Face Recognition Web Application

Submitted in partial fulfilment of the requirements

of the degree of

Bachelor of Technology

By

Apoorv Tomar

Roll no: 141100050

Under the guidance of

Ms. Pooja Gupta

Assistant Professor

Department of Computer Science Engineering

National Institute of Technology, Delhi

2017-2018

Declaration

I declare that this written submission represents my ideas in my own words and where other's ideas or words have been included, I have adequately cited and referenced the original

sources, I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea /data / fact/ source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not properly cited or from whom proper permission has not been taken when needed.

Name: Apoorv Tomar

Roll no: 141100050

Department of Computer Science Engineering

Date: 10/10/2017

To Whom So Ever It May Concern

This is to certify that project entitled "**Face Recognition Web Application**" is an original bonafide work carried out by B.Tech student **Apoorv Tomar** of National Institute of Technology, Delhi during academic year 2017-2018 under my supervision. The matter embodied in this project is a genuine work done by the student and has been submitted to the Department of Computer Science and Engineering, National Institute of Technology, Delhi for fulfilment of the requirements of the degree of BACHELOR OF TECHNOLOGY and I consider it worthy of acceptance for the award of the degree of **BACHELOR OF TECHNOLOGY** of the institute.

Signature of Supervisor
Ms. Pooja Gupta
Assistant Professor-CSE,
NIT Delhi.

Approval Sheet

This project work entitled by **Face Recognition Web Application** by **Apoorv Tomar** for the degree of Bachelor of Technology in Computer Science and Engineering.

Examiners

____

____

____

Supervisor(s)

____

____

____

Chairman

____

Date: ___/___/___

Place: __________

Contents

Abstract

The main objective of the project was to develop a face-based search web application. This web application would serve as a social platform where users could perform face-based search to find other registered users. It would provide an alternative method over conventional text-based search for social platforms. The platform uses face detection and face recognition techniques to successfully identify a user's face image.

In order to sign up for the platform, a person must upload their face image and fill out basic information such as name, age and date of birth.

The platform allows users to: -

- Perform face-based search on the platform by uploading face image of another registered user.
- Edit their profile.
- Share/Post textual content.

This project is in continuation from the work of previous semester. Last semester, I researched about state of the art techniques for face detection as well as face recognition. I prototyped an android application for Realtime face recognition. The application used openCV library to detect and recognize face images. However, it lacked the feature of uploading the face image as a file since it captured images Realtime using camera sensor of the device.

This semester, I aimed at developing a web platform which could be deployed to run both on mobile as a web app and on web browser for computer system. It would ask users for their permission to use camera sensor (if present in the device) and also provide the feature of uploading images as file (JPEG, JPG, PNG etc).

Layout of Project

- **Introduction-** Elaborates on the motivation of the project, background study for related tools. It highlights the aim and scope of the project that is to be done this semester.
- **Literature Review-** Critical appraisal of the previous work published in the literature pertaining to face detection and recognition. It also elaborates on state of the art techniques used by researchers nowadays.
- **Report on Present Investigation-** This section covers the requirements specification and design diagrams of the product system.
- **Implementation-** Contains a detailed insight to the various techniques used for development of the web app.
- **Results and Discussions-** All design issues and algorithms are covered in this chapter. Also details about the testing plan, risk and mitigation are explored.
- **Summary and Conclusion-** Re-enforces the logical findings of the project and throws light on its future scope.

Abbreviations, Notations and Nomenclature

1. **HTML**: Hypertext Markup Language is standard markup language for creating web pages and web application.
2. **CSS**: Cascading Style Sheets is a style sheet language used for describing presentation of a document written in markup language.
3. **API**: Application Programming Interface is a set of subroutine definitions, protocols, and tools for building application software.
4. **RESTful architecture**: A RESTful API is an API that uses HTTP requests to GET, PUT, POST and DELETE data.
5. **Web Service Endpoint**: It is the port upon which you connect a Web service client to the server.
6. **JSON**: JavaScript Object Notation
7. **AWS**: Amazon Web Services is a subsidiary of Amazon.com that provides on-demand cloud computing platforms to individuals, companies and governments.
8. **DB**: Database.

List of Figures and Tables

Figures

Tables

Chapter 1.1: Introduction

Section 1.1.1: General Introduction

With advent of wearable technology, learning about yourself and the surroundings around you has not only become high tech but real time. In today's world, there are a lot of devices and apps that help us track heart rate, our food consumption, monitor our heart rate etc. The "quantified self" is now a reality for everyday person. One of the upcoming devices in this domain is optical head mounted display (OHMD). Many technology analysts and enthusiasts have speculated that head mounted display will be the next iteration to the wearable and handy devices in the near future. This device will contain various sensors along with a Micro Controller Unit which would offer many different functionalities to the end user. One of the main features of such devices will be inbuilt Artificial Intelligence.

Through the use of camera sensor, The AI would be able to perform Real Time Image processing; perform face recognition of the person in the view of camera sensor. If the person gets recognized, the device will retrieve information about the identified person and notify the user. This would allow a user to know about a person in his surrounding, without interacting with him/her.

This particular functionality is the motivation behind my project.

Due to limited availability and high cost of the optical head mounted display, I decided to implement such feature in low budget environment.

Section 1.1.2: Problem Statement

Develop a web application which allows users to do face-based search using face images of other users and retrieve their information. The application should allow users to sign up for the platform, create profile, edit profile, search for other users and post textual content.

Section 1.1.3: Empirical Study (Field Survey, Existing Tool Survey, Experimental Study)

I researched about various existing tools/software available for face recognition.

- **DeepFace**[5] is a robust face recognition system developed by Facebook researchers which showed accuracy of 97.3% on unconstrained environment. It is patented by Facebook Inc. and isn't openly available to use.
- **APIs services**. There are many companies offering APIs calls to their cloud computing facilities. **AWS Rekognition service, Google Cloud Vision API** etc. are available to perform image content analysis. These Software as service, however, charge money per a number of API calls.
- **Openly available sources**: There are many open sourced projects/libraries/tools available on the internet for face detection and recognition. Trained models can be downloaded as xml files and then used to detect faces.

Section 1.1.4: Approach to problem in terms of technology/platform to be used

The main technologies and libraries used for the project were as follows: -

Subsection 1.1.4.2: Programming Languages

- **HTML**
  HTML is markup language which was used to create the front end for the web pages. All the necessary document elements like forms, divisions etc. were created using this language.
- **CSS**
  Cascading Styles Sheet, style sheet language, was used to set the visual style of the web pages, used to improve the UI/UX.

- **JavaScript**

  JS is high-level multi paradigm interpreted language. It was used for developing the backend of the web application. I did client-side scripting and server-side scripting using this technology. Also, there are large number of libraries and framework which help in faster and efficient development of website.

- **Ruby**

  Ruby is general purpose programming language. It is known for its simplicity and elegant syntax. I chose ruby for server-side scripting, to make database connection to NoSQL database.

- **Python**

  At the start of the project, I chose python for testing the various algorithms that I researched about. Python is known for its algebra, matrix capabilities. There are vast number of libraries available which help in reducing program size and effort required to code. I found various openly available projects on github on python, which helped me with my project.

- **SQL**

  Structured Query language was used for programming and designing database management system. MySQL software was used to create database and update/modify it.

Subsection 1.1.4.2: Libraries and Frameworks

- **Sinatra**

  It is free and open source software web application library in Ruby. It provides an alternative to other MVC frameworks such as Ruby on Rails etc. I chose Sinatra library as it reduces the development time for server-side scripting.

- **Dlib**

  It is a C++ library containing support for machine learning algorithms. It is widely used in academia and industry. It has various implemented functions for deep learning and machine learning. I used Dlib to implement Convolutional Neural Networks for face recognition.

- **Bootstrap**

  It is a free and open source front end library for designing web platforms. I is built upon CSS. It essentially divides a web page into tabular form, which allows easy development of individual columns of the webpage.

- **jQuery**

  It is a cross platform JavaScript library which is used for client-side scripting. It reduces the lines of code and promotes efficient development of the front end of a web application.

Section 1.1.5: Support for novelty/ significance of problem

There are many efficient Image Analysis systems which have shown high accuracy for the task of face recognition, however they are patented, and being used for different purposes.

Google Photos is used to find similarities between input image and image available on the web, it is known for excellent object identifying capabilities.

Deepface, by Facebook Inc., is being used only for automated tagging of users on digital images on the Facebook platform.

Others: There are many face recognition systems used by government facilities for identifying criminals and possible threats to the country.

But for my objective, there is no as such freely available web platform where users could register and explore each other’s profile using face images of another person.

Chapter 1.2: Review of Literature

Section 1.2.1: Summary of Papers Studied

Subsection 1.2.1.1: Face Detection

Viola Jones Algorithm [1]

The **Viola–Jones object detection framework** was proposed in 2001 by Paul Viola and Michael Jones. It is the object technique which gave competitive object detection rates in real-time for first time. It was primarily used for object detection, but it proved to be an efficient technique for detecting faces. It showed accuracy of 92% on an constrained environment.

It is still the most commonly used face detection framework.

The algorithm has four stages:

1. Haar Feature Selection
2. Creating an Integral Image
3. Adaboost Training
4. Cascading Classifiers

Haar Features

There are similar properties amongst different human faces. These common properties can be matched using **Haar Features**.

A few properties common to human faces are:

- The eye region is usually darker than the upper-cheeks in an image.
- The nose bridge region is brighter than the eyes due to illumination present more near nose bridge area.

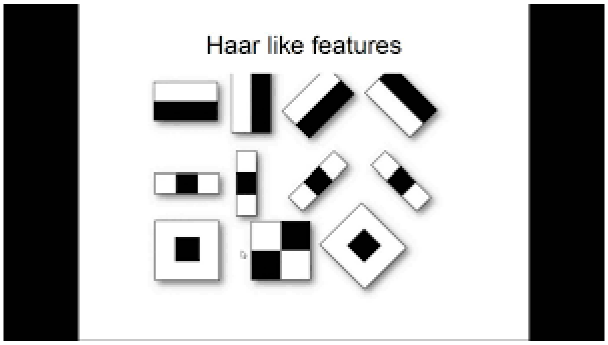


Fig 1.2.1.1: Haar Features

AdaBoost, short for *Adaptive Boosting*, is a machine learning technique. It is generally used along with other learning algorithms to improve performance. The output of other algorithms is summed into a weighted combination that represents the final result of the boosted classifier.

Subsection 1.2.1.2: Face Recognition

Eigen Faces [2]

Eigenfaces is essentially a set of eigenvectors. This approach of using eigenfaces for face recognition was introduced by Sirovich and Kirby (1987) and used by Matthew Turk and Alex Pentland in face recognition. This technique is based upon mathematical process called principal component analysis; PCA. The eigenvectors are derived from the covariance matrix of the probability distribution over the high-dimensional vector space of face images. The eigenfaces are used to construct the basis set which is used to derive a covariance matrix between face images. Classification is based on how the input face image can be formulated using the basis set of images i.e. Eigenfaces. This technique is susceptible to

Fischer Faces [3]

It is a technique in which Linear Discriminant Analysis (LDA) is used to reduce the dimensions instead of PCA.

The resulting basis vectors that define a subspace representation of set of face images are called Fischer faces.

This solution is more desirable if the goal is to classify rather than represent the face images.

Convolutional Neural Networks [4]

A convolutional neural network is a feed-forward network which has the ability of extracting topological properties from the input image. It can extract features from a raw image and then a classifier classifies the extracted features. The advantage of using CNNs is that they are invariant to distortions and other simple geometric transformations. Convolutional Neural Networks combine three architectural ideas to ensure some degree of shift, scale, and distortion invariance: local receptive fields, shared weights, and spatial or temporal sub-sampling. Back propagation technique is used to train the network.

A convolutional layer is used to obtain features from receptive fields in previous layers. Feature maps are built in formations of planes of convolutional layers to extract different types of local features. A weight is assigned to each join, but all units of a feature map have same weights. This technique is called weight sharing technique and is applied in all CNN layers, it reduces the number of trainable parameters. Logistic Regression classifier is then used to classify the set of images.

Section 1.2.2: Integrated Summary of Literature Studied

Detection and recognizing a face in a digital image is complex problem. Since, digital images are characterized by numerous external (environmental) factors, it has posed a hard challenge for researchers to devise a robust face detection and recognition algorithm. Factors that affect the accuracy of face detection techniques are:

1) Illumination: Humans appearance changes with varying illumination.
2) Different poses, expression and partially occluded images
3) Change in image position and variation in scale

Viola Jones algorithm is still commonly used due to high accuracy of upto 90% in normalized face images. However, researchers are now using deep learning techniques for face recognition. EigenFaces and FischersFace show good accuracy only in highly constrained environment. So, researchers have moved on from using these techniques. Instead, CNNs are being used to detect and classify face images. This technique is robust against different postures, illumination conditions and other external factors.

Chapter 1.3: Report on Present Investigation

Section 1.3.1: Overall Description

1.3.1.1 Product Perspective

The product has following constraints with respect to its requirements.

- **System Interfaces** shall operate on Linux, windows operating system.
- **System Interfaces** shall be given access to MySQL database.
- **System Interface** shall be allowed to use camera sensor of the device.
- **User Interfaces** shall operate on Chrome, Mozilla Firefox browsers.
- **User Interface** can have any screen format, aspect ratio.
- **Hardware Interface:** Devices with camera sensors are supported.
- **Software Interface:** Ruby Version 2.5

  MySQL Database 2.2

  ECMAScript 6, HTML5, CSS3

1.3.1.2 Product Functions

- The product will allow user to sign up, login, create & edit profile.
- The product requires face image at the time of sign-up.
- The product will allow user to post textual content.
- The product will take input as digital face images, pre-process it to detect faces.
- The product will return the best matched profile in case of positive recognition.

1.3.1.3 User Characteristics

- The product is developed in form of web application, so user must have basic knowledge of how to operate a web browser.
- User must know how to upload a digital image.

1.3.1.4 Constraints

- The product is only available in the form of web application.
- The product is dependent on AWS Rekognition service.

1.3.1.5 Assumptions and Dependencies

- Linux and Windows operating system are available to the user.
- Availability of camera sensor.

Section 1.3.2: Functional Requirements

S.No.	Functional requirements
1.	System shall be given access to camera sensor.
2.	System shall be able to take input from camera device.
3.	System shall be able to listen to HTTP requests.
4.	System shall be able to access AWS API.
5.	System shall be given privilege to create RDB.
6.	System shall be able to access Database.

Table 1.3.2 Functional Requirements

Section 1.3.3: Non-Functional Requirements

S.No.	Non-Functional Requirements
1.	The system response time should be low and easily accessible
2.	The system should be reliable for processing the digital image
3.	The system should be easily maintainable
4.	The system should be portable over the platforms like Windows and Linux distributions.
5.	The system and data should be secure, i.e. it should safeguard the user information and prevent the traffic data from manipulation attacks
6.	The data that has been collected should follow data integrity.

Table 1.3.3. Non-Functional Requirements

Section 1.3.4: Logical Database Requirements

A Relational Database Management System is required to store the user’s information.

S. No.	**Database**	**Purpose**
1.	MySQL Database	Required to store user’s login credentials, metadata, personal information etc.

Table 1.3.4 Logical Database Requirements

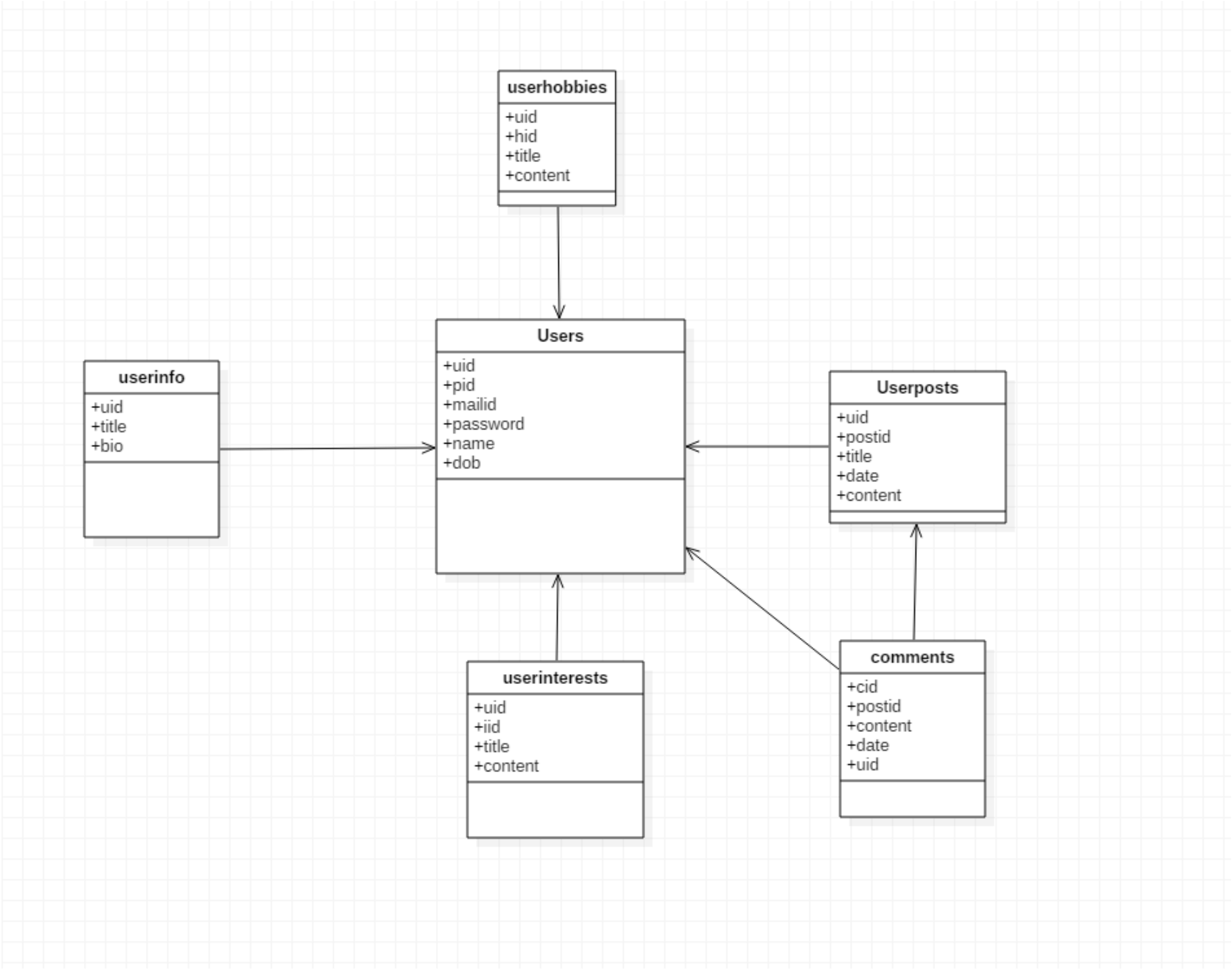


Fig 1.3.4 Database Schema

The database schema shown in the figure above is implemented on MySQL database.

The User table contains following attributes: -

- Uid (Primary Key) –to index users.
- Pid – to index user's profile picture
- Mailid – to store email of an user
- Name – to store the full name of user
- Password – to store hashed password of user
- Dob – to store the date of birth of user

The Userinfo table contains following attributes: -

- Uid (Foreign Key) – to refer to the Users table.
- Title – Stores the title/headline of the user's profile.
- Bio – stores the introduction of the user.

The Userinterests table contains following attributes: -

- Uid (Foreign Key) – to refer to the Users table
- Iid – to index the interests of a user
- Title – to store the name of the interest
- Content – to store the description about the interest

The Userhobbies table contains following attributes: -

- Uid (Foreign Key) – to refer to the Users table
- Hid – to index the hobbies of a user
- Title – to store the name of the hobby
- Content – to store the description of the hobby

The Userposts table contains following attributes: -

- Uid (Foreign Key) – to refer to the Users table
- Postid – to index the posts by a user
- Title – to store the title of the post
- Date – to store the date of the post
- Content – to store the textual content of post

The Usercomments table contains following attributes: -

- Uid (Foreign Key) – to store which user made the comment, and refer to the Users table
- Postid (Foreign Key)– to refer to Userposts table
- Cid – to index the comments
- Date – to store the date of the comment
- Content – to store the textual content of the comment

-

Section 1.3.5: Design Diagrams

Subsection 1.3.6.1: System Design

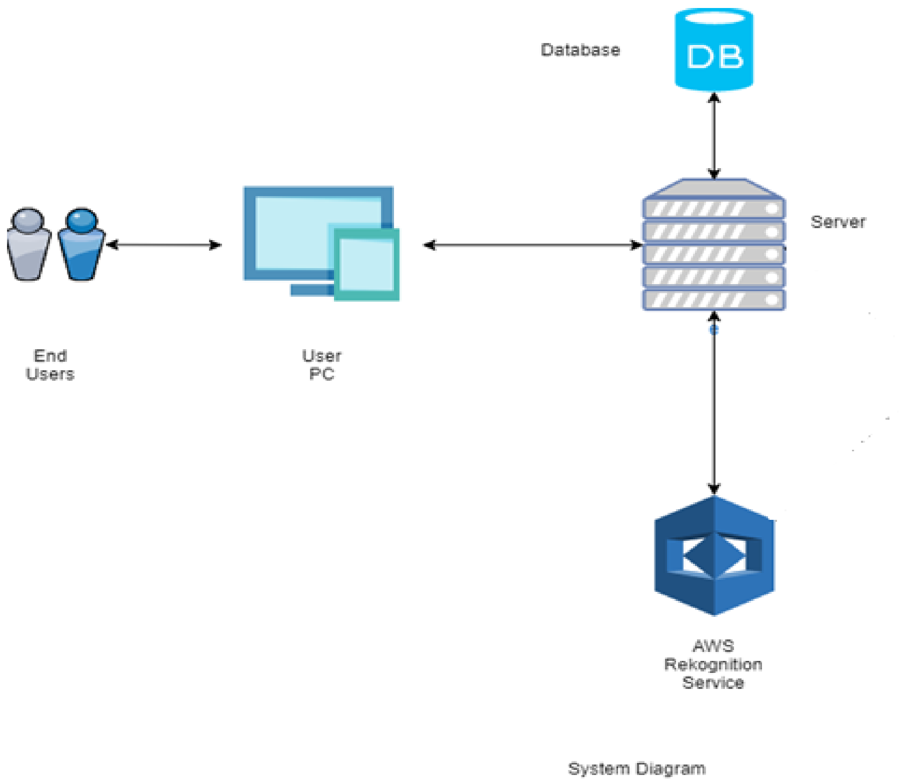


Fig 1.3.6.1 System Design

The above figure shows the overall system design and various entities involved.

Subsection 1.3.6.2: Sequence Diagram

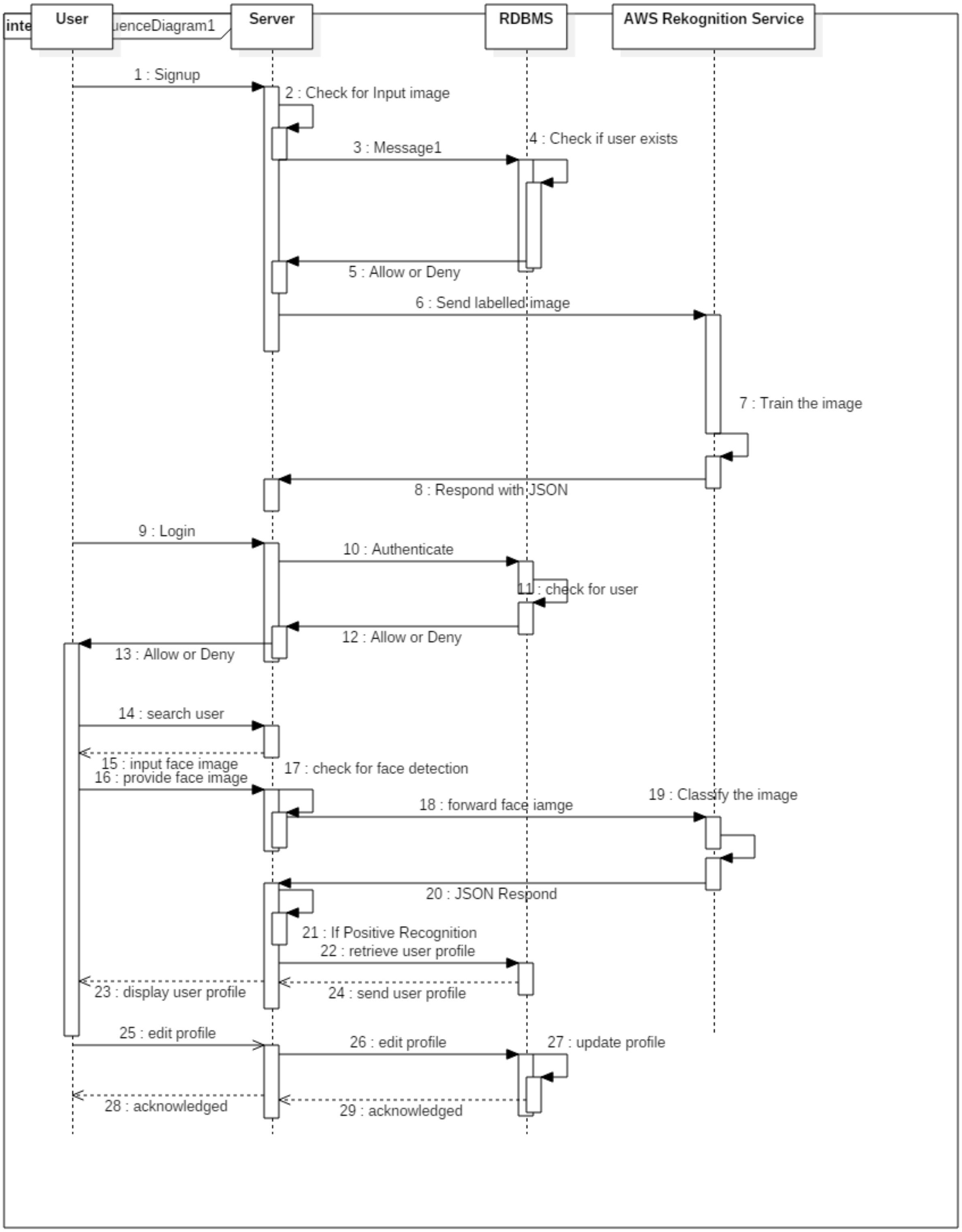


Fig 1.3.6.2 Sequence Diagram

Chapter 1.4: Results and Discussions

Section 1.4.1: Implementation Details and Issues

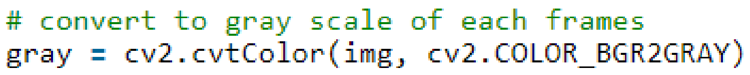
Subsection 1.4.1.1: Implementation Issues

- Normalizing the digital images uploaded to the platform
- Storing the profile pictures of users locally on the server.
- Indexing the face images corresponding to the users.
- Realtime uploading the pictures captured through camera sensor.
- Bypassing S3 storage on AWS platform, to directly send face images as Base64 encoded byte stream.
- Making API calls to the AWS endpoint, and retrieving & parsing the JSON response.

Subsection 1.4.1.2: Algorithms (Module wise- with respect to design)

Subsection 1.4.1.2.1: Face Detection Algorithm (Viola Jones)

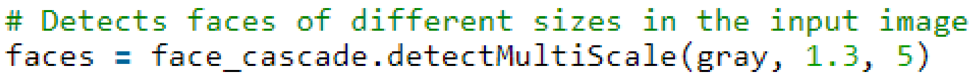
1. Input the image from the user. Using python program, preprocess the image.
2. Convert the image into grayscale color space.



```
# convert to gray scale of each frames
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
```

Fig 1.4.1.2.1 Code Snippet – change color space

3. Apply HAAR cascade classifiers on the stored image.to detect faces. If multiple set of coordinates are returned, then the image is discarded. If single set of coordinates is returned, the image is then sent for training.



```
# Detects faces of different sizes in the input image
faces = face_cascade.detectMultiScale(gray, 1.3, 5)
```

Fig 1.4.1.2.2 Code Snippet – face detect

Subsection 1.4.1.2.2: Face Recognition Module (AWS Rekognition)

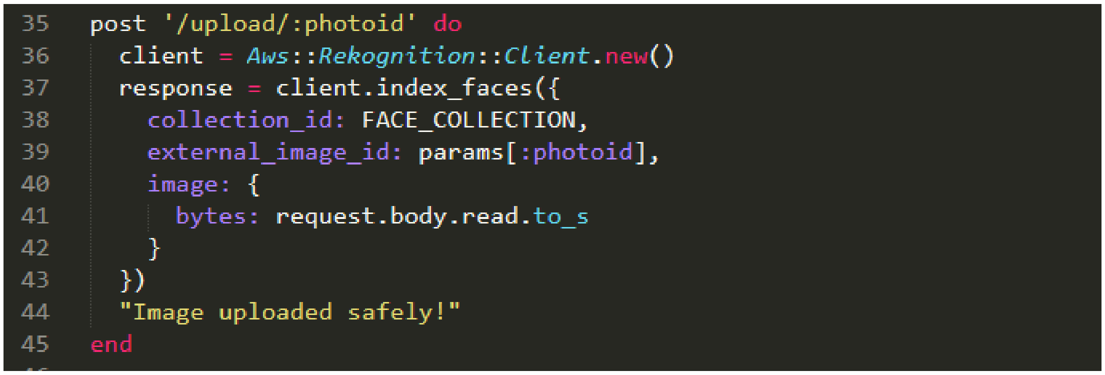
Training

1 After the image is parsed through the face detection module, it is assigned a label.

2 The image is then converted into Base64 stream of bytes.

3 The image with its label is sent using HTTP POST to the AWS *endpoint/upload/label* for API call.

4 The classifier at AWS Rekognition service is trained.



```ruby
post '/upload/:photoid' do
  client = Aws::Rekognition::Client.new()
  response = client.index_faces({
    collection_id: FACE_COLLECTION,
    external_image_id: params[:photoid],
    image: {
      bytes: request.body.read.to_s
    }
  })
  "Image uploaded safely!"
end
```

Fig 1.4.1.2.3 Sending HTTP POST

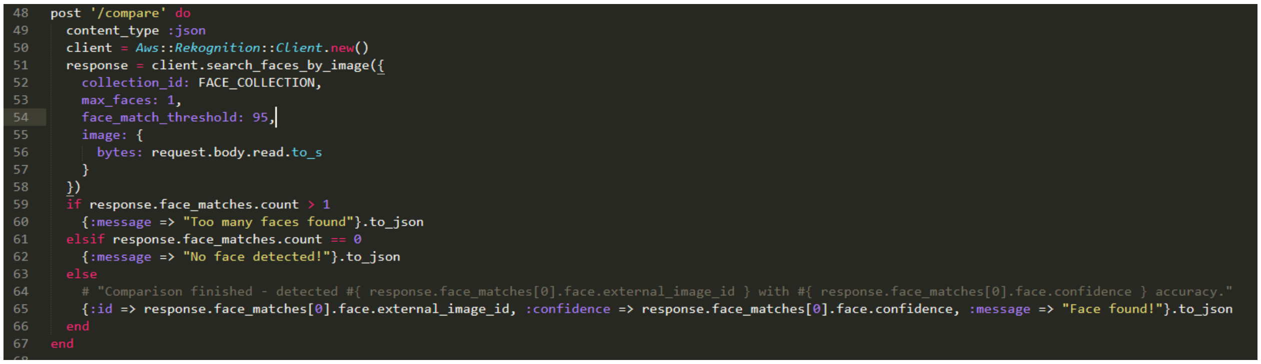
Classifying

1 Threshold value for confidence of matched image is set for AWS rekognition.

2 API call is made at *endpoint/compare*, to retrieve the result of face recognition.

3 The Rekognition service responds with JSON object. The server retrieves JSON object using HTTP GET.

4 If the image is matched, it returns the confidence parameter with the label of the image.



```ruby
post '/compare' do
  content_type :json
  client = Aws::Rekognition::Client.new()
  response = client.search_faces_by_image({
    collection_id: FACE_COLLECTION,
    max_faces: 1,
    face_match_threshold: 95,
    image: {
      bytes: request.body.read.to_s
    }
  })
  if response.face_matches.count > 1
    {:message => "Too many faces found"}.to_json
  elsif response.face_matches.count == 0
    {:message => "No face detected!"}.to_json
  else
    # "Comparison finished - detected #{ response.face_matches[0].face.external_image_id } with #{ response.face_matches[0].face.confidence } accuracy."
    {:id => response.face_matches[0].face.external_image_id, :confidence => response.face_matches[0].face.confidence, :message => "Face found!"}.to_json
  end
end
```

Fig 1.4.1.2.4 Retrieving JSON object

Section 1.4.2: Risk Analysis and Mitigation

Risk

1. The AWS allows to make API calls to Rekognition service for a limited number of times. Since I’m using free tier package, I can only make 1000 calls per month.
2. Creating face image database for the registered users poses memory problem. Even after reducing the size of the image, it poses scalability issue.

Mitigation

1. To remove the dependency on the AWS Rekognition service, a robust face recognition module can be developed which will be integrated with the face detection module.
2. Instead of storing images on the main server, the images could be stored on cloud database and then tried to access. However, this may increase network load.

Section 1.4.3: Test Plan

Type of Test	Will/Was test performed?	Comments/Explanations	Software Components
Requirement Testing	Yes	None	Windows, Web Browser
Unit	Yes	Unit test cases written in Ruby	Web Browser, Ruby
Integration	Yes	Tested Face detection and Face Recognition module after integration.	Web browser
Performance Test	Yes	Checked performance using the web browser console.	Web Browser, Ruby, JavaScript.
Data Security Test	Yes	Tested reliability of login page.	MySQL

Table 1.4.3 Test Plan

Section 1.4.4: Component decomposition

S.No	Components that require testing	Type of testing Required	Technique for writing test cases
1	Face Detection module	Integration, Unit, Performance, Requirement	White box
2	Login/signup webpage	Unit, Data Security, Integration	White box
3	Face Recognition module	Integration, Unit, Performance, Requirement	White box
4	Upload face image	Unit, performance, Requirement	White box

Table 1.4.4.1 Component decomposition and tests required

Test Cases for the Web App:

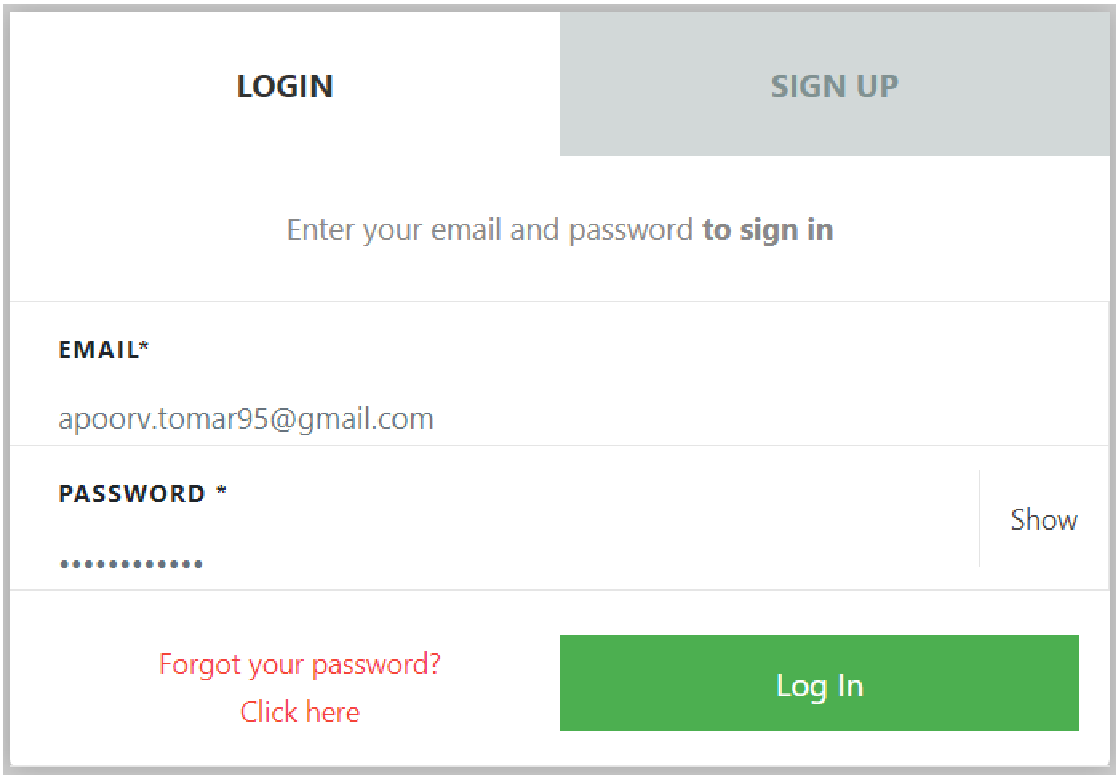
Test Case ID	Input	Expected Output	Status
T1	Enter user’s login credentials	Redirect to user *homepage*	Pass
T2	Upload face image	Successfully Uploaded	Pass
T3	Click search	Return Face Recognition Result	Pass
T4	Edit Profile	Profile of the user is updated	Pass
T5	Click Logout	Redirected to Login page	Pass

Table 1.4.4.2 Tests Cases for the web app

Section 1.4.5: List all test cases

Test Case 1:- Login page

Entering a registered user’s credentials and clicking on login button



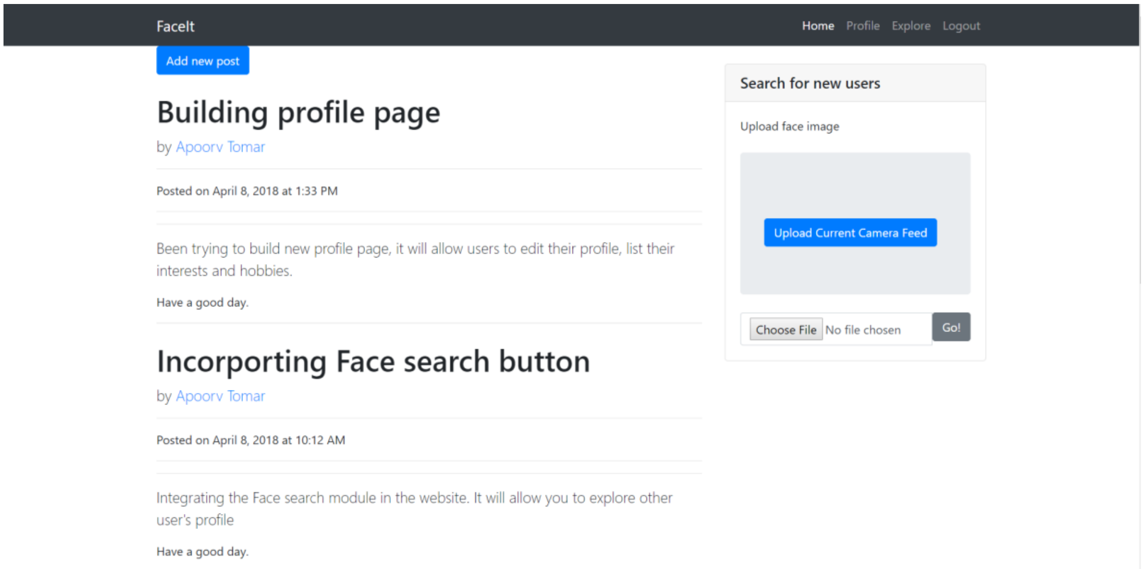
Enter your email and password **to sign in**

EMAIL*

apoorv.tomar95@gmail.com

PASSWORD *

••••••••••••



Building profile page

by Apoorv Tomar

Posted on April 8, 2018 at 1:33 PM

Been trying to build new profile page, it will allow users to edit their profile, list their interests and hobbies.

Have a good day.

Incorporting Face search button

by Apoorv Tomar

Posted on April 8, 2018 at 10:12 AM

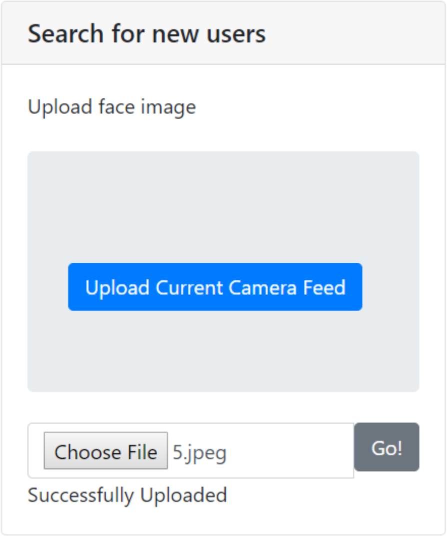
Integrating the Face search module in the website. It will allow you to explore other user's profile

Have a good day.

Fig 1.4.5.1 Test Case T1

Test Case 2: - Uploading face image

Successfully uploading a face image.

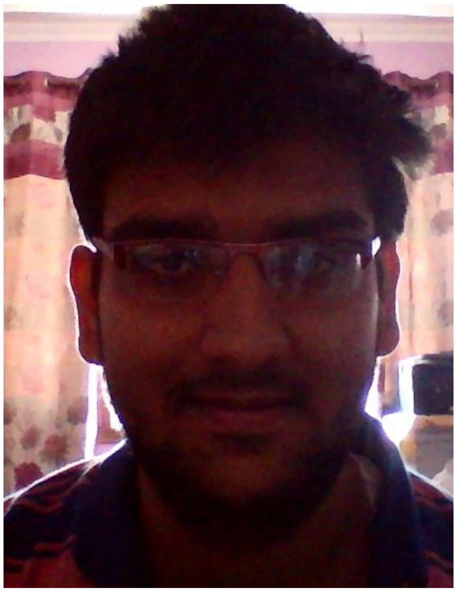


Search for new users

Upload face image

5.jpeg

Successfully Uploaded

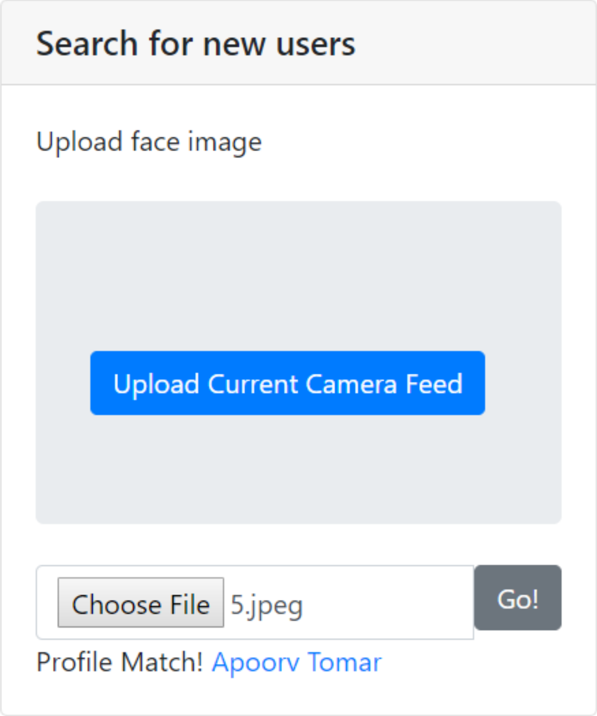


5.jpep

Fig 1.4.5.2 Test Case T2

Test Case 3: - Face Recognition module

Finding the profile of a user after uploading face image.

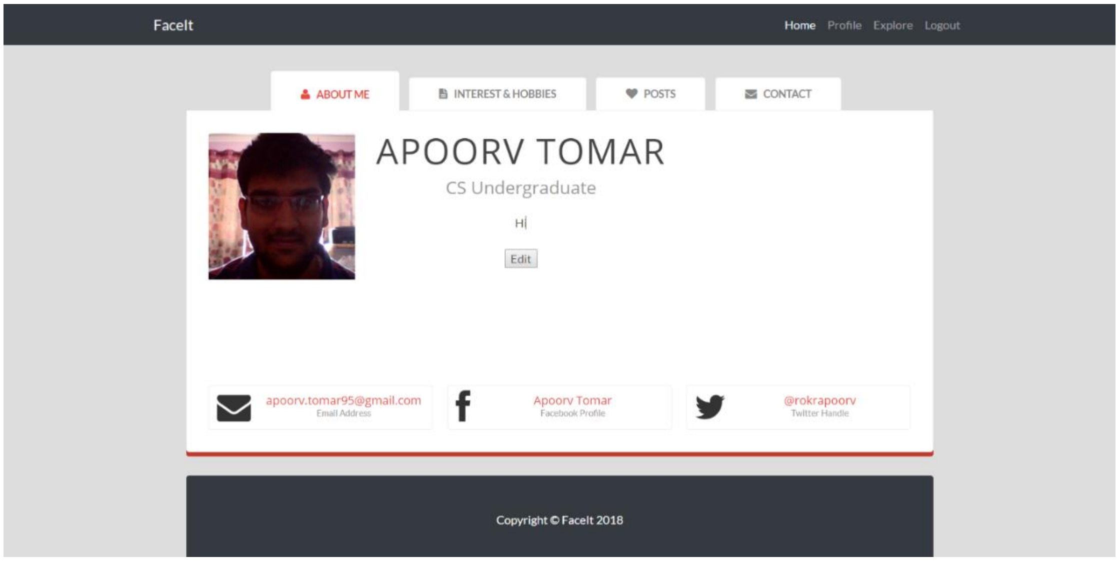


Search for new users

Upload face image

5.jpeg

Profile Match! Apoorv Tomar



APOORV TOMAR

CS Undergraduate

Hi

apoorv.tomar95@gmail.com
Email Address

Apoorv Tomar
Facebook Profile

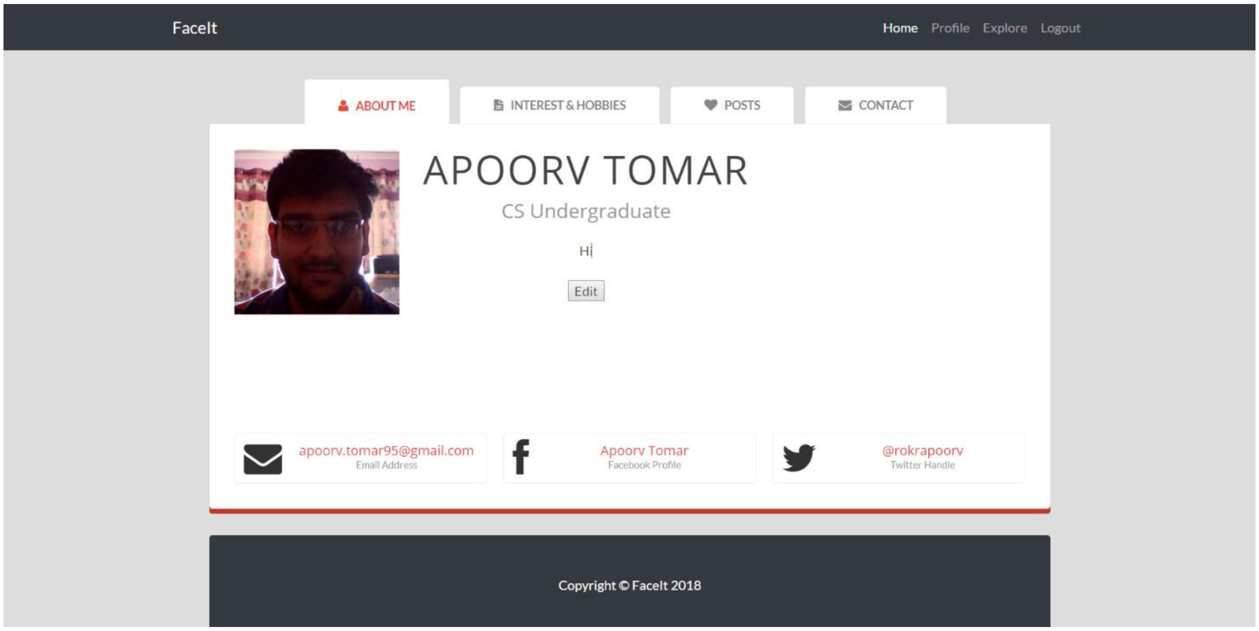
@rokrapoorv
Twitter Handle

Copyright © FaceIt 2018

Fig 1.4.5.3 Test Case T3

Test Case 4: - Profile Page

Editing the profile page and pressing Edit button.



APOORV TOMAR

CS Undergraduate

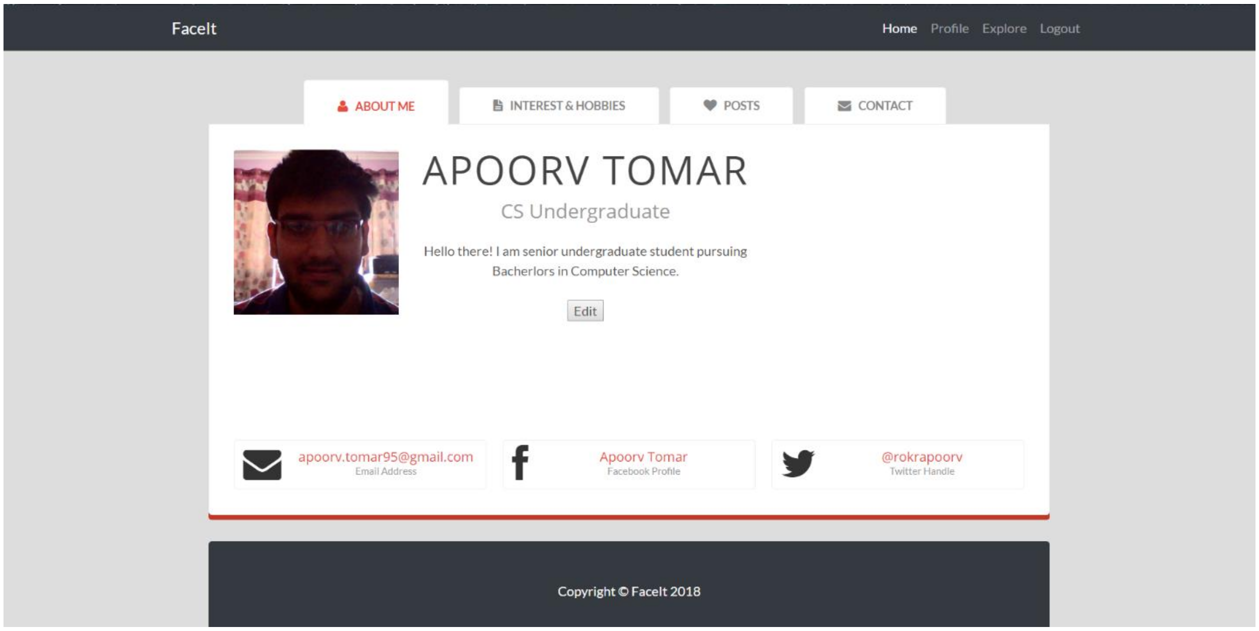
Hi

apoorv.tomar95@gmail.com
Email Address

Apoorv Tomar
Facebook Profile

@rokrapoorv
Twitter Handle

Copyright © FaceIt 2018



APOORV TOMAR

CS Undergraduate

Hello there! I am senior undergraduate student pursuing Bacherlors in Computer Science.

apoorv.tomar95@gmail.com
Email Address

Apoorv Tomar
Facebook Profile

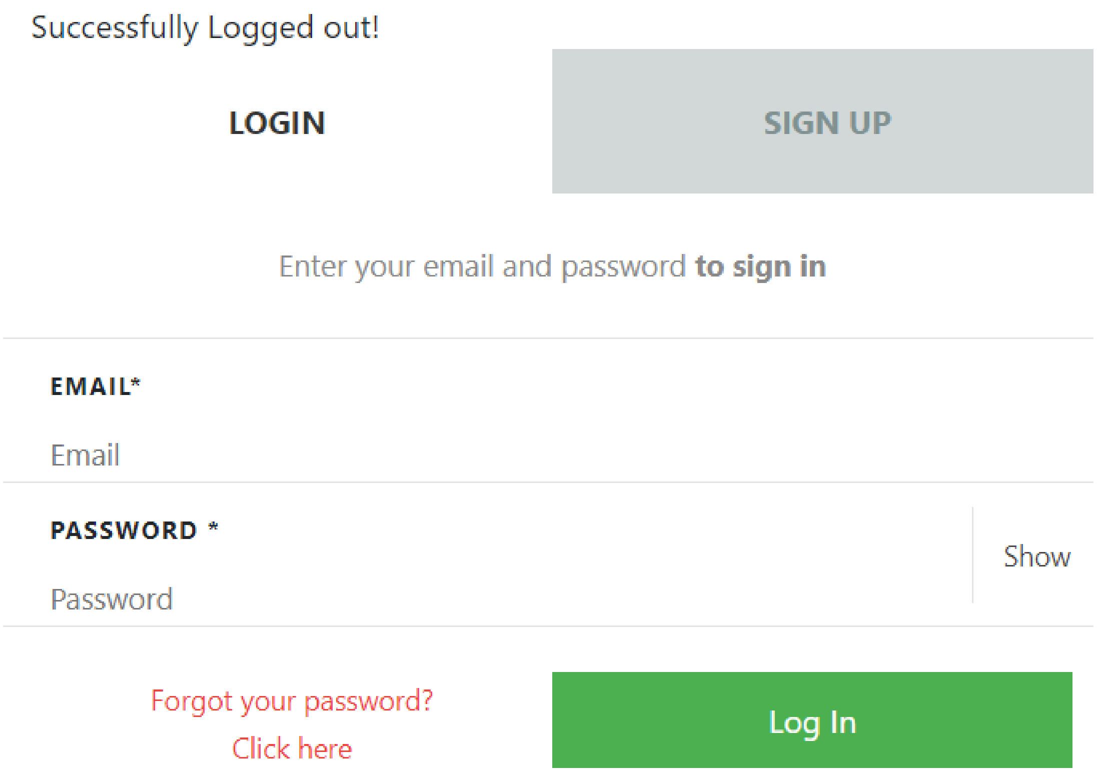
@rokrapoorv
Twitter Handle

Copyright © FaceIt 2018

Fig 1.4.5.4 Test Case T4

Test Case 5: - Logout

Pressing logout button.



Successfully Logged out!

Enter your email and password **to sign in**

Fig 1.4.5.5 Test Case T5

Section 1.4.6: Limitations of the solution

- Since the web application is dependent on third party service, hence only limited number of API calls can be made.
- The solution formulated is not scalable; t will require large memory to store face images of each user.

Chapter 1.5: Summary and Conclusion

The purpose of developing this project was to increase my skill-set and knowledge in field of computer vision and gets hands on experience on web development and cloud computing. This project led me to work with various technologies like Sinatra library, RESTful API, AWS Stack etc.

A primitive web application is designed to input face image real-time through camera sensor of device and push the image to the cloud. The classification of the images is done on the cloud machine hence reducing the computation cost of the main server. For now, User of the application can simply test the platform for its face recognition accuracy.

In future, I plan to create a robust platform where users would be able to do faced based search to find other users, I plan to develop and integrate face recognition module which would run locally on the server system.

Instead of storing the face images, I would extract some features from the images and store those values in database. This would reduce the product's dependency on memory as indexing could be done using these features.

Literature Cited

Research Papers

[1] M. Y. M. R. L. W. Yaniv Taigman, DeepFace: Closing the Gap to Human-Level Performance in Face Verification, 2014.

[2] P. Viola and M. J. Jones, "Robust Real-Time Face Detection," *International Journal of Computer Vision*, p. 137–154, 2004.

[3] M. Slavković and D. Jevtić, "Face Recognition Using Eigenface Approach," *SERBIAN JOURNAL OF ELECTRICAL ENGINEERING* , vol. 9, pp. 121-130, 2012.

[4] P. N. Belhumeur, J. P. Hespanha and D. J. Kriegman, "Eigenfaces vs. Fisherfaces: Recognition," *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, vol. 19, no. 7, pp. 711-720, 1997.

[5] H. Li, Z. Lin, X. Shen, J. Brandt and G. Hua, "A Convolutional Neural Network Cascade for Face Detection," *Computer Vision and Pattern Recognition (CVPR),* 2015.

Web Resources

[6] "Sinatra Documentation," [Online]. Available: http://sinatrarb.com/documentation.html.

[7] "Rekognition Service Documentation," [Online]. Available: https://aws.amazon.com/documentation/rekognition/.

[8] "Face Detection using openCV," [Online]. Available: https://docs.opencv.org/3.4.1/d7/d8b/tutorial_py_face_detection.html.

[9] "Accessing Device camera using Javascript," [Online]. Available: https://github.com/jhuckaby/webcamjs.

[10] "Database Connection using Ruby," [Online]. Available: https://www.jetbrains.com/help/ruby/connecting-to-a-database.html.

Acknowledgement

I express my profound gratitude and indebtedness to Ms. Pooja Gupta Assistant Professor, Department of Computer Science Engineering, National Institute of Technology Delhi for providing her valuable suggestions throughout the project work.

Most importantly, I would like to extend my gratitude to my parents whose help and support made this endeavor possible.

I perceive as this project as a big milestone in my professional development. I will strive to use gained skills and knowledge in the best possible way, and I will continue to work on their improvement, in order to attain desired career objectives.