5-29-2018

Login Authentication with Facial Gesture Recognition

Ruiwen Li
Santa Clara University, rli2@scu.edu

Songjie Cai
Santa Clara University, scai@scu.edu

Tor Saxberg
Santa Clara University, tsaxberg@scu.edu

SANTA CLARA UNIVERSITY
DEPARTMENT OF COMPUTER ENGINEERING

Date: May 31, 2018

I HEREBY RECOMMEND THAT THE THESIS PREPARED UNDER MY SUPERVISION BY

Ruiwen Li
Songjie Cai
Tor Saxberg

ENTITLED

Login Authentication with Facial Gesture Recognition

BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

BACHELOR OF COMPUTER SCIENCE AND ENGINEERING

Thesis Advisor

Department Chair

Login Authentication with Facial Gesture Recognition

by

Ruiwen Li
Songjie Cai
Tor Saxberg

Submitted in partial fulfillment of the requirements
for the degree of
Bachelor of Computer Science and Engineering
School of Engineering
Santa Clara University

Santa Clara, California
May 29, 2018

Login Authentication with Facial Gesture Recognition

Ruiwen Li
Songjie Cai
Tor Saxberg

Department of Computer Engineering
Santa Clara University
May 29, 2018

ABSTRACT

Facial recognition has proven to be very useful and versatile, from Facebook photo tagging and Snapchat filters to modeling fluid dynamics and designing for augmented reality. However, facial recognition has only been used for user login services in conjunction with expensive and restrictive hardware technologies, such as in smart phone devices like the iPhone x. This project aims to apply machine learning techniques to reliably distinguish user accounts with only common cameras to make facial recognition logins more accessible to website and software developers. To show the feasibility of this idea, we created a web API that recognizes a users face to log them in to their account, and we will create a simple website to test the reliability of our system. In this paper, we discuss our database-centric architecture model, use cases and activity diagrams, technologies we used for the website, API, and machine learning algorithms. We also provide the screenshots of our system, the user manual, and our future plan.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Artificial intelligence and Machine Learning have gained increasing popularity in recent years due to their ability to handle tasks that would otherwise take too much computational power, and due to their versatility, the wide range of problems they have been shown to solve. One of the most well known tasks that machine learning has made possible is face recognition.

Face recognition technology has been used for a variety of applications including automatic tagging in Facebook photos, Snapchat lenses that overlay dog ears on someones head, and security and surveillance, with the more recent capacity to track individuals moving throughout a closed space as they cross in front of security cameras .

Facial recognition systems rely on unique facial features as an additional layer of security to identify and distinguish people whether theyre new faces or old ones in a database. We set out to apply this technique to the field of internet security, along with Captchas, Im not a robot checkboxes, security questions, two factor authentication, and many others. Facial recognition has the potential to be a much simpler approach to security than remembering additional security information or connecting other accounts and devices.

1.2 Problem

However, theres an obvious issue with using facial recognition to login to your account. Anyone with you picture can log into it too. Furthermore, if anyone hacks the site, they may gain access the the database of face data and be able to relate user accounts to actual faces, then do some reverse processing to label those faces with real names rather than whatever alias may have been their username. And is facial recognition even accurate enough to avoid false positives and log someone into the wrong account? Can someone just sit in front of a camera long enough to get sneak past a sites security?

Apple has provided a solution to solve this photo trick by relying on dual cameras and an array of projected infrared dots to detect depth in its new facial recognition system. However, such a solution is limited to devices with expensive hardware upgrades and cant be applied to lower cost applications. Higher costs often limit other improvements, complicate manufacturing, and raise the price for consumers.

1.3 solution

To address the issues of impersonation, we propose a system that would use a video stream, rather than a still image, to check that the correct person is logging in. if our facial recognition detects a video frame without the correct matching face, the login step will fail. However, such a system could be spoofed by simply holding a video up to the camera, so we will also ask the user to perform some random gesture to ensure its a real person in front of the camera.

Our solution is pure software-based, requiring no additional hardware expenses, and can be applied to a wide range of applications including building security, unlocking cellphones, and website logins. The reliability and ease of use of our system will be reliant on the accuracy of the facial recognition and the set of gestures available to use.

Chapter 2

Requirements

2.1 Functional Requirements

- Critical:
  - Account creation and login system
  - Certification with camera
  - Distinguish between real person and images
  - Recognize faces and gestures

The functional requirements of our project describe features our system must have to be successful. Users need to be able to create accounts and store their facial data for the site to identify them later. The site needs to be able to identify users with facial recognition, which requires some form of video camera. The site must be able to stop impersonation,distinguishing between real people and fake copies. And finally, the site needs to be able to recognize faces and gestures to log users into the correct accounts

2.2 Non-functional Requirements

- Critical:
  - Secure API
- Recommended:
  - Fast authentication
- Suggested:
  - Continuous improvement

The non-functional requirements describe features that improve the sites performance, or would otherwise benefit the system. A secure API for storing and using face data to identify users and learn new faces would be ideal to avoid the risk of losing identifiable information in the event of a site hack. Since we imagine nobody wants to sit in front of a camera for minutes on end trying to access an account, the site should be able to identify users quickly, and provide a quick login experience. Since Software programs often require continuous debugging efforts as new features are added and new issues are discovered, the source code should be clear and easy to follow, relying on software tools to simplify the implementation.

2.3 Design Constraints

- Uses computer camera
- Uses video
- Web based platform
- Camera frame speed

The constraints are limiting factors of our system, which in our case is mostly technology. Since we are building a login system for websites, the APIs and other software tools need to work well with websites. We also expect that everyone will be logging in with a computer (we arent supporting phones for this project), so our facial recognition needs to work in poor lighting conditions and without relying on expensive camera features.

Chapter 3

Use Cases

The use case represents the list of actions and event steps which define the interactions among users, websites, and the API.

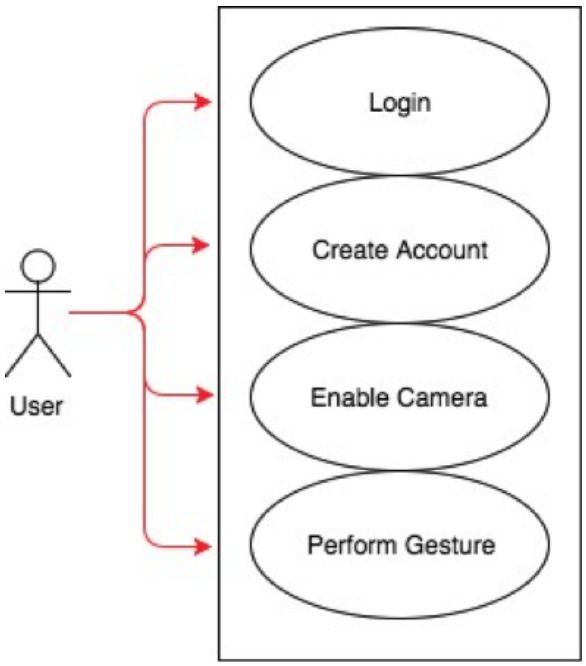

Figure 3.1: Use case diagram of the user

3.1 User

3.1.1 Use Case 1: Login

- Name: Login
- Goal: Provide user access to the system
- Actor: User
- Pre-conditions:
  - The user has an active connection to website

  - The user has previously signed up
  - The user knows his or her username
- Steps:
  - The user types in the username
  - The user clicks "verify"
  - The system recognizes the user's face and verifies that he or she is a real person
- Post-conditions:
  - The user's username has been verified in the system
- Exception: The user enters invalid username

3.1.2 Use Case 2: Create account

- Name: Create account
- Goal: Provide user access to the system
- Actor: User
- Pre-conditions:
  - The user has an active connection to website
- Steps:
  - The user types in the username
  - The system takes some photos of the user
  - The system stores the pictures and username in the database
- Post-conditions:
  - The user's account is created
- Exception: N/A

3.1.3 Use Case 3: Enable camera

- Name: Enable camera
- Goal: Provide the camera in user's computer access to record user's face
- Actor: User
- Pre-conditions:
  - User's username has been verified
- Steps:
  - The system pops out a request to enable camera
  - The user clicks "yes"
- Post-conditions:
  - The camera is enabled
- Exception: The computer does not have a camera

3.1.4 Use Case 4: Perform gesture

- Name: Perform gesture
- Goal: Verify whether the user is a real person or a photo
- Actor: User
- Pre-conditions:
  - The user enters valid username
  - The user's camera is enabled
  - The user's face has been detected by the system
- Steps:
  - The system displays a certain gesture for the user to perform
  - The user perform the gesture
  - The system analyze the user's gesture
- Post-conditions:
  - User successfully login
- Exception: The user performs wrong gesture

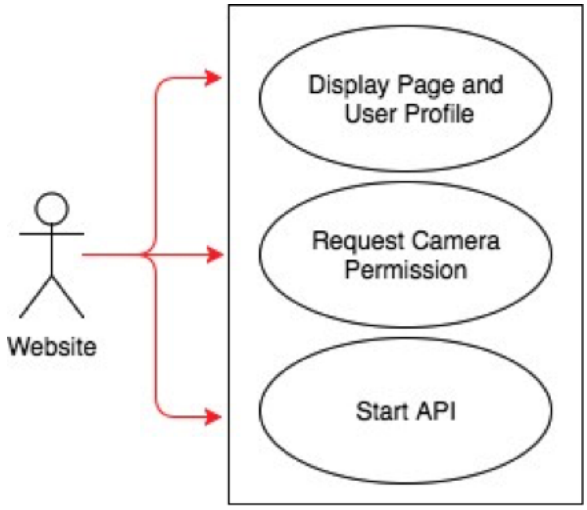

Figure 3.2: Use case diagram of the website

3.2 Website

3.2.1 Use Case 1: Display page

- Name: Display page
- Goal: Provide user access to the page in browser
- Actor: Website

- Pre-conditions:
  - The user has an active connection to website
  - The browser is compatible to the user's computer
- Steps:
  - Read HTML, CSS, and JavaScript files
  - Execute python files to make operations with server
- Post-conditions:
  - The page is successfully displayed
- Exception: 404 Not Found

3.2.2 Use Case 2: Start API

- Name: Start API
- Goal: Retrieve information using API technologies
- Actor: Website
- Pre-conditions:
  - The page is successfully display
  - User is interacting with the system
- Steps:
  - The system send a request to other websites using API to request information
  - The system receives responses from other websites
- Post-conditions:
  - The result is successfully return by the API
- Exception: The API does not work

3.2.3 Use Case 3: Display user profile

- Name: Display user profile
- Goal: Provide user permission to see his or her information
- Actor: Website
- Pre-conditions:
  - The user is successfully logged in
  - The user's browser is compatible with his computer
- Steps:
  - The system user's information from the database
  - The system lists the information on the web page
- Post-conditions:
  - User's information is successfully displayed
- Exception: Database connection error

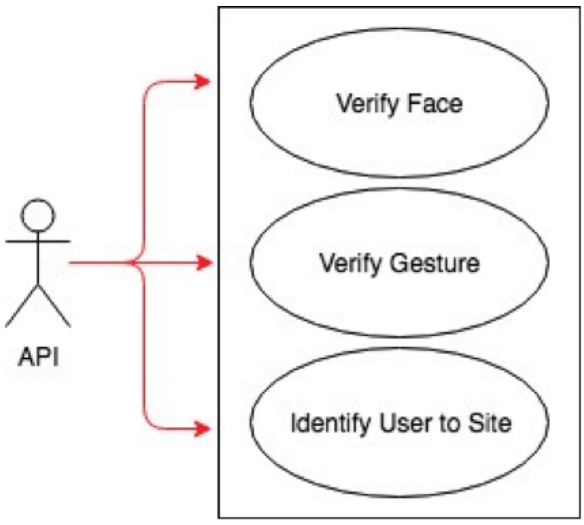

Figure 3.3: Use case diagram of the API

3.3 API

3.3.1 Verify Face

- Name: Verify face
- Goal: Recognize user's face in the camera
- Actor: API
- Pre-conditions:
  - The user's camera is enabled
- Steps:
  - Separate the camera video into multiple frames
  - Use pre-trained machine learning model to check if there is a face in the frame
  - Draw bounding box around the face detected
- Post-conditions:
  - User's face is successfully detected
- Exception: User's face is not found

3.3.2 Verify Gesture

- Name: Verify gesture
- Goal: Test if the user is real
- Actor: API
- Pre-conditions:
  - User's face is detected

- Steps:
  - The system displays a specific gesture on the screen
  - The user perform the gesture accordingly
  - The system verify is the gesture is correct
- Post-conditions:
  - The gesture is verified and user is logged in
- Exception: User performs wrong gesture

3.3.3 Identify user to site

- Name: Identify user to site
- Goal: Map user's face with those in the database
- Actor: API
- Pre-conditions:
  - The user's face is detected
- Steps:
  - Search the database
  - Pick the user in the database with the most similar picture as the login user
  - return the user's information
- Post-conditions:
  - User is found in the database
- Exception: Database connection error

3.4 Swim Lane Diagram

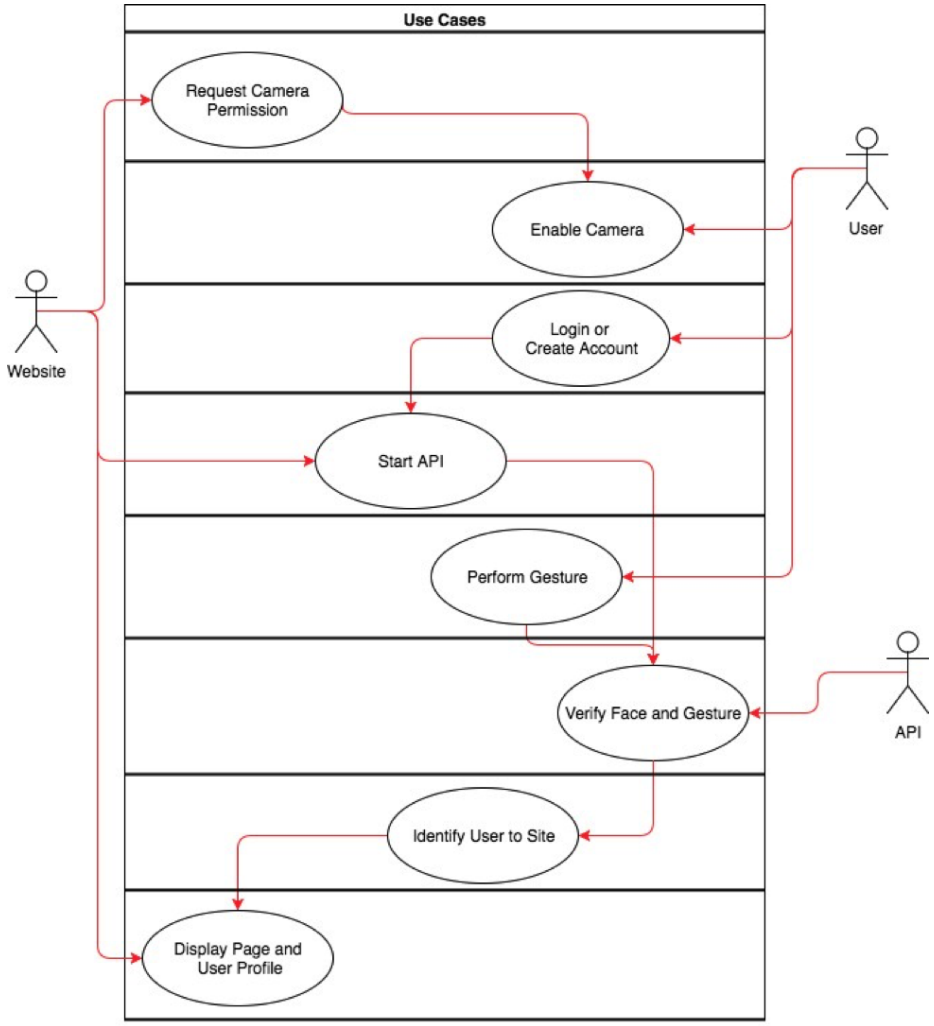

Figure 3.4: Use case diagram of the whole system in a timely basis

Chapter 4

Activity Diagrams

4.1 User

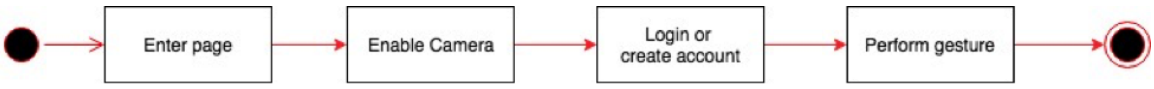

Figure 4.1: High level view of user operations

As a user, after he/she enters the index page of the website, he/she may attempt to log into by enabling the camera or create an account. The user needs to wait for the system response after the system finishes recording. After that, the user should perform given gestures from the server

4.2 Website

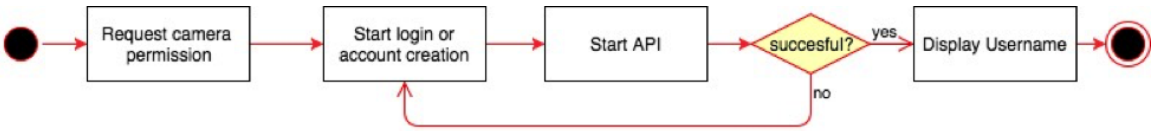

Figure 4.2: High level view of website operations

The website will always wait for camera permission to start the next step. Depending on different user operations, the website will call different APIs. After the user successfully logs in, the username will be display on the page

4.3 API

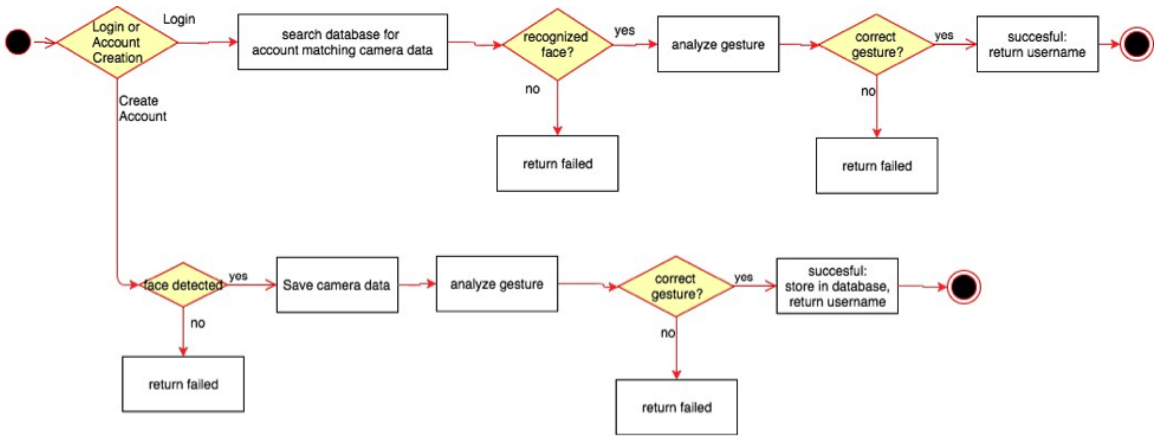

Figure 4.3: High level view of API operations

If the server sends a "login" request, the API will be called to camera recording, which will be sliced into pictures later. If the face is recognized, the API analyzes the gesture and returns the username if both tests are passed.

For account creation, the camera data will be saved if the face is detected. After that, the gesture will be analyzed and the user's data will be saved in database if the gesture is correct

Chapter 5

Architecture

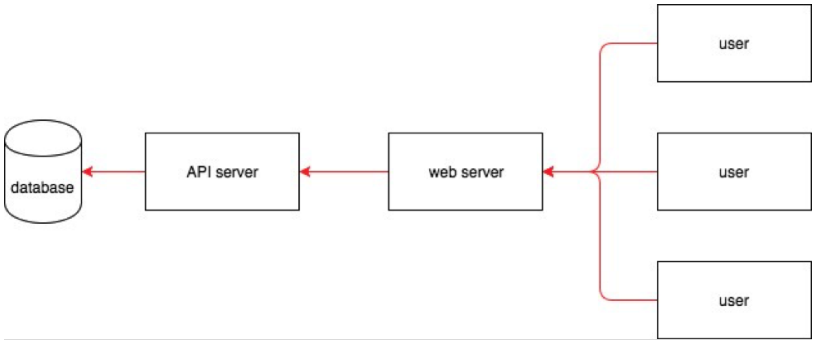

Figure 5.1: Architecture diagram of the system

Since all data will be stored in our database, We decide to build an advanced data-centric architecture for our design to make the data more accessible and manageable for users. In addition, considering security and management of APIs, we choose to have a three-tier architecture because it helps protect data security and improve manageability of different APIs. In our design, users post requests to the web server and the server will call different APIs depending on the requests from users. Multiple users can login to the system at same time from different places.

Chapter 6

System Implementation

6.1 Main Page

The main page (See figure 6.1) shows two components of the system, login and registration. When the user first enters the page, he or she can either choose to login or create and account.

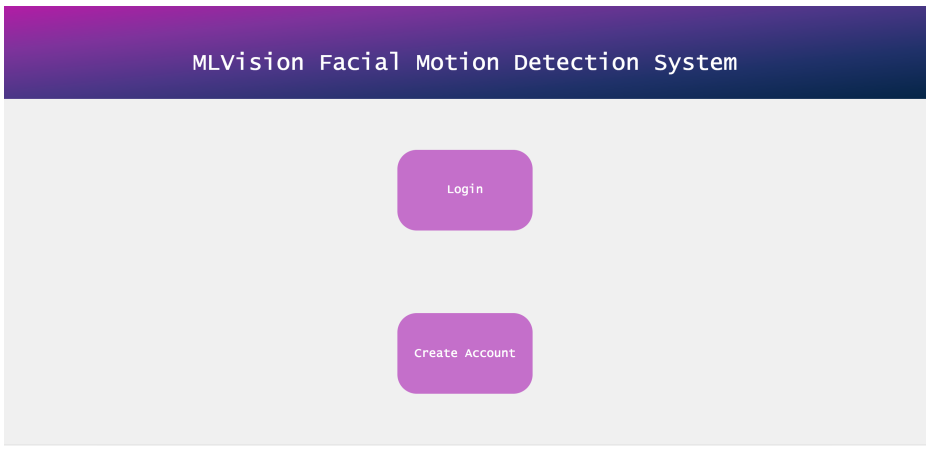

Figure 6.1: Main page of the system

6.2 Enter Username

The the username entering page (See figure 6.2) allows the user to enter his or her username in both login and registration process. During the registration, if the username conflicts with those in the database, the system will have a warning "Username already exists".

6.3 Enter Password

The password entering page (See figure 6.3) allows user to enter the password. During the login process, if the password is incorrect, the system will show the warning "Invalid username or password".

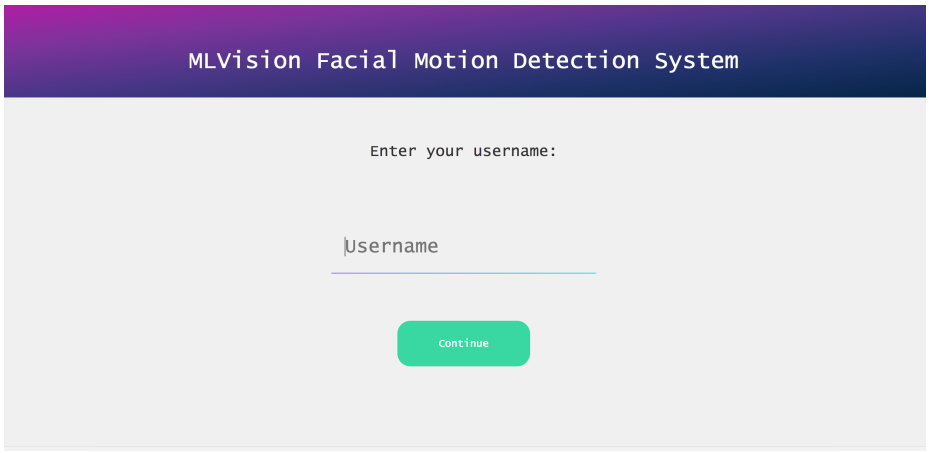

Figure 6.2: Entering username conceptual model

Figure 6.3: Entering password conceptual model

6.4 Facial Detection and Recognition

The facial detection and recognition technique (See figure 6.4) retrieves machine learning algorithm, comparing user's face data from the web cam with the photo in the database. If the photo is matched, the system will show "authentication succeeded"; otherwise, the system will show "authentication failed" and the user will be blocked from further steps.

6.5 Gesture Detection

After the user has been authenticated, the system will assign random gestures (See figure 6.5) for users to perform to avoid photo cheating. In figure 6.5, the user is required to blink 4 times in order to pass the test.

6.6 Take Photo

The registration process require user to take a photo (See figure 6.6) after the gesture has been verified. The photo will be stored in the database for future login.

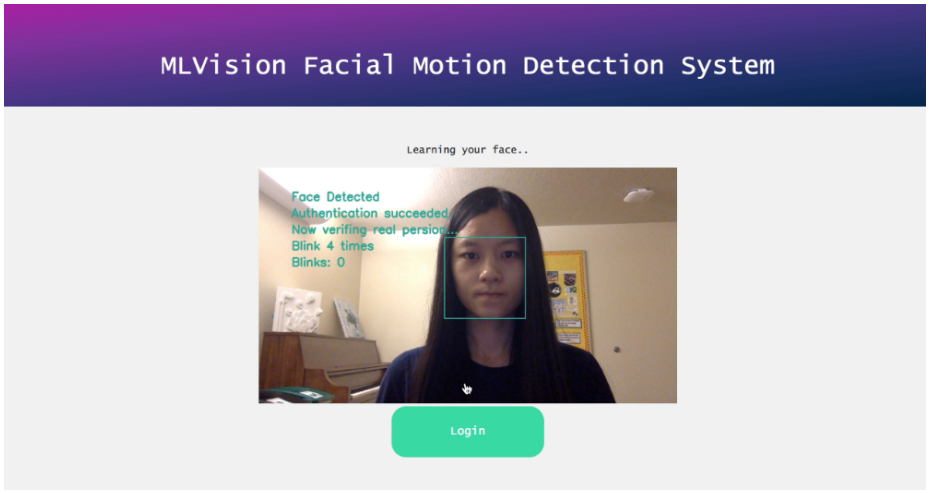

Figure 6.4: Facial recognition conceptual model

6.7 Personal Page

After the user has been both authenticated and verified as a real person, the system will direct the user to his or her personal page (See figure 6.7). If the user presses "Logout", it will go back to the main page.

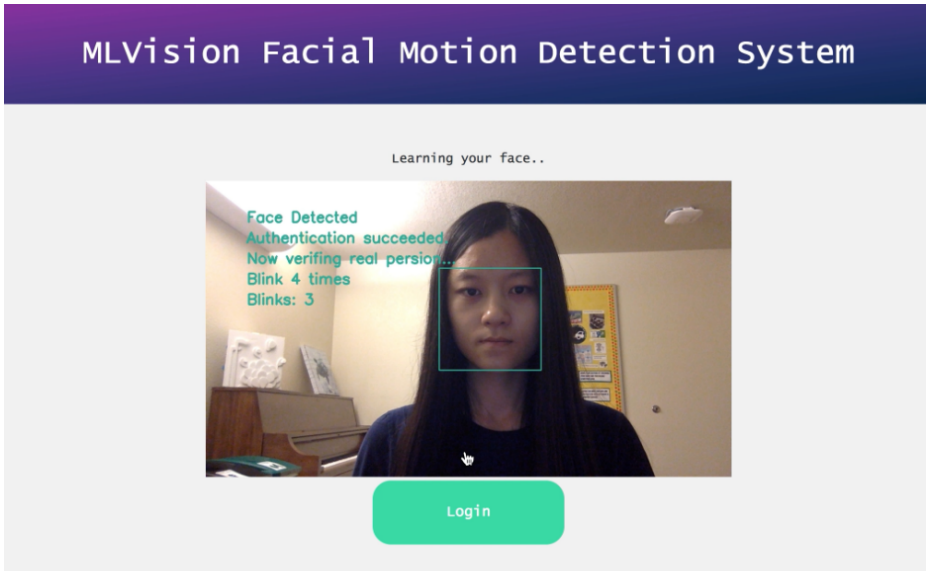

Figure 6.5: Gesture Detection conceptual model

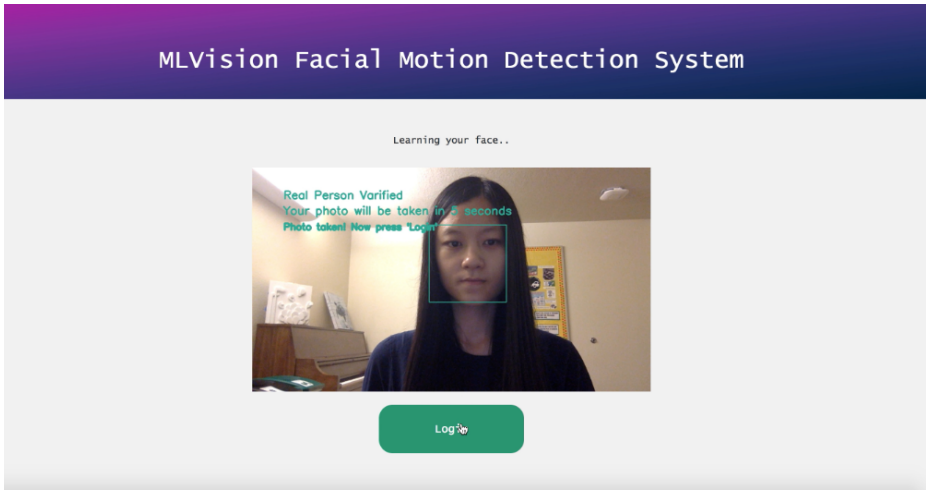

Figure 6.6: Take photo conceptual model

Figure 6.7: Personal page conceptual model

Chapter 7

Technology Used

7.1 Programming Languages

- HTML5

HTML5 is used to create documents on the web page. It defines the structure and layout of a Web document by using a variety of tags and attributes.

- CSS3

CSS3 is used to describe the presentation of Web pages. It also makes the web page responsive to different devices.

- JavaScript

JavaScript is used as a client side scripting language. Its code is written into an HTML page. When a user requests an HTML page, the script is sent to the browser.

- Python

Python is used for back end operations such as training machine learning models and retrieve information using the Facial Recognition API.

- SQL

SQL is used to communicate with a database. As the standard language for relational database management systems, SQL statements are used to perform tasks such as update data on a database, or retrieve data from a database.

7.2 Python libraries

- Scikit Learn

Scikit Learn is a simple and efficient tool for data mining and data analysis.

7.3 Applications

- SQLlite

SQLlite is a self-contained, high-reliability, embedded, full-featured, public-domain, SQL database engine. It is used to store both account information and pictures of the users.

- API technologies

API technologies provide access of data stored in the database to the public.

- Flask

Flask is a python Web frameworks, which supports the development of the web API.

- Github

Github is used for version control of the project. It allows programmers to update, revise, or restore a history version of the project. It also facilitates team collaboration on a project.

Chapter 8

Design Rationale

8.1 Facial Recognition API

We opted to use a deep neural network for the machine learning part of facial recognition because it is the most appropriate model to solve unconstrained classification problems, especially when distinguishing class characteristics aren't known. furthermore, we were able to find an open source and free to use software package for facial recognition that used an neural network that had already been trained on a large dataset, so we didn't have to build and train our own.

8.2 Labeled Faces in the Wild

The face recognition API we use scored 99.38% on Labeled Faces in the Wild benchmark (a database of face photographs for studying unconstrained face recognition). The machine learning model was trained on a wider and more diverse image set than we could have prepared ourselves.

8.3 Accuracy Threshold

The accuracy rating suggests that we should expect 99% of each frame to correctly indicate whether the face matches the correct user. Thats still not good enough, especially if we want to speed up the gesture recognition, since a single false detection restarts the gesture process. To reduce the risk of this annoying users, weve lowered the threshold for matches to shift errors towards false positives. the API is still good enough at detecting true negatives among all the video frames, so imposters can't access another's account.

Chapter 9

Test Procedure

9.1 Alpha

9.1.1 White Box

For the white box testing, we tested each portion of the code to ensure that they function as intended. We designed a series of varied test cases to use on the system to verify that all of the constraints are met in any situation. The test cases asserted the system's ability to maintain its functions in standard use situations, as well as potential edge cases.

- Login and Registration

We tested regular login and registration function. We first tested to register with a conflict username, and the system showed a warning. We then login with username and incorrect password, and the system also provided a warning.

- Facial detection and recognition

We tested our facial detection and recognition part using different faces. The system was able to detect all faces appeared on the screen and successfully authenticated the correct user.

- Gesture recognition

We tested gesture recognition by performing requested gestures assigned by the system. The system successfully recognized the gestures.

- Image impersonation

The last test we did is to test image impersonation. Face recognition test can be passed by using photos if the camera is not good enough to detect image depth. Our design solves this problem because it requires the person to perform gestures. We tested that although a different person holding the picture can pass the face recognition part, the gesture testing cannot be passed.

9.1.2 Black Box

Black box testing enlists the help of individuals not involved with the development of our application, which include classmates and friends. They understand what the system is designed to do, but not how it is done. The testers then used the system to sign up and login, as if they are real users of the system.

9.2 Beta

Finally, our beta test simulates real life scenario, apply the login system to a larger applications and people. While the program is mostly tested and confirmed to work as intended by this point, we will continue to monitor its performance as there is always the possibility of new issues appearing during live use over an extended period of time.

Chapter 10

Obstacles Encountered

Table 10.1: Obstacle table

Obstacles	**Consequences**	**Solution**
Delayed schedule due to heavy course load	Less time spent on the project	Caught up the progress during holidays and other quarters
Long response turnaround time	Some fast facial motions are hard to be detected	Increase video frame processing rate if possible, if not, perform the gesture slowly

There are two main obstacles encountered when we were doing our design. First is the delayed schedule due to heavy course load. Since in the winter quarter, all of our team members were very busy, we spent not much time on the project. But later we caught up the progress during holidays and other quarters. The second obstacle is frame rate limitation. Since the camera we use cannot detect fast motions due to video frame processing rate and CPU limitation, some fast facial motions are hard to be detected, so we had to perform the gesture slowly which takes more time, or increase video frame processing rate if we had a better device.

Chapter 11

Development Timeline

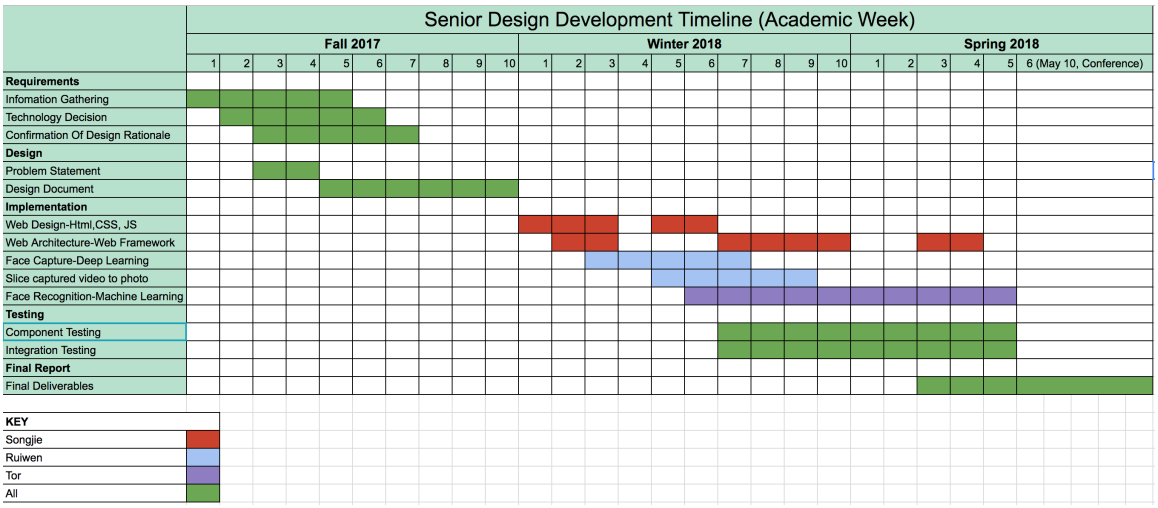

Figure 11.1: Development Timeline

The Development Timeline (figure 11.1) shows the project timeline in three quarter, each quarter in ten weeks. It includes tasks and estimated duration by weeks. The overall progress of the project is represented in the timeline. We finished different tasks each week according to the timeline. It makes our project organized and well-panned. In the timeline, each color represents a team member's contribution, purple is Tor Saxberg's contribution, blue is Ruiwen Li's contribution, and red is Songjie Cai's contribution.

Chapter 12

Future Plan

- Recognize more gestures

We plan to add more gestures into our system such as waving hands and opening mouth. The increasing number of random gestures will make the user harder to predict the move, and thus improve the security of the system.

- Test with more people and a larger image set

We plan to test with more people and get a more precise result of how accurate our system is on recognizing users' faces and gestures. We will also improve the accuracy of the system by training and testing with a larger image set.

- Store and encrypt face data in a separate database

The photos of the users in the database have a potential security issue. To solve that, we plan to store and encrypt the photos of the user in a separate database so that even if the database is being hacked, the photos will not leak.

- Replace users photo with face encoding

To bring the security of our system to the next level, we could only store the characteristics of the user's face in the database instead of the photo. Even if the face encoding data is leaked, the hacker cannot recover user's face.

- Remove the need for password in our implementation

When our system reaches really high accuracy, we could remove process of entering the password during login. User's account can be verified with facial recognition as the single security check.

Chapter 13

Conclusion and Lessons Learned

13.1 Modern Face Recognition

in the course of this project, we have learned that there has been much progress in the field of machine learning, deep neural networks, and face detection.

The first real time face detection algorithm was the ViolaJones object detection framework, which looked for light and dark areas of an image and compared them to the general light/dark arrangement of a face. Nowadays, we use a histogram of oriented gradients to detect faces in an image to reduced processing consumption.

face landmark detection was made possible by advances in deep neural networks to be able to detect the location of facial features, like nose and mouth. this makes it possible to linearly alter the face to make face recognition more reliable by improve the initial neural network training and reducing variety in the images the network has to classify. in this way, face recognition has become as simple as applying a support vector machine to classify data vectors.

13.2 API's and Open Source Code

the abundance of examples on sites such as GitHub and Stack Overflow have made it much easier to learn new programming techniques and tools. we were able to solve most of the issues we encountered developing this project by simply googling what we needed to know about. as time passes, these code hubs will only continue to grow, increasing the breadth of solutions that may save other programmers time.

we also found that well refined tools are now available that make it much easier to implement solutions without delving into low-level code like building a neural network from scratch. the Facial Recognition API we found on GitHub essentially removed the need to design or train our own network, so we could focus on the design and implementation of the larger aspects of our project like the gesture base, recognition speed, login procedure, and database integration in our site.

13.3 Conclusion

Since the face recognition algorithm we use is more than 99% accurate, the site only needs to compare the user's face to one stored image, rather than against a million different faces like Facebook, and our implementation checks the user's identity over several frames, we believe that face recognition is a secure, reliable, and simple method of authentication. in the future, there may not even be need for passwords or even captcha checks to stop bots.

if we separated the database of face information from the site's data and trained our own neural network, our implementation would be much more secure. just like signing in with your Google account doesn't give your password away to the whatever site, logging in with your face shouldn't give any site access to the associated data either.

lastly, if we improved the set of gestures our site can detect, we could speed up the process substantially compared to only detecting blinks. making the authentication process quick and seamless requires making the gestures simple and unobtrusive, which we have found to be the largest hurdle we would need to solve for our design to be competitive.

Chapter 14

User Manual

14.1 Registration

1. click "Create Account" button in the main page
2. Enter a username, click "Continue"
3. Enter a password, click "Continue"
4. Enable computer camera
5. Once the face is detected, perform the gesture assigned by the system
6. Stay still and take a photo after five second
7. click "Login"

14.2 Login

1. click "Login" button in the main page
2. Enter the username, click "Continue"
3. Enter the password, click "Continue"
4. Enable computer camera
5. Once the face is detected and the user is authenticated, perform the gesture assigned by the system
6. click "Login"