MINI PROJECT REPORT

On

FACE CLUSTERING WEB APPLICATION

Submitted in partial fulfillment for the completion of

B.E. VI Semester

COMPUTER SCIENCE AND ENGINEERING

By

HARKANT MAGOTRA	**(1005-20-733042)**
BAIRA SAI VIKAS	**(1005-20-733081)**
MADHAGOUNI VARUN TEJA GOUD	**(1005-20-733091)**

Under the guidance of

Dr. V.B.Narasimha
Assistant Professor,
Dept. of CSE, UCEOU.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

UNIVERSITY COLLEGE OF ENGINEERING (A)

OSMANIA UNIVERSITY, HYDERABAD - 500007

2022-2023

University College of Engineering Osmania University (A)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

(Osmania University)

AMBERPET, HYDERABAD – 500 007

CERTIFICATE

This is to certify that Mini Project entitled "**FACE CLUSTERING**" submitted to **UNIVERSITY COLLEGE OF ENGINEERING (A), OSMANIA UNIVERSITY,** in partial fulfillment of the requirements for the completion of B.E. VI semester Computer Science and Engineering, during the academic year 2022-2023, is a record of original work done by

HARKANT MAGOTRA	**(1005-20-733042)**
BAIRA SAI VIKAS	**(1005-20-733081)**
MADHAGOUNI VARUN TEJA GOUD	**(1005-20-733091)**

during the period of study in the Department of CSE, UCEOU, HYDERABAD, under my supervision and guidance.

INTERNAL GUIDE — **Head of Dept. CSE**

(Dr. V.B.Narasimha) — **(Prof. P.V. Sudha)**

External Examiners

ACKNOWLEDGEMENT

I would like to express my deep sense of gratitude and whole-hearted thanks to my project guide Dr. V.B.Narasimha, Assistant Professor, Department of Computer Science and Engineering, University College of Engineering Osmania University, for giving me the privilege of working under his guidance, with tremendous support and cogent discussion, constructive criticism and encouragement throughout this dissertation work carrying out the project work. Department of Computer Science and Engineering

I also thank Prof. P.V.Sudha, Head Department of Computer Science and Engineering for her support from the Department and allowing all the resources available to us students.

I also extend my thanks to the entire faculty of the University College of Engineering, Osmania University who encourages us throughout the course of our Bachelor degree and allowing us to use the many resources present in the department.

Our sincere thanks to our parents and friends for their valuable suggestions, morals, strength and support for the completion of our project.

ABSTRACT

Face clustering is a fundamental task in computer vision, aiming to group similar faces together in each dataset. This single-page abstract presents an overview of a face clustering system designed to enhance facial recognition and enable effective grouping of individuals based on their facial features. The proposed system leverages deep learning techniques for feature extraction, using Convolutional Neural Networks (CNNs) to obtain discriminative facial representations. A suitable clustering algorithm is then applied to categorize faces into distinct groups based on their visual similarities Data collection involves assembling a diverse dataset of facial images, which is preprocessed for face detection, alignment, and normalization to ensure consistency in the features. The clustering algorithm's performance is evaluated using metrics such as silhouette score and completeness to assess the accuracy and quality of the clustering results.
To enhance user-friendliness, a graphical user interface (GUI) is developed to allow users to input their datasets and visualize the clustered results. The system's potential applications include social media analysis, security surveillance, and image organization.

In conclusion, this face clustering system offers an efficient and accurate approach to group similar faces, empowering various domains with improved facial recognition and image management capabilities. Its contributions demonstrate the potential impact of computer vision in real-world scenarios, paving the way for further research and advancements in face clustering technologies.

TABLE OF CONTENTS

1. INTRODUCTION

For the project we propose to build a Face clustering web app for the digital age, we capture countless photographs of ourselves and our loved ones, leading to vast collections of images that can become challenging to manage and organize. Recognizing the need for an efficient solution, we present a groundbreaking Face Clustering web App, a cutting-edge application that revolutionizes the way we interact with our photo libraries.

At its core, the app harnesses the power of computer vision and machine learning to automatically group similar faces together, streamlining the process of organizing and navigating through extensive image collections. By intelligently clustering faces based on their visual features, the app offers users a seamless and visually appealing experience, allowing them to find specific individuals ease.

The App has following main components:

1.Taking pictures from the user.

2.Applying machine learning algorithms on the user images.

3.Clustering and displaying clustered images to the user.

The Web App is implemented using python and Vue js and its powerful built packages like scikit-learn which helps in clustering images based on faces identified in the picture and face recognition which helps in detecting faces in an image and Axois used to act as a link between front-end and back-end for communication.

2. AIMS AND OBJECTIVES:

The aim of the “FACE CLUSTERING WEB APPLICATION” is to develop an intelligent system capable of automatically grouping similar faces from an input dataset into distinct clusters. This system will leverage computer vision algorithms and machine learning techniques to achieve accurate and efficient face clustering.

Objectives are:

Data Collection and Preprocessing:

- Gather a diverse dataset of facial images, ensuring variability in lighting conditions, facial expressions, poses, and demographic

User Interface (UI):

- Design and implement a user-friendly interface for users to interact with the face clustering system.
- Allow users to upload their own images for clustering

Testing and Deployment:

- Implement and evaluate DBSCAN clustering algorithm to group the facial feature vectors into distinct clusters based on their similarity.

By achieving these objectives, the face clustering Web project will deliver a powerful tool capable of effectively organizing and grouping facial images.

3. METHODOLOGY

To have the completion of the proposed work, a proper methodology is supposed to be followed. Firstly, the dataset needs to be fetched and cleaned which can then be used for training purposes.

The study made use of face recognition and scikit learn module from python for face detection and clustering and saving them to SQLite database. Front end made use of Vue js to dynamically adapt to user interactions and medium to process the requests of users. Axios is used as a middleware to get the data from the backend and show them to the user.

3.1 DATASET COLLECTION

Dataset for our project are pictures. We chose a wide range of pictures which consisted of group photos, solo photos and persons with different race, sex etc for testing our ML model under different circumstances

3.2 FACE DETECTION AND ENCODING

Face Locations:

Once images are uploaded we standardize Images, Images in our dataset can be of any encoding so, we convert them into RGB color grading.

Now we detect faces (If any) present in our image and then the function returns a list of tuples, where each tuple represents the coordinates of a detected face's bounding box in the format (top, right, bottom, left). Each tuple corresponds to a single face found in the image.

(top, right): The coordinates of the top-right corner of the bounding box.

(bottom, left): The coordinates of the bottom-left corner of the bounding box.

These coordinates can be used to extract the face regions from the original image for further processing, such as face recognition or analysis.

Face Detection:

Once we have our face locations then we pass it as our parameters to our face_encoding function present in the face_recognition module.

The face_encodings function uses a deep neural network (usually a pre-trained model from dlib or a similar library) to extract the 128-dimensional face encoding from the face region.

Face_encodings function returns a list of 128-dimensional face encodings (one for each face in the image).

3.3. DBSCAN CLUSTERING

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is a popular clustering algorithm that is implemented in scikit-learn, a widely-used machine learning library in Python. DBSCAN is particularly useful for clustering data points that are close to each other in high-density regions and separating outliers or noise points.

Here we use a face-encoding 128-bit array as our input to DBSCAN function Minimum number of images of a face are 3 to perform clustering.

3.4 LABELING AND GROUPING

After performing clustering, we extract labels from each group of Images using numpy module. Now we create a dictionary with label id as our key and list of file names as our value.

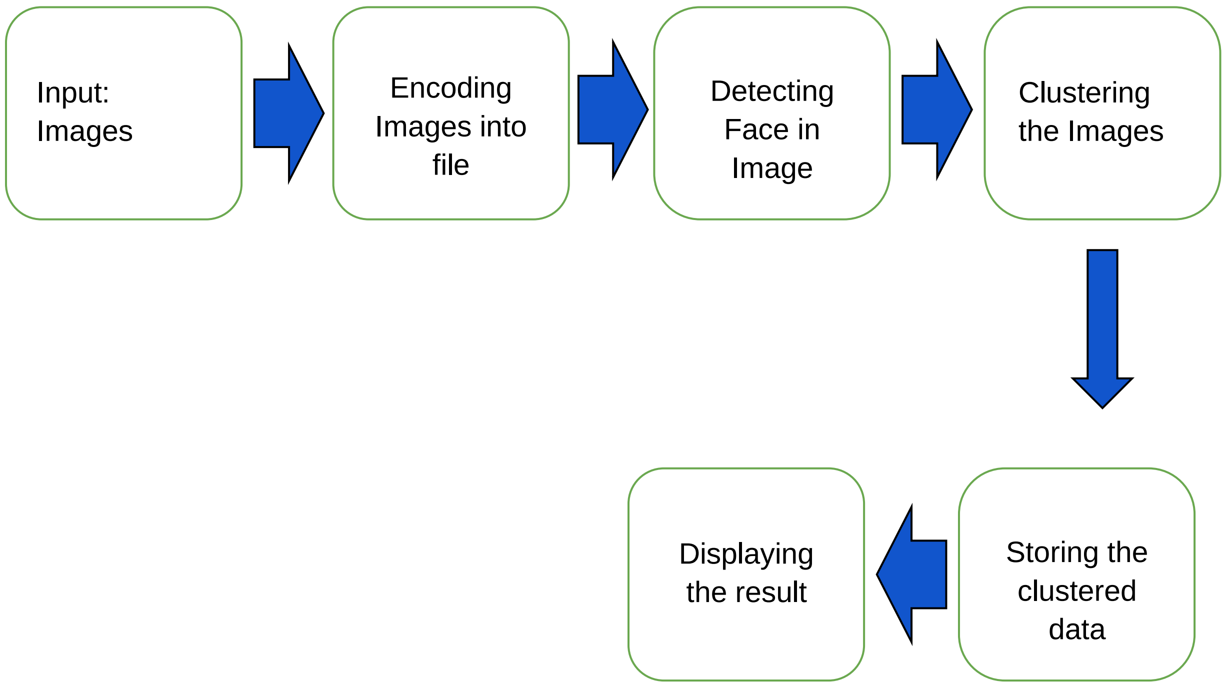
A Image with a unique face will be labeled as -1 therefore, we avoid grouping of images labeled as -1.

3.5 SQLite INTEGRATION

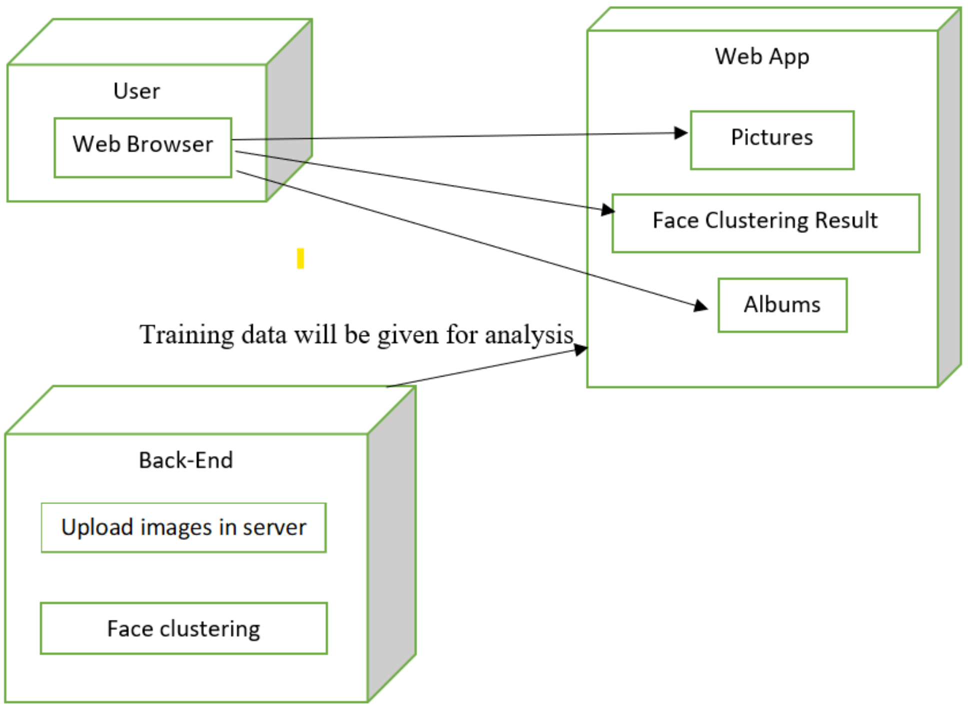
SQLite is a lightweight, serverless, self-contained, open-source relational database management system (RDBMS). Unlike traditional client-server databases, SQLite is embedded directly into the application, making it easy to use and manage.

Once Images are clustered save them into our database with face_id and file name as values.

3.6 DATA FLOW AND SYSTEM DESIGN



Data Flow Diagram of the Proposed Web Application



System Design of the Proposed Web Application

3.7 TECHNOLOGIES USED

BASICS DEFINITIONS AND TERMS

PYTHON:

Python is a high-level, interpreted, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation. Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly procedural), object-oriented and functional programming. It is often described as a "batteries included" language due to its comprehensive standard library. Python is often used as a support language for software developers, for build control and management, testing, and in many other ways. Python is a computer programming language often used to build websites and software, automate tasks, and conduct data analysis. It is used in machine learning, web development, desktop applications, and many other fields.

NumPy :

NumPy, which stands for Numerical Python, is a library consisting of multidimensional array objects and a collection of routines for processing those arrays. Using NumPy, mathematical and logical operations on arrays can be performed. Numpy is a statistical computing library. NumPy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays. It is the fundamental package for scientific computing with Python. NumPy's main object is the homogeneous multidimensional array. We can install the NumPy using the pip tool:

pip install numpy.

OpenCV:

OpenCV is the huge open-source library for computer vision, machine learning, and image processing and now it plays a major role in real-time operation which is very important in today's systems. By using it, one can process images and videos to identify objects, faces, or even handwriting of a human. When integrated with various libraries, such as NumPy, python is capable of processing the OpenCV array structure for analysis. To Identify image pattern and its various features we use vector space and perform mathematical operations on these features.

pip install opencv-python

Flask:

Flask is a web framework; it is a Python module that lets you develop web applications easily. It has a small and easy-to-extend core: it is a microframework that does not include an ORM (Object Relational Manager) or such features. Flask provides easy access to HTTP request data and facilitates handling and processing of incoming requests.

pip install Flask

Vue.Js

Vue.js is an open-source JavaScript framework for building user interfaces. It is designed to be simple, flexible, and scalable. Vue.js is based on the Model-View-ViewModel (MVVM) pattern, and it uses a declarative programming style. Vue.js is a popular framework for building single-page applications (SPAs). It is also used for building web components, which are reusable pieces of code that can be embedded in any HTML page. Vue.js is a powerful and versatile framework that can be used to build a wide variety of web applications. It is easy to learn and use, and it is supported by a large community of developers.

Scikit-learn:

Scikit-learn is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support-vector machines, random forests, gradient boosting, k-means and many more. It also provides tools for model selection, data preprocessing, model evaluation and visualization. Scikit-learn is a popular choice for machine learning tasks in Python because it is easy to use, has a wide range of algorithms and is well-documented. It is also actively maintained and updated with new features and bug fixes.

pip install scikit-learn

SQLite

SQLite is a lightweight, fast, and easy-to-use relational database management system (RDBMS). It is a single-file database, which means that the entire database is stored in a single file. This makes it easy to transport and share SQLite databases. SQLite is a popular choice for embedded databases, which are databases that are embedded in a larger application. This is because SQLite is small and lightweight, and it does not require any special configuration or administration. SQLite is also a popular choice for mobile development. This is because SQLite is available on all major mobile platforms, and it is easy to use with the native development tools.

4. IMPLEMENTATION OF PROJECT

4.1 DBSCAN algorithm

DBSCAN is a density-based spatial clustering of applications with noise. This can detect arbitrary shaped clusters as well as clusters with noise (i.e. outliers).

The DBSCAN algorithm works on the principle that a point belongs to a cluster whether it is close to many other points from such a cluster.

DBSCAN has two important variables:

• **eps**: The distance between neighborhoods that defines them. If the distance between two points is below or similar to eps, they are termed neighbors.

• **minPts:** The minimal number of the data points required to define a cluster is minPts. Points are classified as core, boundary, or outlier two of the following two parameters:

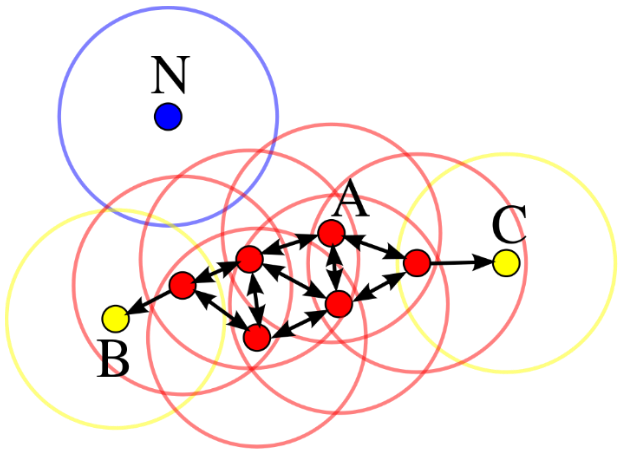
• **Core point:** If there are minimum minPts number of points (along with the point itself) in its neighbourhood region with radius eps, it is a core point.

• **Border point:** A point would be a border point if it can be reached from a core point and also the number of points in its immediate vicinity is fewer than minPts.

Outlier: If a point isn't a core point and can't be reached from any other core points, it's called an outlier.

The following illustrations may help to clarify these topics.

The image below is sourced from Wikipedia.



The value of minPts is 4 in this example. Because there are at least four points of radius eps within its surrounding(neighborhood) region, red points are considered core points. The circles there in figure depict this area. Because they may be reached from a core point and have fewer than four points in their neighborhood, the yellow points are considered very border points. The term "reachable" refers to points that are within a certain radius of a central point. Within the neighborhood of points B and C, there are two points (along with the point including itself) (the surrounding area with which a radius of eps). Finally, since N is not a core point and cannot be visited from one, it is an outlier

How the algorithm works:

- Initially, minPts and eps are determined.
- A initial point will be chosen at random from its surrounding area using radius eps. If there should be at least minPts number of points in the neighborhood, that point is designated as a core point, and a cluster formation begins with it. If this is not the case, the point is labelled as noise. When the development of clusters (let's call it cluster A) begins, every point in the vicinity of the starting point becomes a member of cluster A. If the newly added points are already core points, the points in their immediate vicinity are likewise added to the cluster A.

- A point that has been labelled as noise can be revisited and become part of a cluster.

- The next step in this process is to select another point at random among those that were not visited there in previous steps. Then follow the same instructions as before
- When all the points have been covered, the process is finished.
- A distance measuring approach, like that used in the k-means algorithm, is used to determine the distance between each pair of points. For this, the Euclidean distance approach is often utilized.

DBSCAN method can obtain high density areas and differentiate them over low density regions utilizing these steps

- A cluster is made up of neighbors (i.e., points that may be reached from one another) and the boundary points of these neighbors. The presence of at least one core point is required for the formation of a cluster. We will have a cluster of only single core point and all its boundary points, which is extremely unusual.

DBSCAN's Advantages and Disadvantages:

DBSCAN’s advantages :

- It is not necessary to mention the number of clusters that have formed.
- It works well with clusters of arbitrary shapes.
- DBSCAN is resistant to outliers and will also be able to identify them.

DBSCAN’s disadvantages :

- In some cases, calculating a suitable neighborhood(eps) distance is difficult and may need domain expertise.

DBSCAN is not well suited for determining clusters when in-cluster densities are substantially varied. A combination of eps-minPts variables can be used to specify the clusters' properties. We can't generalize successfully to clusters with significantly varied densities because we only provide
the algorithm one eps-minPts combination.

4.2 APPLYING DBSCAN TO FACE CLUSTERING

We use DBSCAN algorithm to cluster faces.DBSCAN algorithm works as follows:

1. Randomly select a data point that has not been visited.
2. If the selected point is a core point, expand its neighborhood to find all directly or indirectly reachable points.
3. If the neighborhood has at least MinPts points, form a cluster and mark all the points within the neighborhood as members of that cluster.
4. If the selected point is not a core point, mark it as noise.
5. Repeat steps 1 to 4 until all data points have been visited.

```
os.mkdir('faces')

data = None

with open(enc, 'rb') as file:

    data = pickle.load(file)

data_arr = np.array(data)

encodings_arr = [item["encoding"] for item in data_arr]

cluster = DBSCAN(min_samples=3)

cluster.fit(encodings_arr)

labelIDs = np.unique(cluster.labels_)

lbl = cluster.fit_predict(encodings_arr)

silhouette_vals = silhouette_samples(encodings_arr, lbl)

silhouette_avg = np.mean(silhouette_vals)

print("The average Silhouette Score is:", silhouette_avg)

numUniqueFaces = len(np.where(labelIDs > -1)[0])

cnt = 0

clu = {}

for labelID in labelIDs:

    idxs = np.where(cluster.labels_ == labelID)[0]
```

```
            t = []

            x,y,w,h=0,0,0,0

            f_idx=-1

            f=True

            img = None

            for i in idxs:

                if(labelID != -1):

                    if(f):

                        f=False

                        x,y,w,h=data_arr[i]['loc']

                        f_idx = i

                        img = cv2.imread(data_arr[i]['imagePath']).copy()

                    t.append(data_arr[i]["imagePath"])

            if(labelID != -1):

                cnt += 1

                clu["face"+str(cnt)] = []

                for tp in t:

                    s = tp.split('/')

                    clu['face'+str(cnt)].append(s[1])

                image =
face_recognition.load_image_file(data_arr[f_idx]['imagePath'])

                img = Image.fromarray(image, 'RGB').copy()

                img_cropped = img.crop((

                    h,x,y,w

                ))

                img_cropped.save(f'faces/face{str(cnt)}.jpeg')
```

```
        con = sqlite3.connect(db_file)

        cur = con.cursor(

        cur.execute("delete from g_to_clu_rel")

        cur.execute("delete from clu_table")

        for face in clu.keys():

            cur.execute('insert into clu_table values(?,?)', (face, face))

        con.commit()

        for face in clu.keys():

            for pic in clu[face]:

                cur.execute('insert into g_to_clu_rel (clu_id,img_id)
values (?,?)', (face, pic))

        con.commit()

        con.close()

        make_id()
```

4.3 MODULES OF THE PROJECT

The proposed Web Application contains five web pages as below:

Web Page for All photos:

This page provides User Interface for users to view all the images that they have uploaded. Users will be able to view the image in full screen mode and will be able to delete multiple photos at a time from the photo gallery at a time.

This is the landing page for our web app where users can even upload their photos from local devices.

Web Page for Face Clustering result:

This page allows users to view images filtered based on faces. Photos are filtered based on faces by machine learning algorithm used in our back-end(Python).Our page sends a HTTP get request to backend then using flask framework we send it to our front end.

Web Page for Albums viewing:

This page provides User Interface for users to view all the images that they have uploaded in that particular Album. Users will be able to view the image in full screen mode and will be able to delete multiple photos at a time from that particular album at a time.

Web Page for creating Albums:

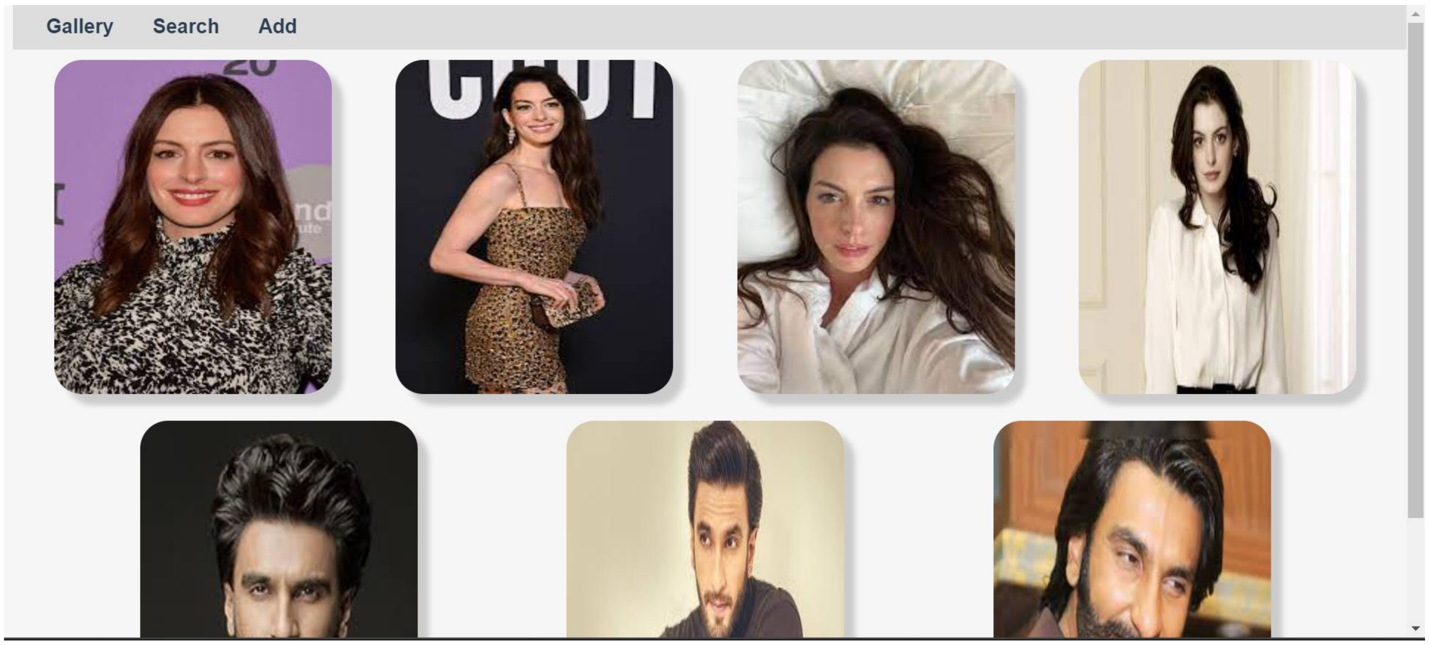
This page provides a user interface so that users can group multiple photos from existing photo gallery and create their album with their custom name, after user clicks create then the data is sent to back end which creates an Album for the user.

This page provides User Interface for user to view all the images that they have uploaded. Users will be able to view the image in full screen mode and will be able to delete multiple photos at a time from the photo gallery at a time.

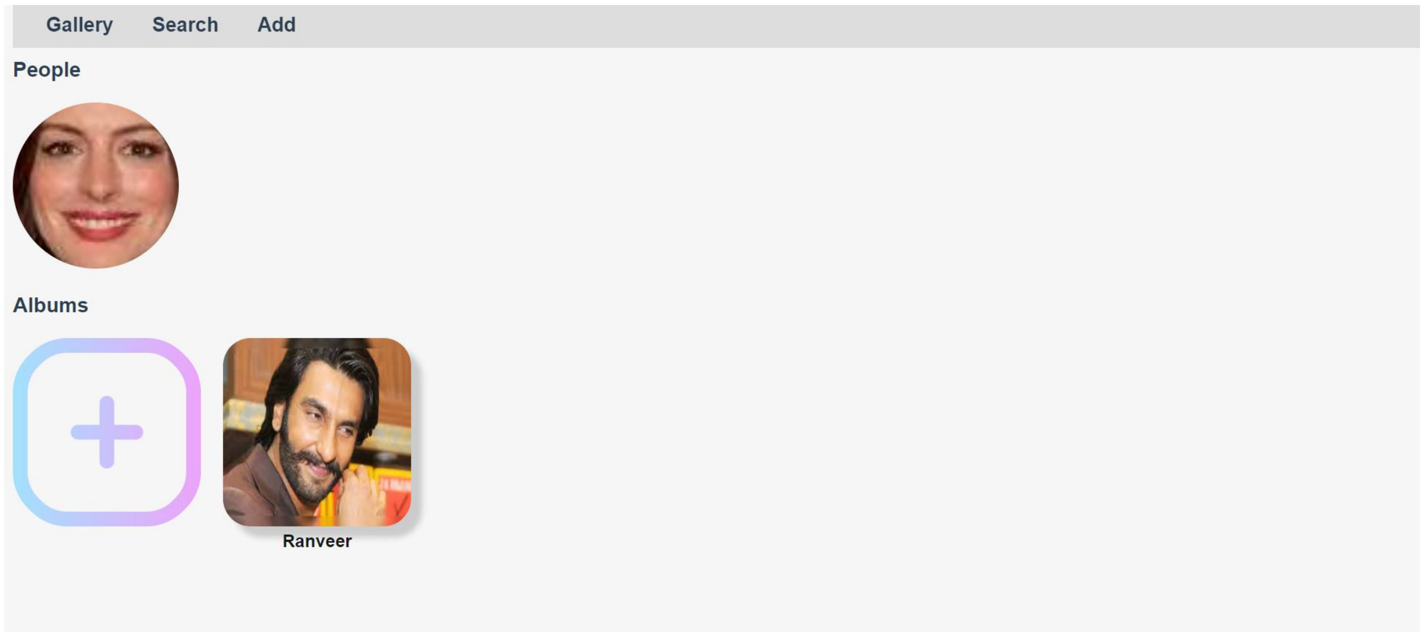
Web Page for Displaying face clusters and Albums:

This page provides User Interface for users to view all the Face clusters that are identified and Albums that are created and provides an option to create an Album.

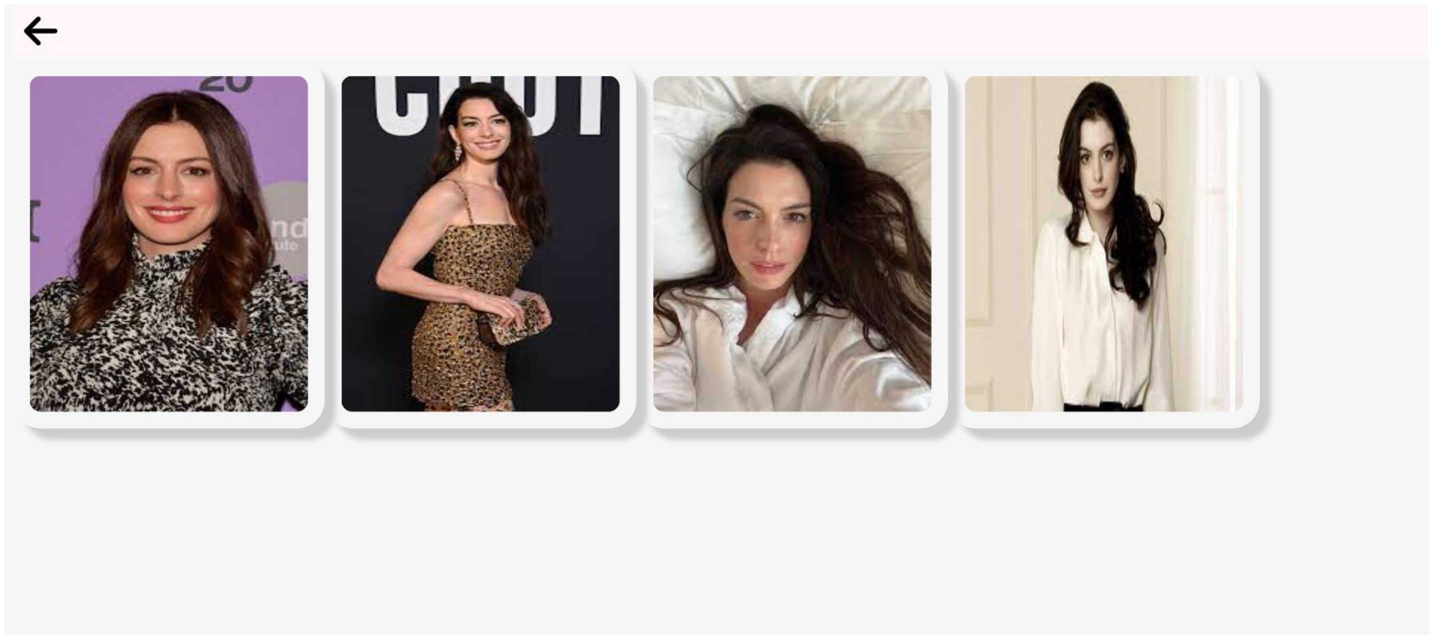
4.4 RESULTS



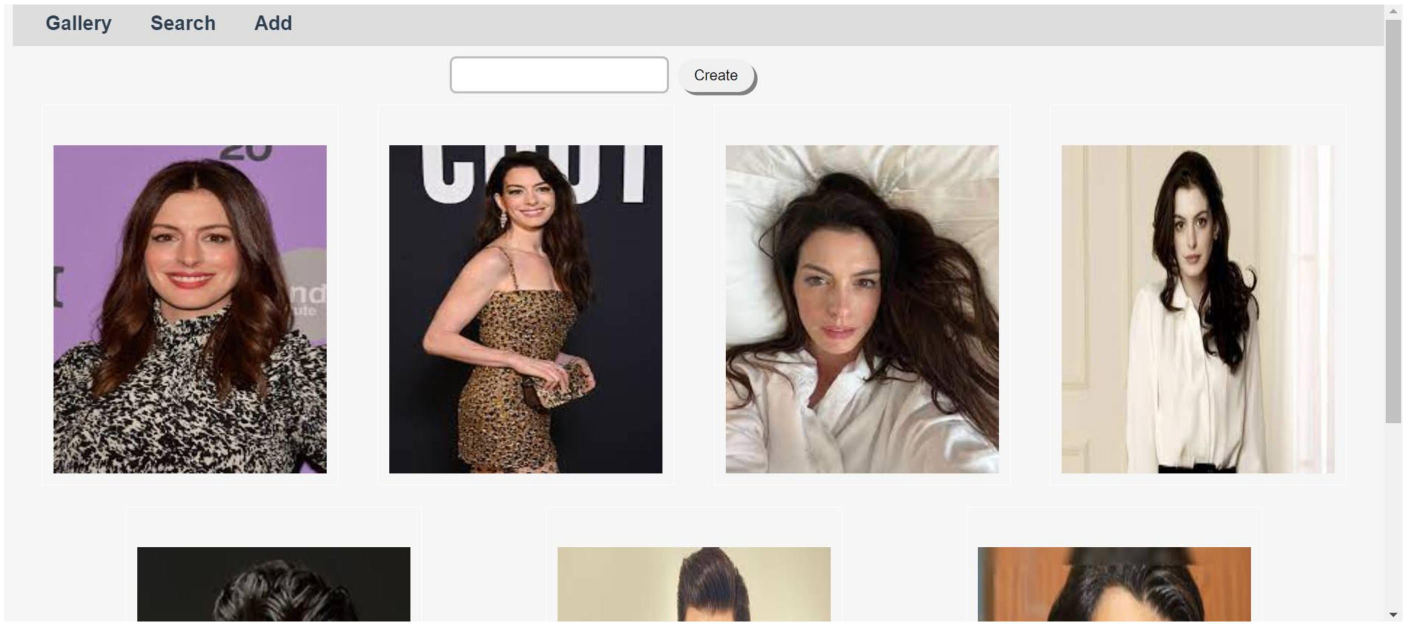
Home Page for Web Application



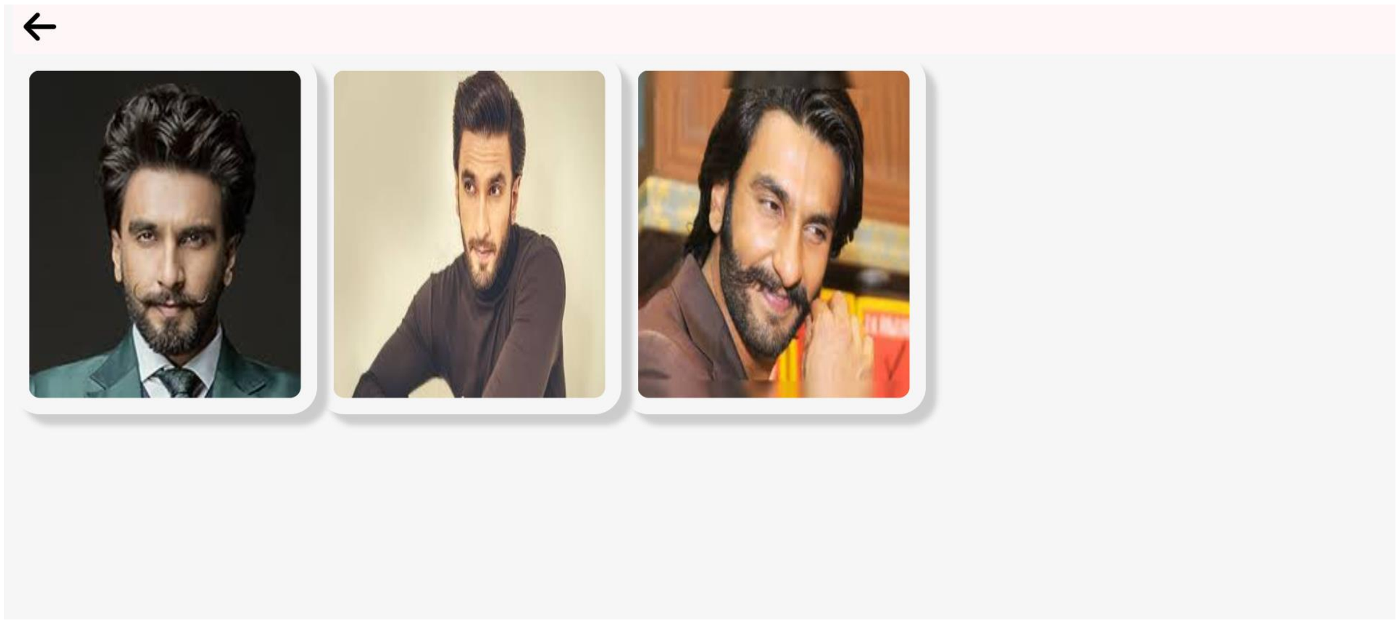
Search Page for Web Application



Web Page for Clustered Face Result



Web Page for creating Albums



Web Page for displaying Albums

5. TESTING

Software testing is an investigation conducted to provide information to partners regarding the nature of the goods item or administration being tested. Software testing can also provide a neutral, objective viewpoint on the product, allowing the company to see and understand the risks associated with programming implementation. The method for executing a programme or application with the goal of identifying programming errors (mistakes or other deformities) and ensuring that only the product item is fit for use is known as testing strategies.

Programming testing entails the implementation of a product portion or framework segment in order to evaluate at least one curiosity property. These properties show to what extent the section or framework underneath test:

● meets the requirements that guided its design and development,

● responds effectively to a variety of data sources,

● performs its functions in a timely manner,

● it is sufficiently usable,

● can be installed and used in its intended environment, and

● achieves the overall result that its partners desire.

Because the number of possible tests for even simple programming components is practically endless, all product testing employs some process to select tests that are feasible given the time and resources available. As a result, programming testing typically (but not always) involves running a program or application in the hopes of finding programming errors (mistakes or different imperfections). Testing is an iterative process because when one fault is corrected, it can reveal other, more serious bugs or even create new ones.

White box testing

White-box testing is a programmer testing approach that examines an application's internal structures or activities rather than its utility (for example discovery testing). White-box testing employs an insider's perspective of the framework, similar to programming skills, to conduct configuration tests.

The analyzer selects contributions to test different paths through the code and determine the standard yield.

Black box testing

This is intended to reveal errors in requirement specification without regard for the project's internal workings. This testing concentrates on the project's information domain, with the test case developed by splitting the input and output domains of programming – A method that ensures complete test coverage.

6. Conclusion and Future scope

Conclusion:

In conclusion, the development of a face clustering-based Web App has proven to be a significant achievement, providing an innovative and efficient solution for organizing and managing large collections of facial images. Through the integration of computer vision techniques and machine learning algorithms, the app successfully groups similar faces into distinct clusters, offering users a seamless and visually appealing experience. By offering enhanced organization, intuitive navigation, the app has the potential to significantly impact how individuals manage and relive their cherished memories.

Future Scope:

The Application can be extended in Future to add more functionalities such as we could explore additional machine learning algorithms to recognize a document and help users to copy text from an image. We will extend this application in future to recognize wanted criminals so that it will be helpful for government to catch suspects easily wherever they are present in the world. In future we can add temporary delete feature which enables users to recover their photos even after deleting them .In future we can also make use of cloud computing so that the user can access his/her photos anywhere from the world.

7.References

[1]https://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html

[2]https://pypi.org/project/face-recognition/

[3]https://flask.palletsprojects.com/en/2.3.x/

[4]https://www.npmjs.com/package/axios

[5]https://stackoverflow.com/questions/48156814/unable-to-install-face-recognition-library-for-python

[6]"Learning OpenCV 4 Computer Vision with Python" by Joseph Howse and Joe Minichino

[7]"Advances in Face Detection and Facial Image Analysis" by Fadi Dornaika and Mohamed Daoudi

[8] D. Yi, Z. Lei, S. Liao, and S. Z. Li, "Learning face representation from scratch," arXiv:1411.7923, 2014.

[9] O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Deep face recognition," in British Machine Vision Conference, 2015.

[10] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in CVPR, 2016.

[11] S. Liao, Z. Lei, D. Yi, and S. Z. Li, "A benchmark study of large-scale unconstrained face recognition," in IJCB, 2014.

[12] M. Turk and A. Pentland, "Face recognition using eigenfaces," in CVPR, 1991.

[13] T. F. Cootes, G. J. Edwards, and C. J. Taylor, "Active appearance models," IEEE Trans. on PAMI, vol. 23, no. 6, 2001.

[14] T. Ahonen, A. Hadid, and M. Pietikainen, "Face description with local binary patterns: Application to face recognition," IEEE Trans. on PAMI, vol. 28, no. 12, 2006.

[15] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma, "Robust face recognition via sparse representation," IEEE Trans. on PAMI, vol. 31, no. 2, 2009.

[16] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in NIPS, 2012.