LingoGym: A Language Learning Web Application

Independent Study Project Presentation

By Yangzom Dolma

Team member: Biruk Worku

April 11, 2022

University of Oklahoma

Introduction

- A web application tool for English as a second/third language learners
- Target audience: Tibet and Ethiopia
- Use image recognition technologies

Background and Motivation

- English: important for social mobility, opportunities, career development
- Difficulty with English learning among some groups of Tibetans
    - Elderly Tibetan refugees in India
    - Elderly Tibetan immigrants in the USA
    - Tibetans in Tibet
- Only 0.22% of Ethiopians speak English as of 2014 [4].

Problem Statement and Objective

- Group of Tibetans/Ethiopians
  - Knows native spoken language only
  - Don't know how to read/write in their native language
  - Struggle with finding resources to learn language
  - Leads to difficulties in using Technology (mostly English-based)
- Objective
  - Create an inclusive web application
  - Flexible platform for users irrespective of their background
  - Image and keyword search facilitation

Existing works

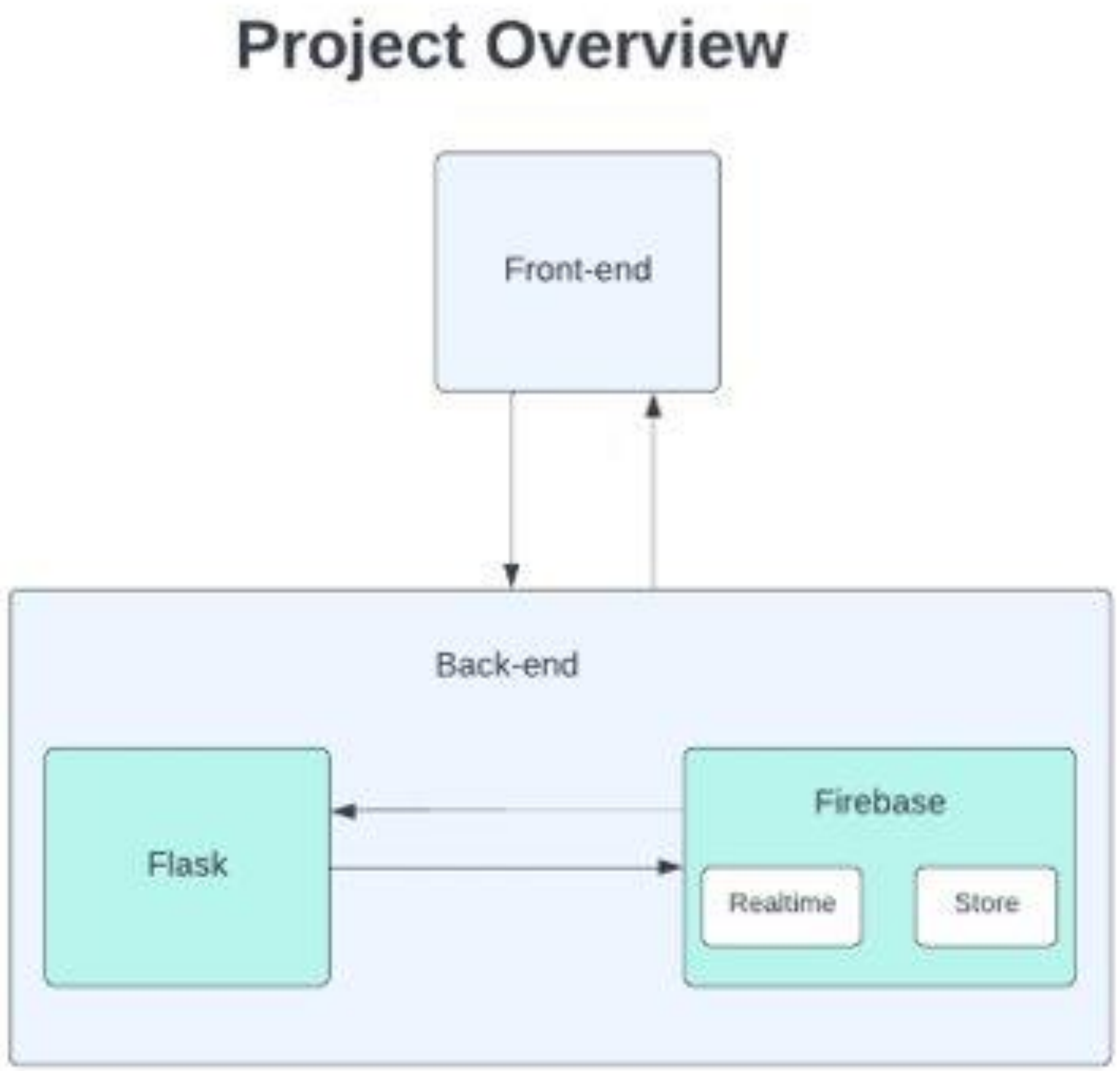
- English as a second language learning web sites
  - Manythings.org
  - BBC learning English website and app
  - Duolingo
  - Babbel
- What do they lack?

What makes this project different?

- Language learning tool
  - Targeted for Tibetan and Amharic
- Provide search by image feature
  - Content-based Image Retrieval technique
- Tibetan and Amharic focus
  - Both has less than 0.1% internet presence
  - Tibetan: No Google Translate support

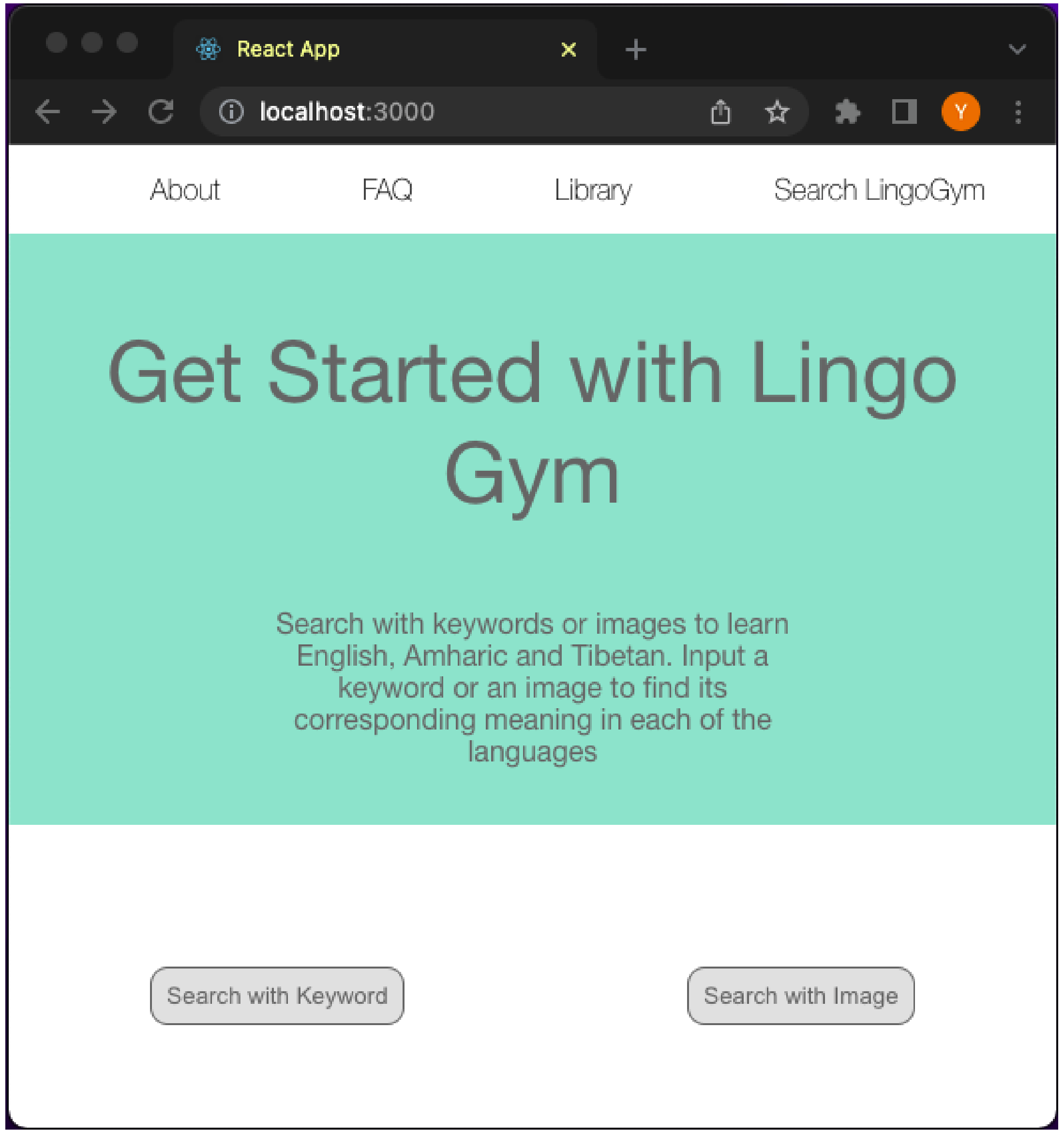
Project Components

- Front-end
- Back-end
    - Flask
    - Firebase
        - Realtime database
        - Firebase Storage
- CMS



Front end

- ReactJS
    - A JS library built and maintained by FB
- Flexible open-source library
- Industry giants: Apple, Netflix etc.

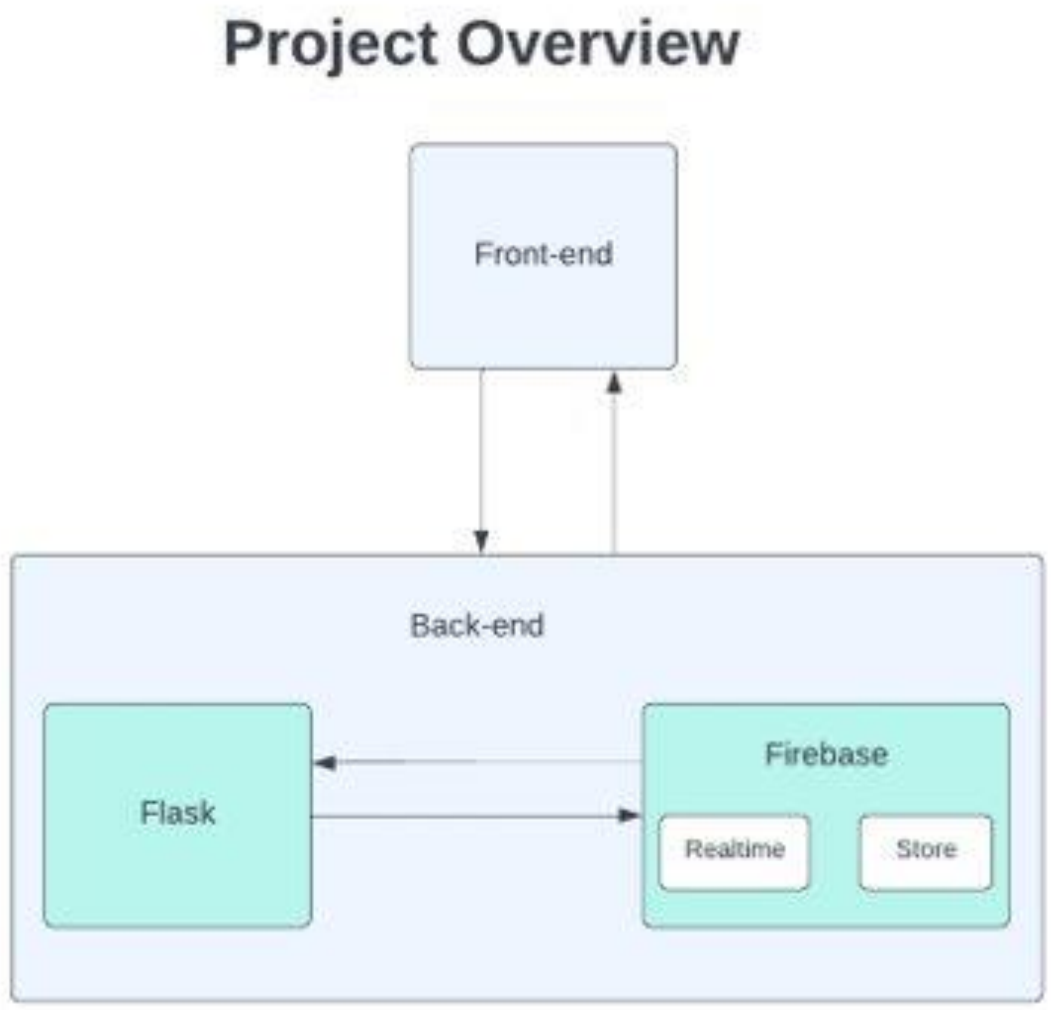


Get Started with Lingo Gym

Search with keywords or images to learn English, Amharic and Tibetan. Input a keyword or an image to find its corresponding meaning in each of the languages

Back end

- Flask
    - Used for developing web app using python
- Firebase
    - Realtime database
    - Firebase storage



Back end: Design decision factors

Flask:

- CBIR technique implementation
  - VGG16
  - Keras library from python

Firebase

- Realtime database
- Firebase storage
- Great API support
- Appropriate with the short development time

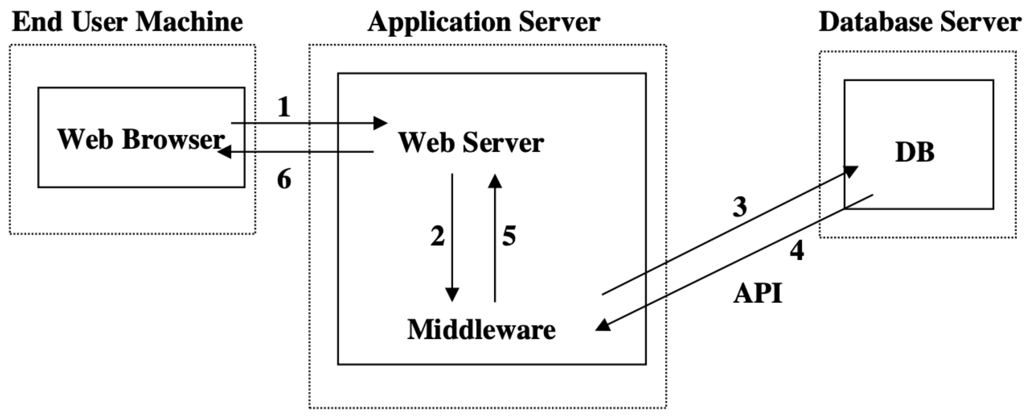
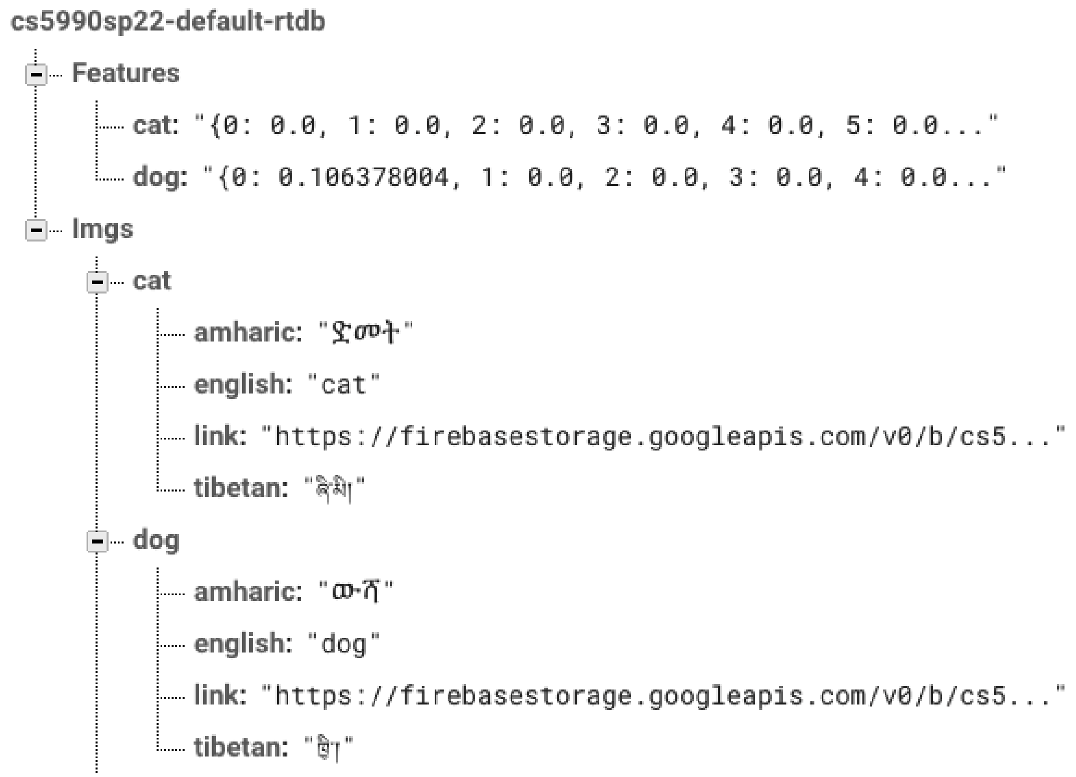


Figure 1. A generic web architecture

Back-end: Realtime database

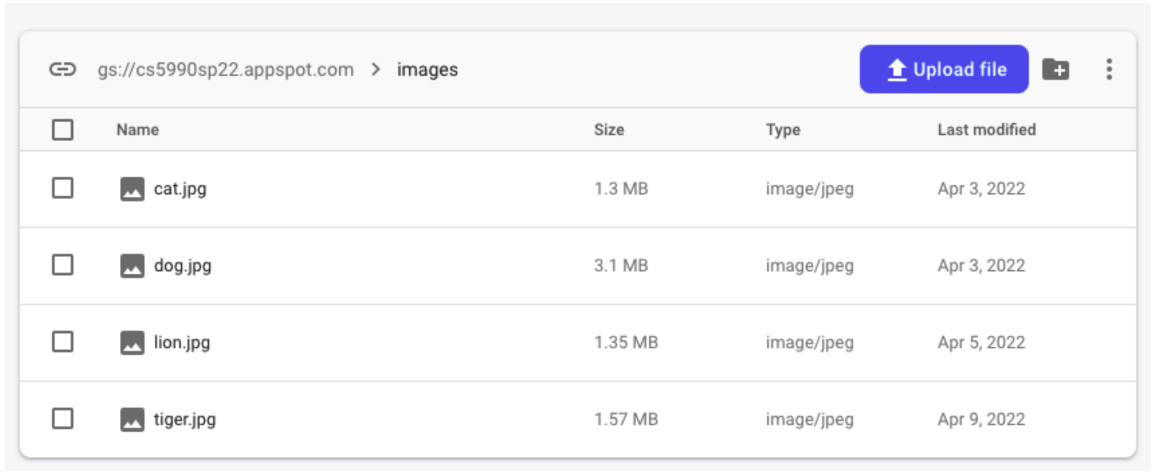
1. Feature vectors
   - For each image in the data set
   - Key-value pair
   - Index, vector element
   - Used to process search by an image
2. Image tags and Info
   - Tag in Tibetan & Amharic
   - Link to the image in fire storage
   - Usable for API calls to perform CRUD on the data set.



```
cs5990sp22-default-rtdb
 ⊟ Features
     cat: "{0: 0.0, 1: 0.0, 2: 0.0, 3: 0.0, 4: 0.0, 5: 0.0..."
     dog: "{0: 0.106378004, 1: 0.0, 2: 0.0, 3: 0.0, 4: 0.0..."
 ⊟ Imgs
     ⊟ cat
         amharic: "ድመት"
         english: "cat"
         link: "https://firebasestorage.googleapis.com/v0/b/cs5..."
         tibetan: "ཞིམི།"
     ⊟ dog
         amharic: "ውሻ"
         english: "dog"
         link: "https://firebasestorage.googleapis.com/v0/b/cs5..."
         tibetan: "ཁྱི།"
```

Back – end: Firebase Storage

- Stores media: files, images
- Links generated for each image
- Accessible via api-calls from Realtime database



Name	Size	Type	Last modified
cat.jpg	1.3 MB	image/jpeg	Apr 3, 2022
dog.jpg	3.1 MB	image/jpeg	Apr 3, 2022
lion.jpg	1.35 MB	image/jpeg	Apr 5, 2022
tiger.jpg	1.57 MB	image/jpeg	Apr 9, 2022

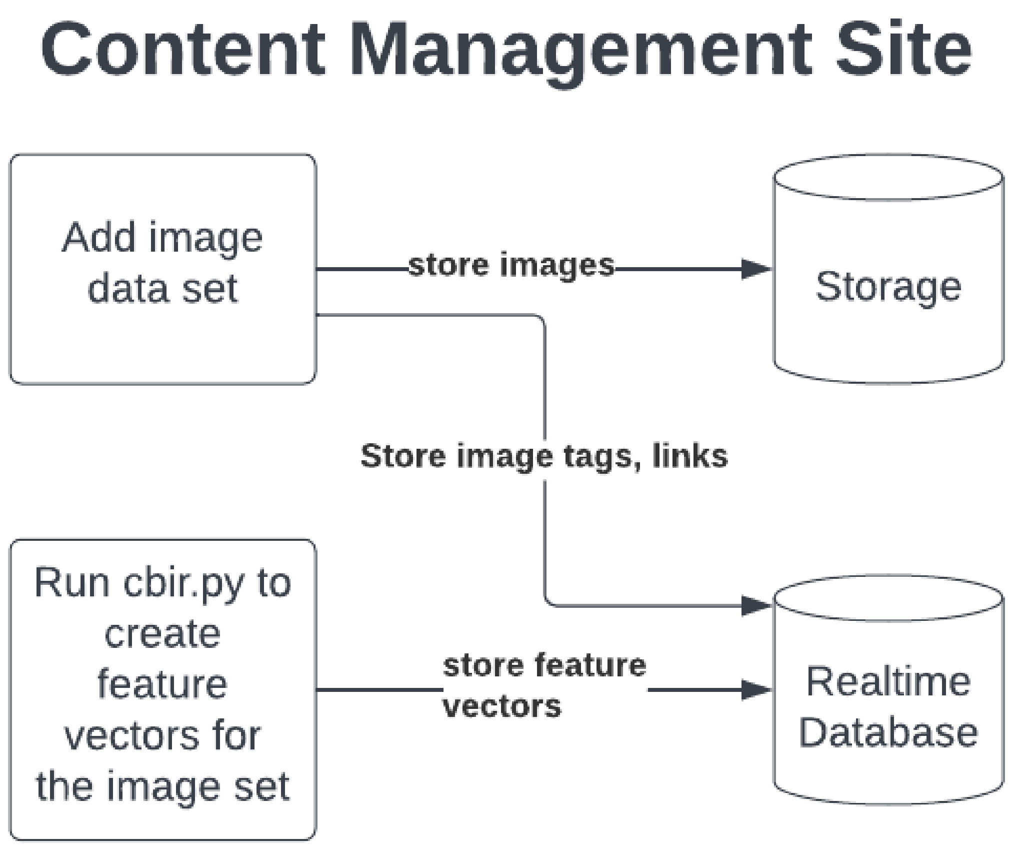
Major features of the project

- Keyword based Content Retrieval
- Image based Content Retrieval
- Content Management Site

Database Preparation

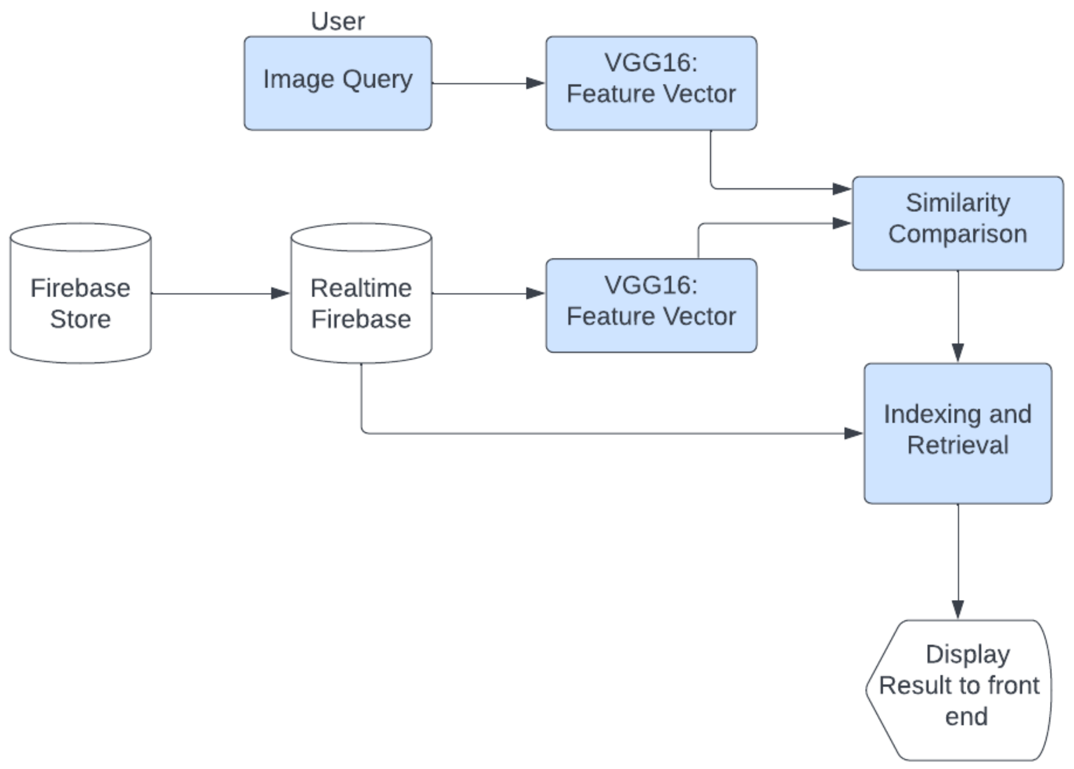
- Content Management Site
  - A separate program
  - Run on its own
- Upload data sets

1. captions for image-tagging
2. Image data set

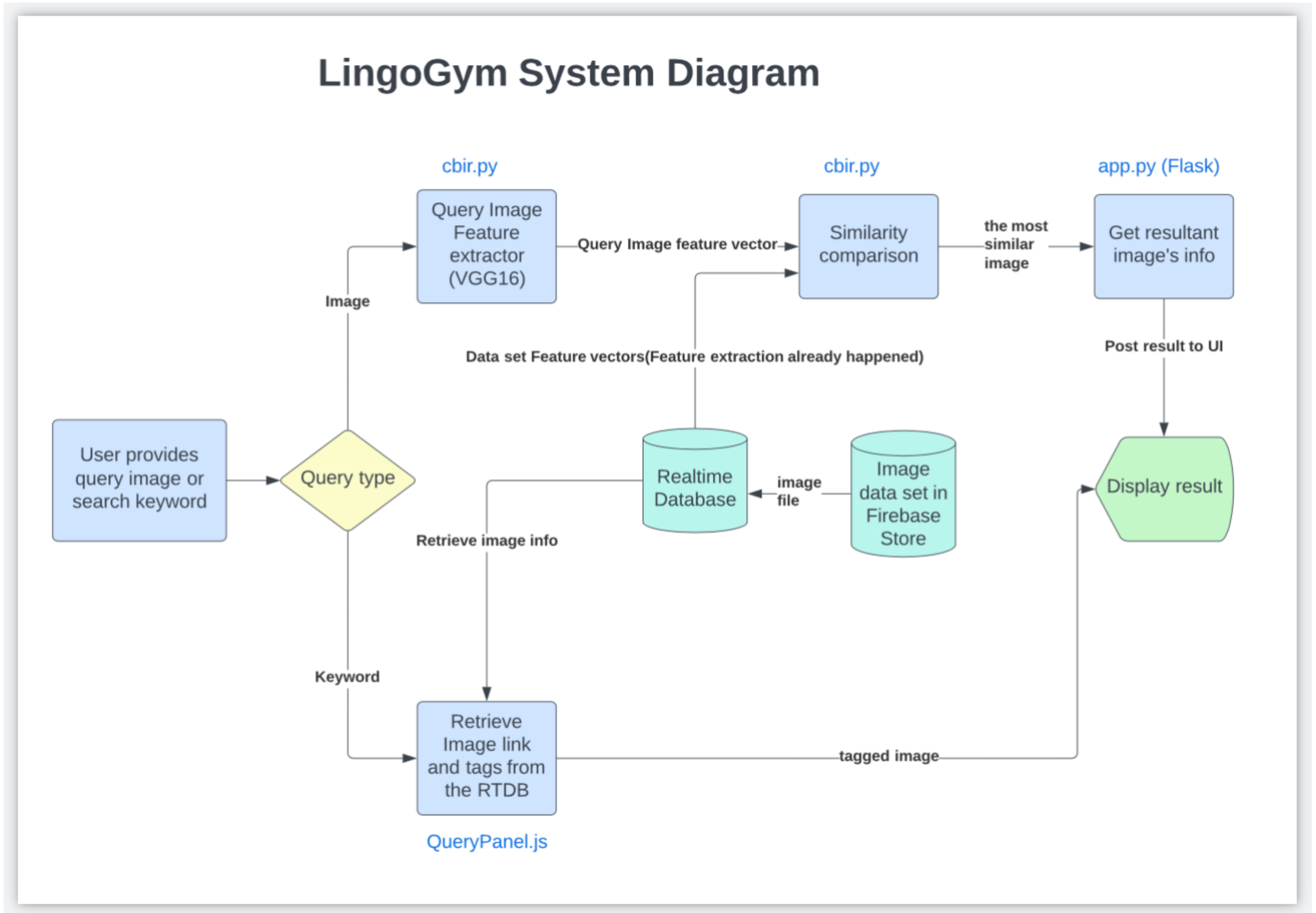


Content-based Image Retrieval

- A technique to query an image over a database of images to get a set of results that most closely matches the queries image.
- VGG 16
    - A pre-trained deep CNN model
- Ease of implementation and model accuracy

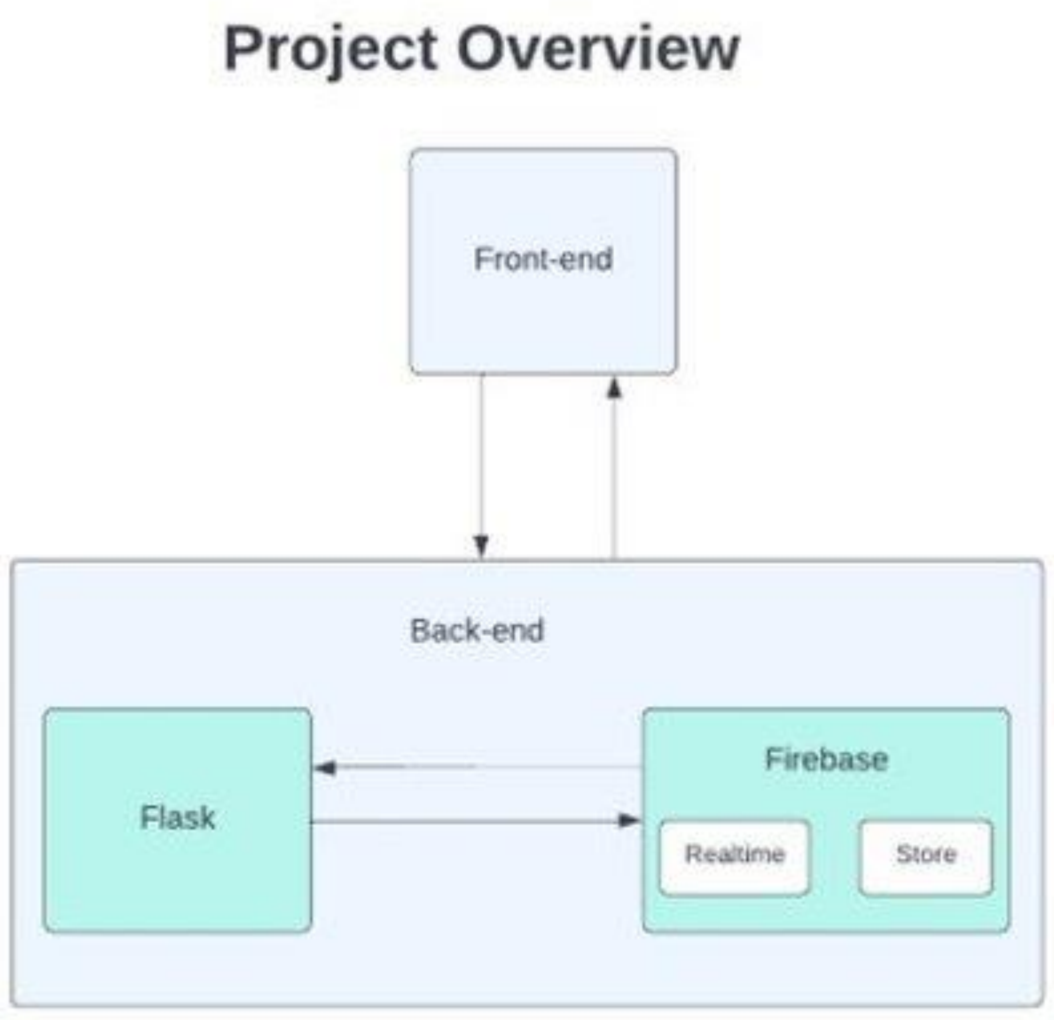


Connecting project components


LingoGym System Diagram
cbir.py
Query Image Feature extractor (VGG16)
Query Image feature vector
cbir.py
Similarity comparison
the most similar image
app.py (Flask)
Get resultant image's info
Image
Data set Feature vectors(Feature extraction already happened)
Post result to UI
User provides query image or search keyword
Query type
Realtime Database
image file
Image data set in Firebase Store
Display result
Retrieve image info
Keyword
Retrieve Image link and tags from the RTDB
tagged image
QueryPanel.js

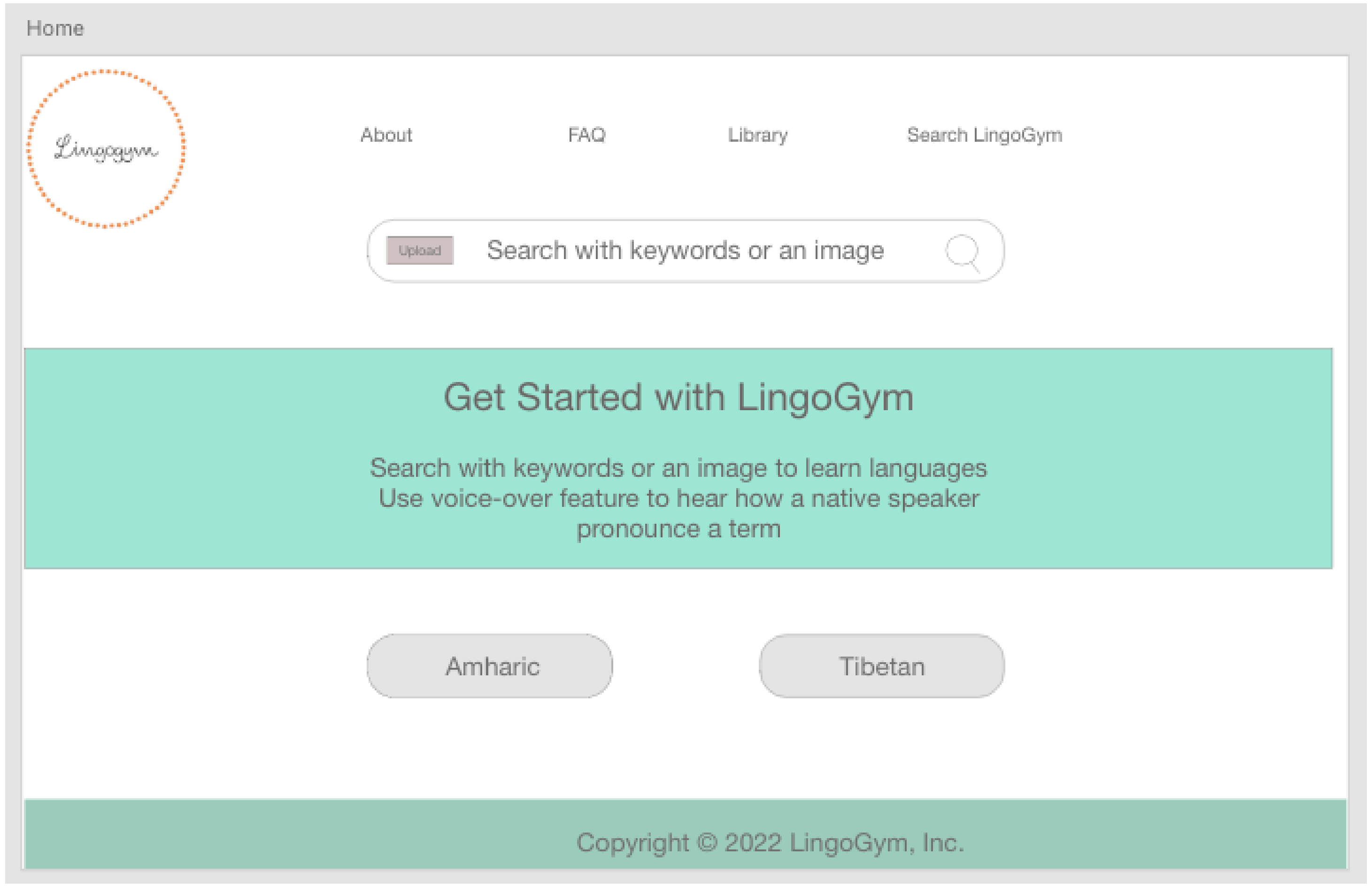
Main Contributions

- Communication between flask and firebase
- Implement CBIR using VGG 16
- Find an efficient way to retrieve similar images
    - Store feature vectors for each image in the data set to realtime database
    - Improve retrieval speed
    - Extract feature once but api-calls needed to access the feature vectors.



Contribution: Front end UI Design

- Adobe XD
- Support from OU IT Software
- Generate links to be shared with teammate
- Easy to use



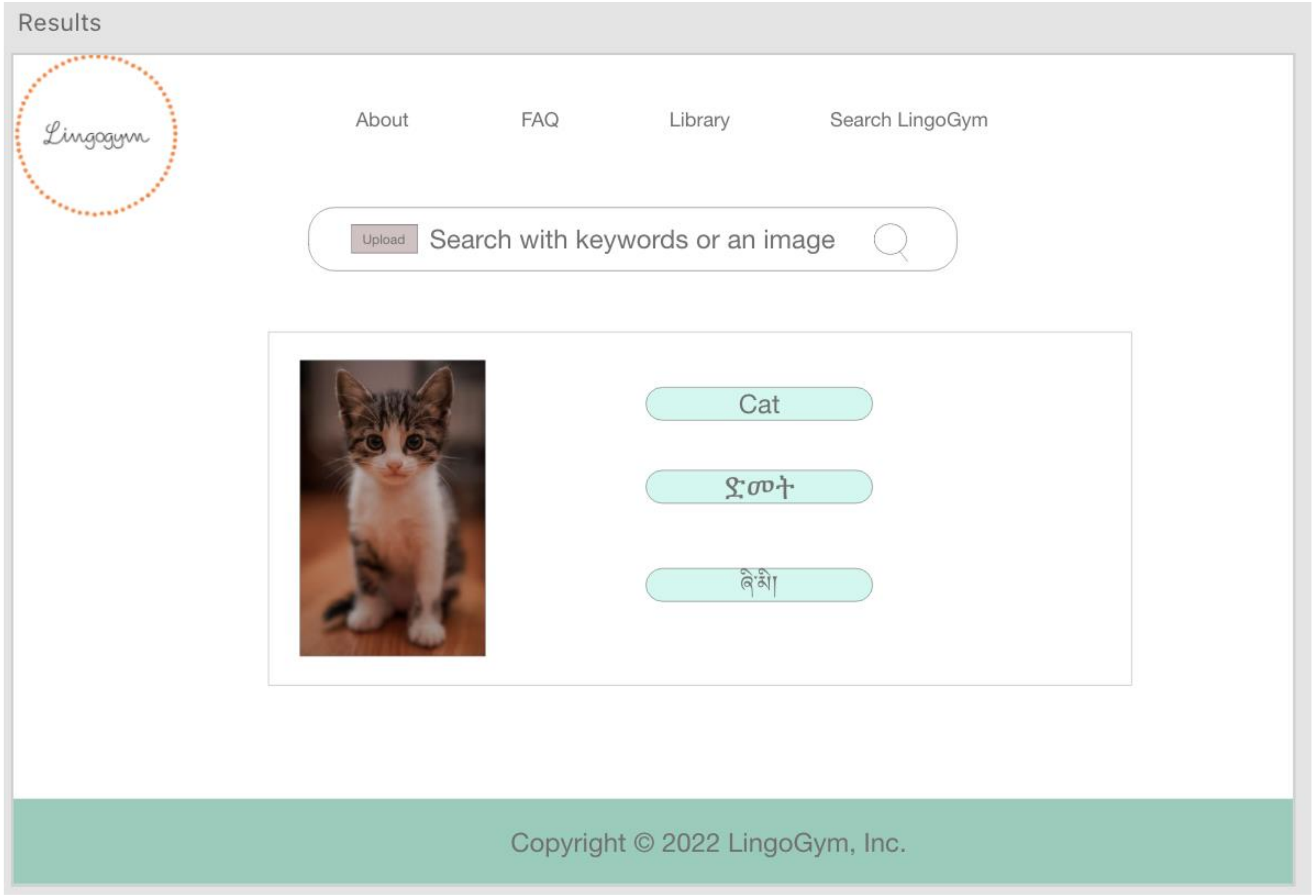
Get Started with LingoGym

Search with keywords or an image to learn languages
Use voice-over feature to hear how a native speaker pronounce a term

Copyright © 2022 LingoGym, Inc.

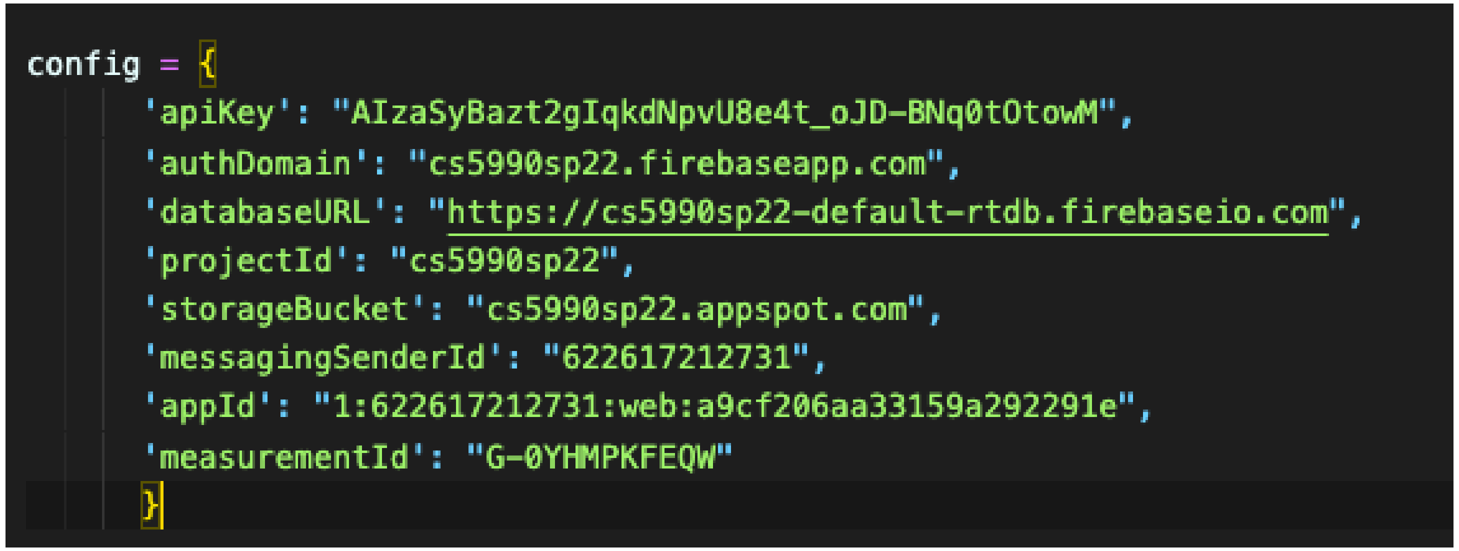
Contribution: Front end UI Design

- UI displaying output


Results
Lingogym
About
FAQ
Library
Search LingoGym
Upload
Search with keywords or an image
Cat
ድመት
ཞི་མི།
Copyright © 2022 LingoGym, Inc.

Communication between Flask and Firebase

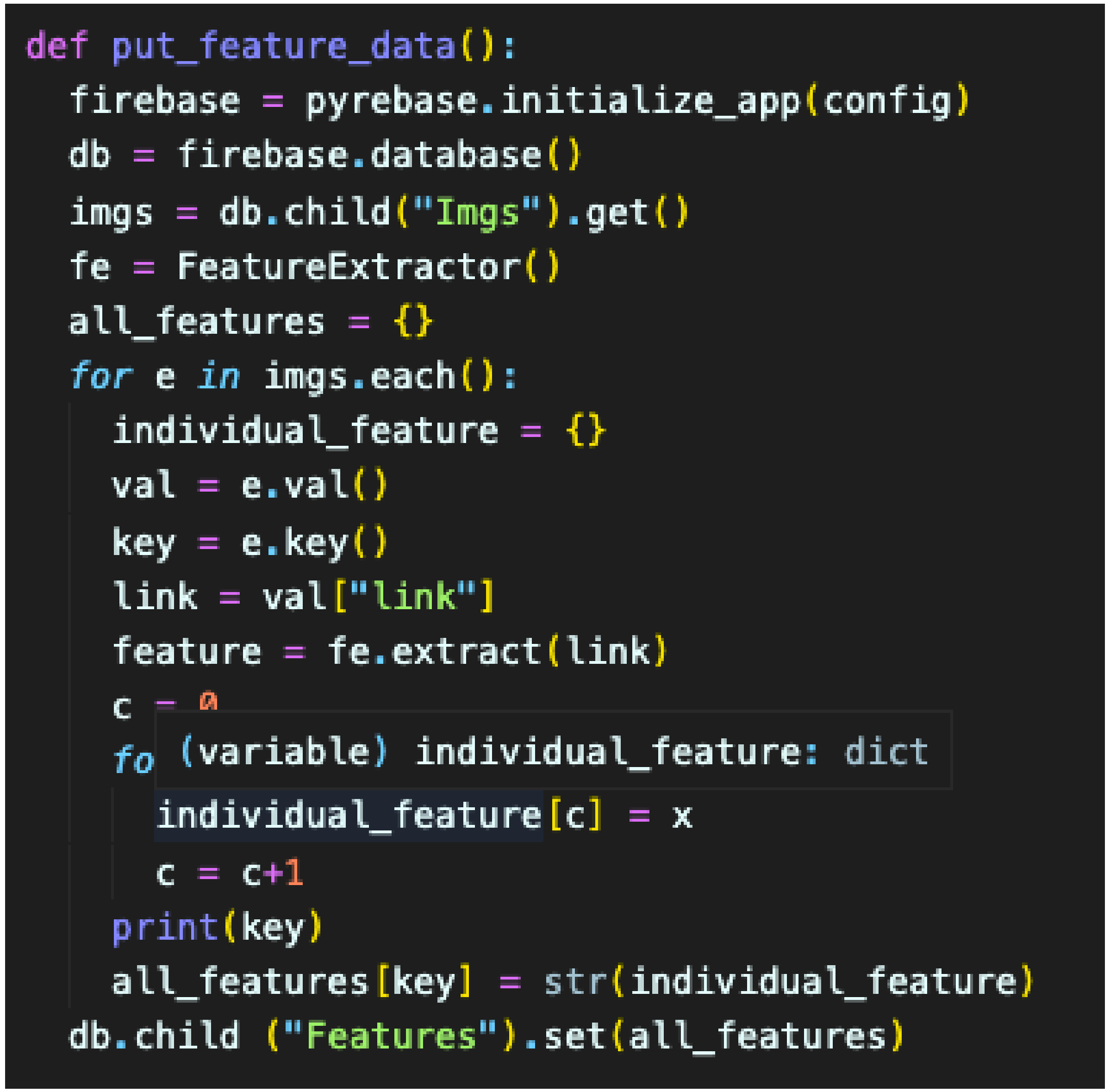
- Pyrebase interface
- Communicate with Firebase's REST API.
- Installation
  - Pipenv install pyrebase



```
config = {
        'apiKey': "AIzaSyBazt2gIqkdNpvU8e4t_oJD-BNq0tOtowM",
        'authDomain': "cs5990sp22.firebaseapp.com",
        'databaseURL': "https://cs5990sp22-default-rtdb.firebaseio.com",
        'projectId': "cs5990sp22",
        'storageBucket': "cs5990sp22.appspot.com",
        'messagingSenderId': "622617212731",
        'appId': "1:622617212731:web:a9cf206aa33159a292291e",
        'measurementId': "G-0YHMPKFEQW"
        }
```

Firebase CRUD Operations using python

Create Feature Vectors for image data set and store in Realtime database

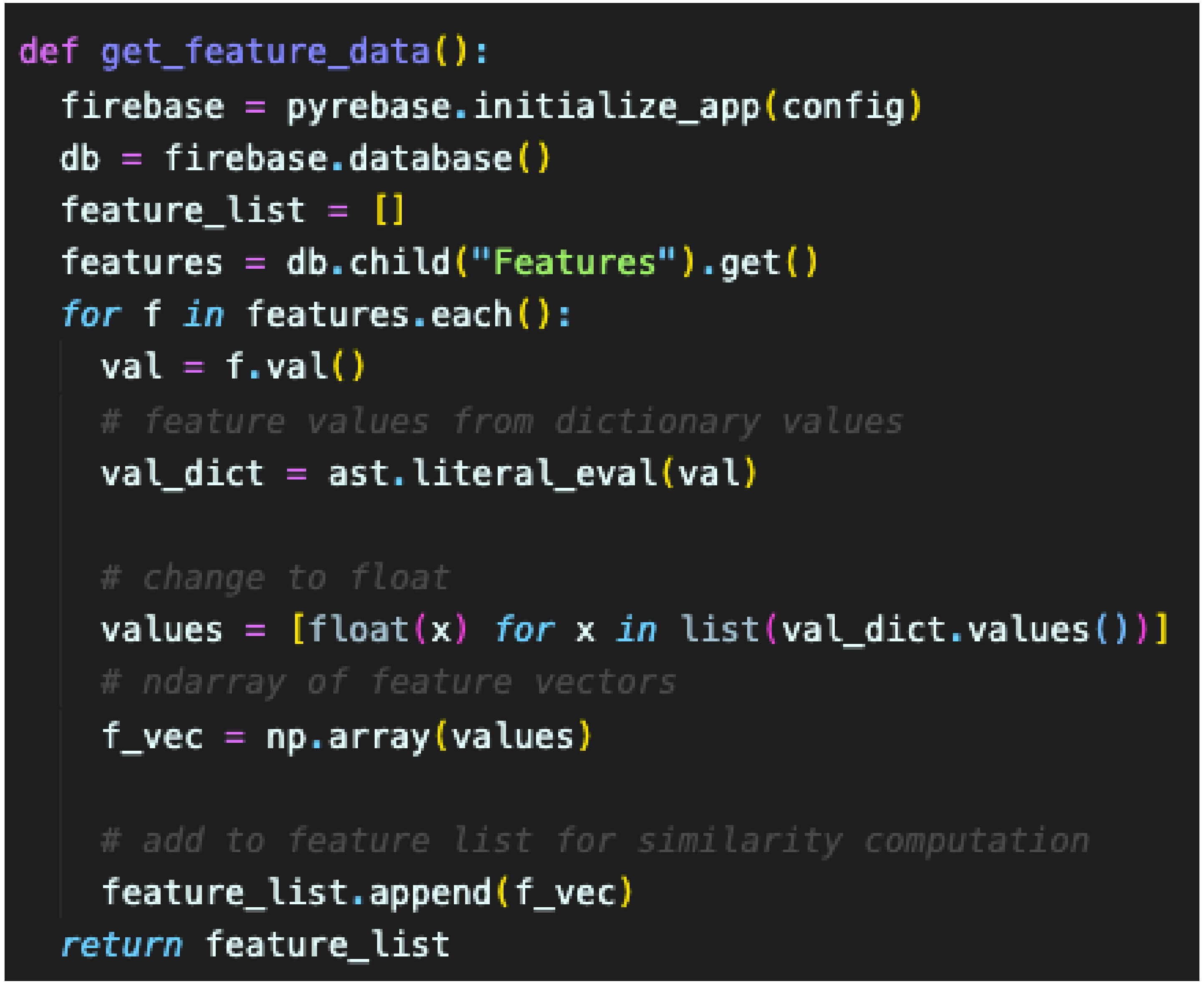


```
def put_feature_data():
  firebase = pyrebase.initialize_app(config)
  db = firebase.database()
  imgs = db.child("Imgs").get()
  fe = FeatureExtractor()
  all_features = {}
  for e in imgs.each():
    individual_feature = {}
    val = e.val()
    key = e.key()
    link = val["link"]
    feature = fe.extract(link)
    c = 0
    fo (variable) individual_feature: dict
      individual_feature[c] = x
      c = c+1
    print(key)
    all_features[key] = str(individual_feature)
  db.child ("Features").set(all_features)
```

Firebase CRUD Operations using python

Read Feature Vectors for image data set stored in Realtime database

For similarity comparison

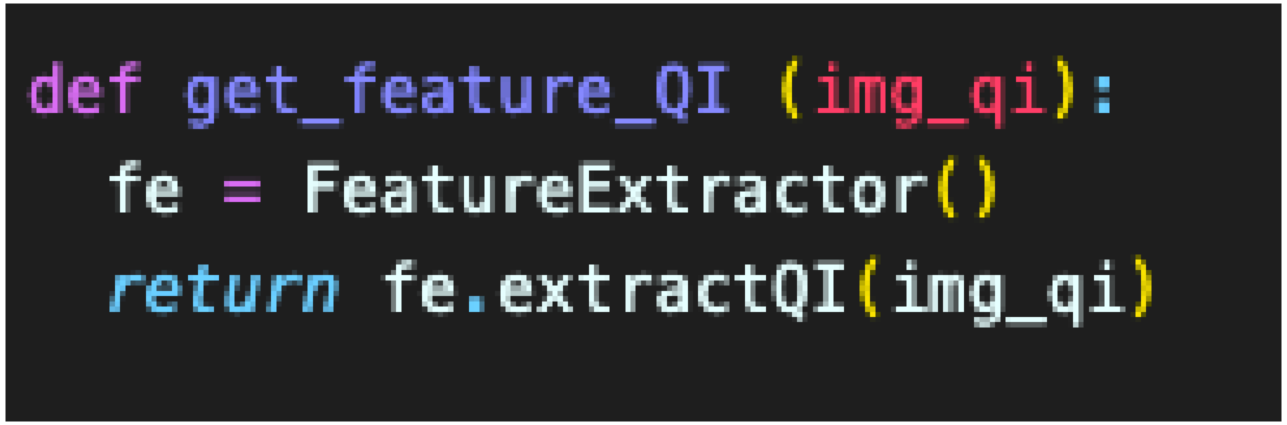


```python
def get_feature_data():
  firebase = pyrebase.initialize_app(config)
  db = firebase.database()
  feature_list = []
  features = db.child("Features").get()
  for f in features.each():
    val = f.val()
    # feature values from dictionary values
    val_dict = ast.literal_eval(val)

    # change to float
    values = [float(x) for x in list(val_dict.values())]
    # ndarray of feature vectors
    f_vec = np.array(values)

    # add to feature list for similarity computation
    feature_list.append(f_vec)
  return feature_list
```

Query Image's Feature Vector

Apply VGG 16 on the query image to extract its feature vector.

For similarity comparison with the feature vectors for images from dataset



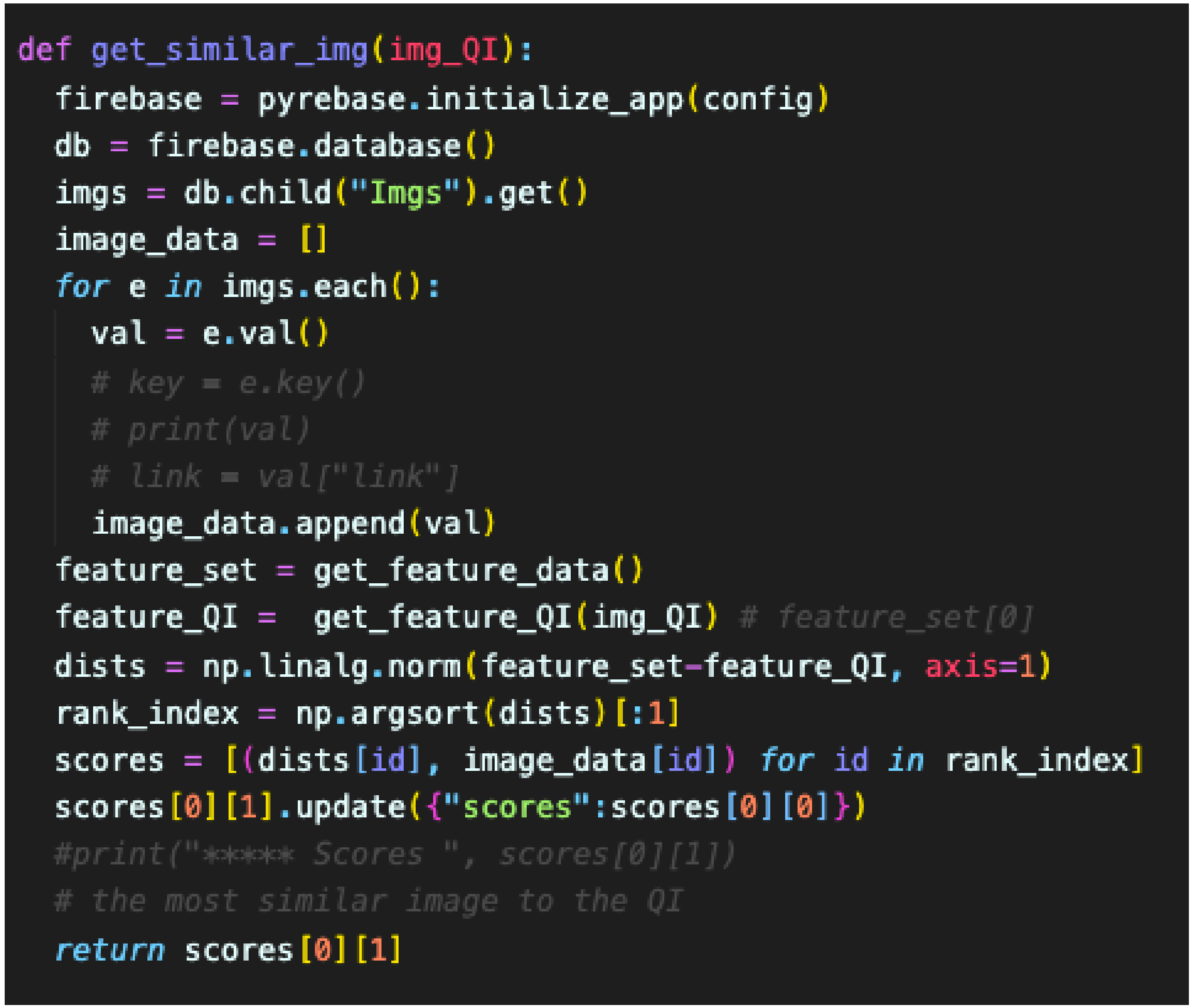
```
def get_feature_QI (img_qi):
  fe = FeatureExtractor()
  return fe.extractQI(img_qi)
```

Similarity Comparison

Subtract query image's feature vector from that of the feature vectors for images in the dataset.

Return information about the most matching feature vector to that of the QI

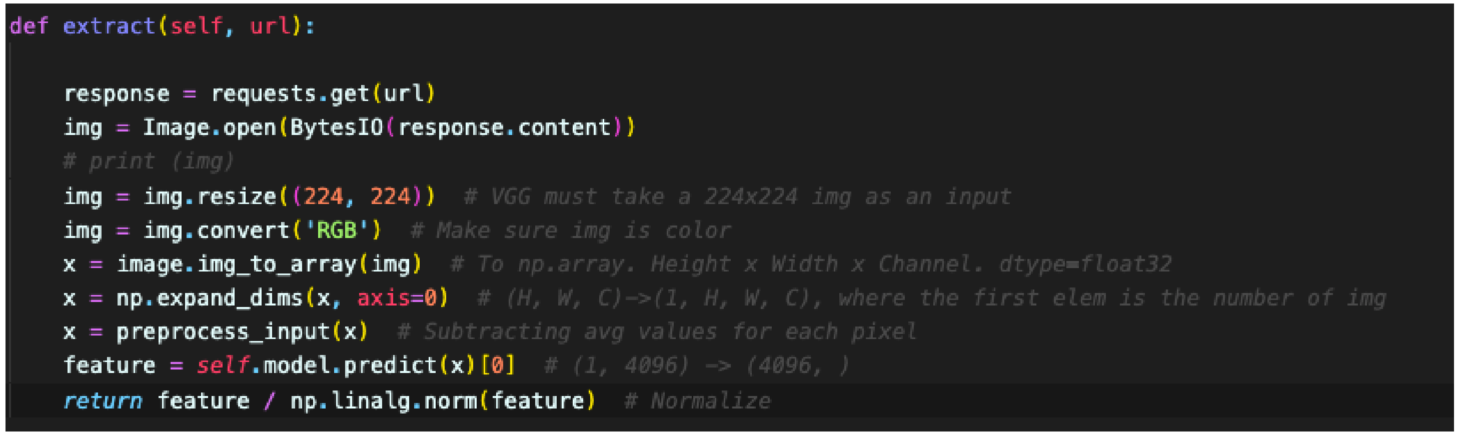
Includes image tags and link



```python
def get_similar_img(img_QI):
  firebase = pyrebase.initialize_app(config)
  db = firebase.database()
  imgs = db.child("Imgs").get()
  image_data = []
  for e in imgs.each():
    val = e.val()
    # key = e.key()
    # print(val)
    # link = val["link"]
    image_data.append(val)
  feature_set = get_feature_data()
  feature_QI =  get_feature_QI(img_QI) # feature_set[0]
  dists = np.linalg.norm(feature_set-feature_QI, axis=1)
  rank_index = np.argsort(dists)[:1]
  scores = [(dists[id], image_data[id]) for id in rank_index]
  scores[0][1].update({"scores":scores[0][0]})
  #print("***** Scores ", scores[0][1])
  # the most similar image to the QI
  return scores[0][1]
```

CBIR Implementation: VGG 16

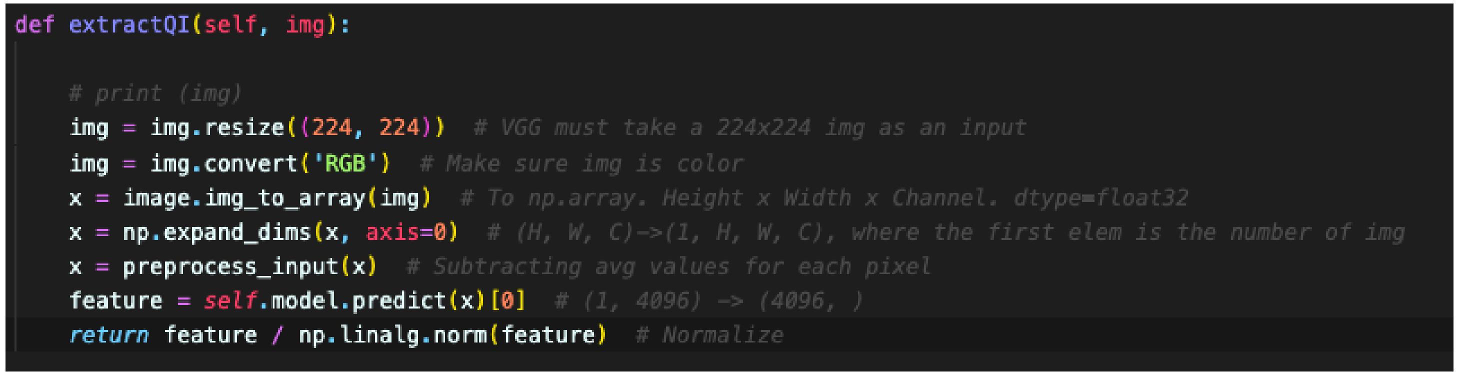
- Feature Extractor method for the image dataset



```
def extract(self, url):

    response = requests.get(url)
    img = Image.open(BytesIO(response.content))
    # print (img)
    img = img.resize((224, 224))  # VGG must take a 224x224 img as an input
    img = img.convert('RGB')  # Make sure img is color
    x = image.img_to_array(img)  # To np.array. Height x Width x Channel. dtype=float32
    x = np.expand_dims(x, axis=0)  # (H, W, C)->(1, H, W, C), where the first elem is the number of img
    x = preprocess_input(x)  # Subtracting avg values for each pixel
    feature = self.model.predict(x)[0]  # (1, 4096) -> (4096, )
    return feature / np.linalg.norm(feature)  # Normalize
```

CBIR Implementation: VGG 16

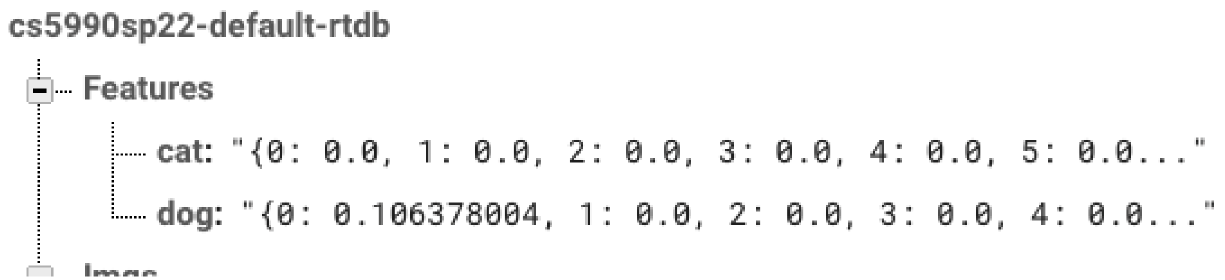
- Feature Extractor method for the query image



```python
def extractQI(self, img):

    # print (img)
    img = img.resize((224, 224))  # VGG must take a 224x224 img as an input
    img = img.convert('RGB')  # Make sure img is color
    x = image.img_to_array(img)  # To np.array. Height x Width x Channel. dtype=float32
    x = np.expand_dims(x, axis=0)  # (H, W, C)->(1, H, W, C), where the first elem is the number of img
    x = preprocess_input(x)  # Subtracting avg values for each pixel
    feature = self.model.predict(x)[0]  # (1, 4096) -> (4096, )
    return feature / np.linalg.norm(feature)  # Normalize
```

Issues faced

- How to store feature vectors in Realtime database?
  - Firebase: no native support for arrays
  - Data stored as key-value pairs
- Solution
  - Key-value pairs created for each feature vector
  - Index as key, element as value
  - Feature vectors successfully stored in Firebase
- Requirements
  - Read feature vectors: make API call to firebase
  - Similarity comparison: convert feature vectors to NumPy array first

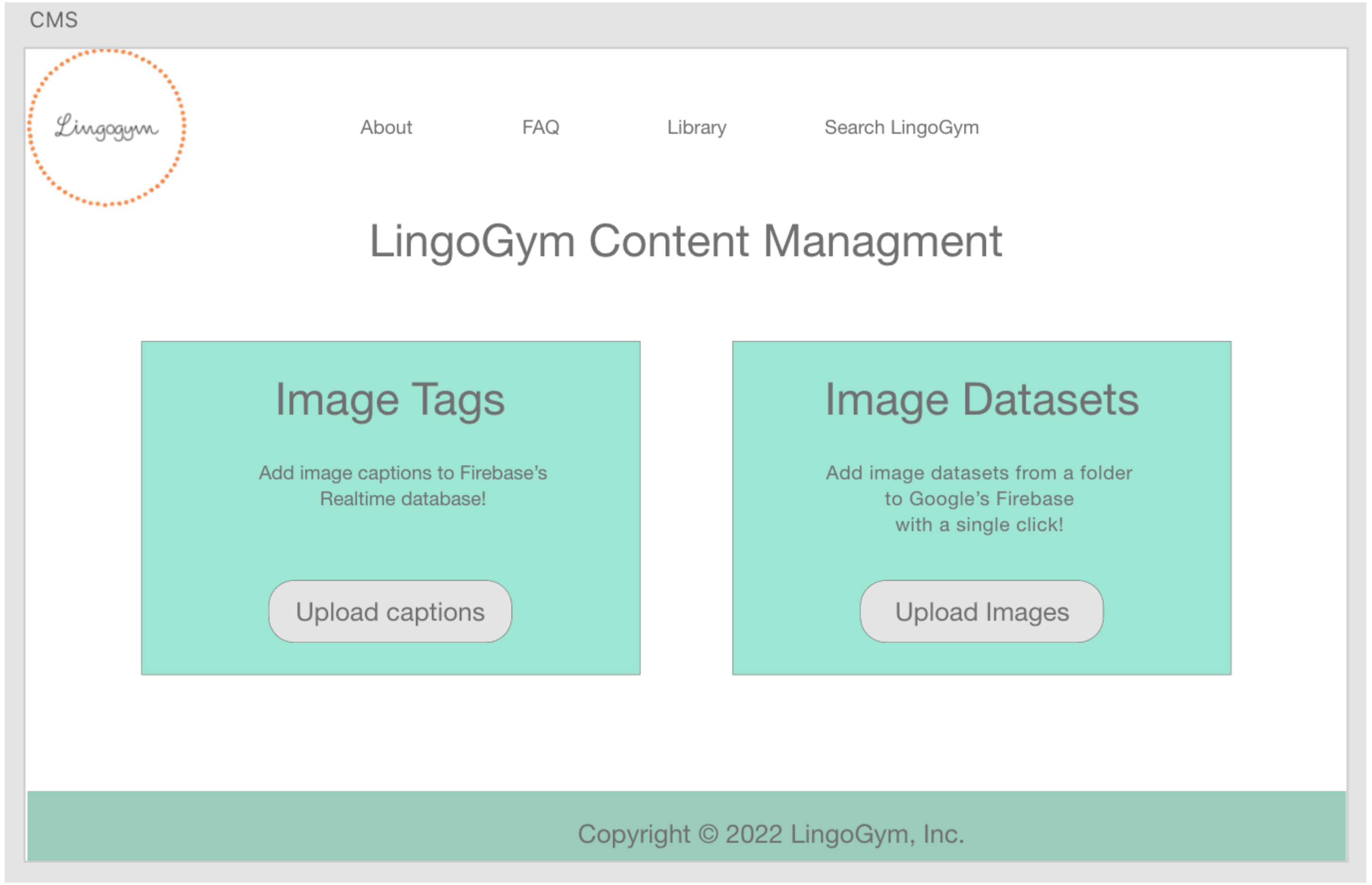


```
cs5990sp22-default-rtdb
  Features
    cat: "{0: 0.0, 1: 0.0, 2: 0.0, 3: 0.0, 4: 0.0, 5: 0.0..."
    dog: "{0: 0.106378004, 1: 0.0, 2: 0.0, 3: 0.0, 4: 0.0..."
```

Demo

Conclusion and Future Work

- Per project objective
  - Have been able to create a web application
  - Retrieve similar images
  - Multilingual search
- Future Work
  - Voice over feature not integrated yet
  - Increase image dataset to cover more diverse and advanced topics
  - Explore other CBIR techniques
  - Integrate more languages
  - Implementation of CMS UI



LingoGym Content Managment

Image Tags

Add image captions to Firebase's Realtime database!

Image Datasets

Add image datasets from a folder to Google's Firebase with a single click!

Copyright © 2022 LingoGym, Inc.

References

- hoge528. (2020, June 19). *[CVPR20 tutorial] live-coding demo to implement an image search engine from scratch*. YouTube. Retrieved April 11, 2022, from https://www.youtube.com/watch?v=M0Y9_vBmYXU
- YouTube. (2020, September 29). *Fullstack flask react tutorial - master Flask Basics and build a python flask react app | 2020 HD.* YouTube. Retrieved April 11, 2022, from https://www.youtube.com/watch?v=cb1vy1HpVwk
- *Discovering the places in the world where English barely exists*. telc - Discovering the places in the world where English barely exists. (n.d.). Retrieved April 11, 2022, from https://www.telc.net/en/about-telc/news/detail/discovering-the-places-in-the-world-where-english-barely-exists.html
- *Learning English as a second language*. Maryville Online. (2021, March 19). Retrieved April 11, 2022, from https://online.maryville.edu/blog/english-as-a-second-language-resources-for-students/
- *Link.springer.com*. (n.d.). Retrieved April 11, 2022, from https://link.springer.com/content/pdf/10.1007%2F978-1-4842-7185-8.pdf
- Image retrieval in the wild. (n.d.). Retrieved April 11, 2022, from https://matsui528.github.io/cvpr2020_tutorial_retrieval/