome mr/ms

1
2
3
4
5
6

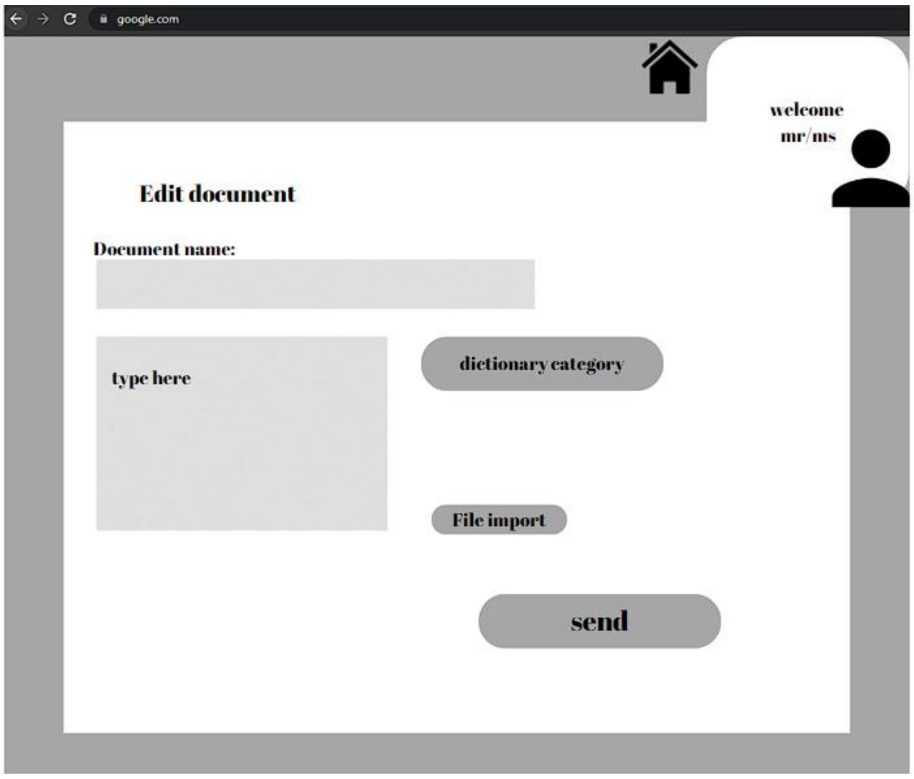
edit

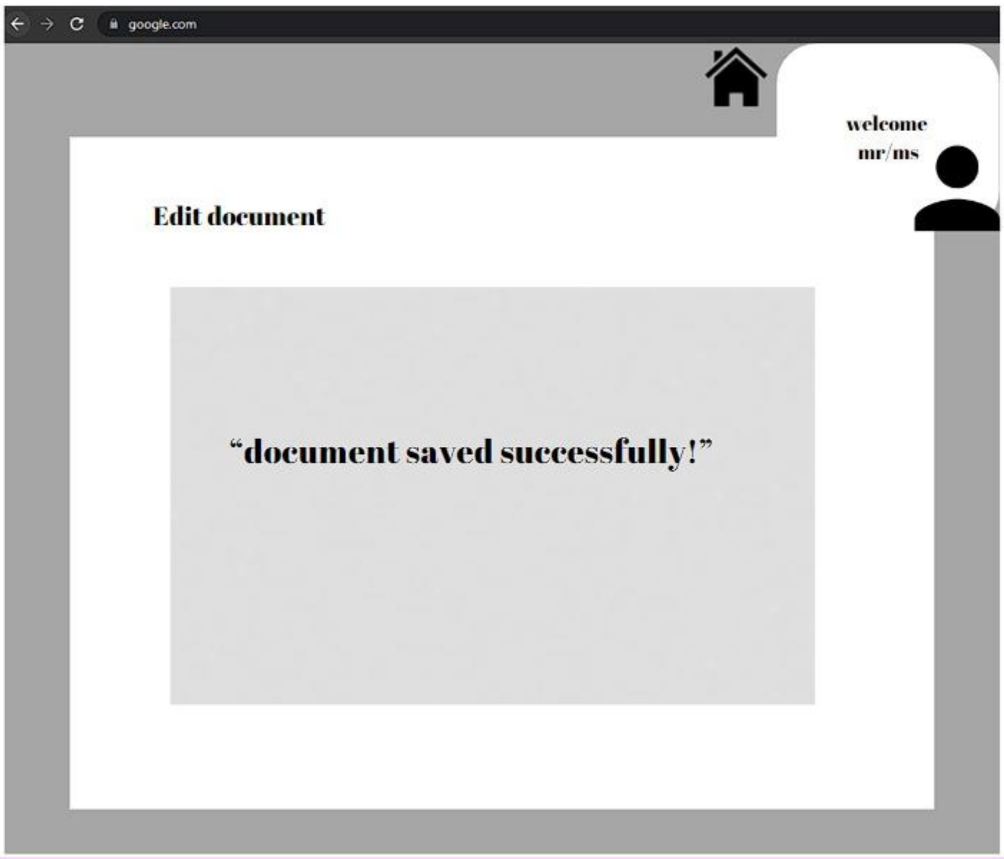
delete



delete documentary

"selected documentary will be deleted from dictionary!"



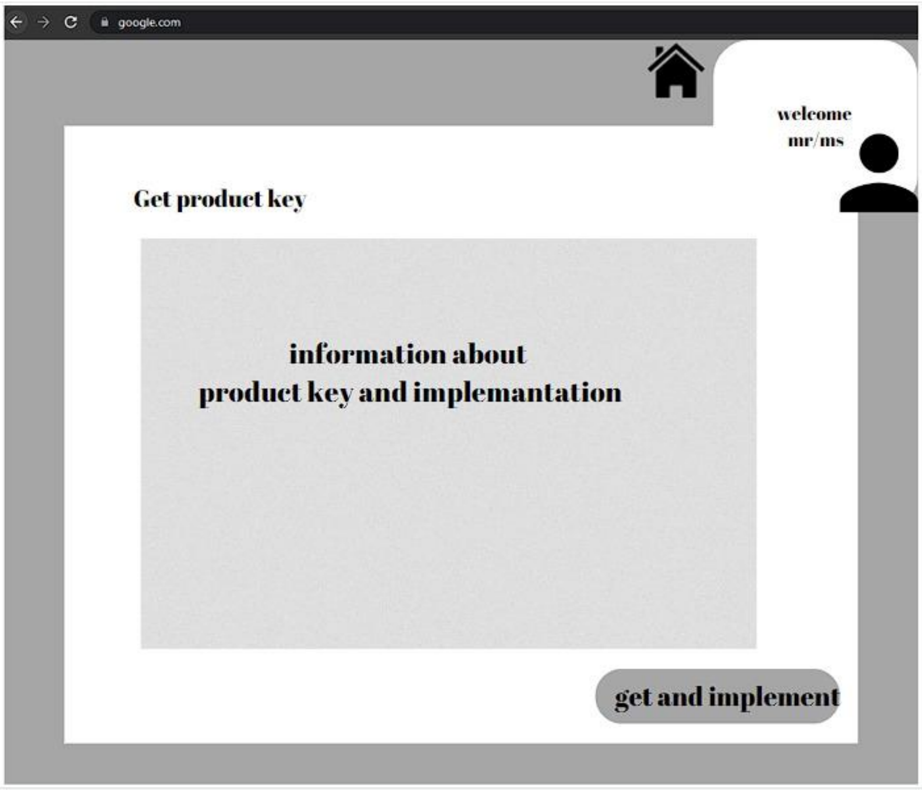


google.com

welcome mr/ms

Edit document

"document saved successfully!"



google.com

welcome mr/ms

Get product key

information about product key and implemantation

3 Glossary

API: An application programming interface is a way for two or more computer programs to communicate with each other.
Repository: A software repository, or repo for short, is a storage location for software packages.
ORM: We will use for managing to data base. Object–relational mapping in computer science is a programming technique for converting data between type systems using object-oriented programming languages