**Attendance system based on face recognition**

**Detailed development plan**

Project Name: Attendance system based on face recognition

Project Unit: Group 9

Project Time: 2019.2.28 -2019.5.10

Version Number: 2.0

**Content**

[1. Introduction 3](#_Toc5655580)

[1.1 Document purpose 3](#_Toc5655581)

[1.2 Background 3](#_Toc5655582)

[1.3 References 3](#_Toc5655583)

[2. Business architecture 4](#_Toc5655584)

[3. Module design 5](#_Toc5655585)

[3.1 System structure design 5](#_Toc5655586)

[3.2 Functional design description 6](#_Toc5655587)

[3.2.1 User APP module 6](#_Toc5655588)

[3.2.2 Management system module 10](#_Toc5655589)

[3.2.3 Face recognition algorithm module 12](#_Toc5655590)

[4. support conditions 18](#_Toc5655591)

[4.1 Hardware Support 18](#_Toc5655592)

[4.2 Software Support 18](#_Toc5655593)

[5. Schedule 19](#_Toc5655594)

[5.1 Overall schedule 19](#_Toc5655595)

[5.2 Learning schedule 19](#_Toc5655596)

[5.3 Development schedule 20](#_Toc5655597)

[6. Risk assessment and solution 22](#_Toc5655598)

[6.1 PLAN A 22](#_Toc5655599)

[6.2 PLAN B 23](#_Toc5655600)

# 1. Introduction

## 1.1 Document purpose

In order to standardize the module design of the software, we provide a detailed description of the software functions, input and output, algorithm flow logic and design documentation. This document will also provide reference for the later stages of programming and software testing.

## 1.2 Background

(1) Software project name：FaceRecognitionAttendance-FRA system

(2) Project participant：

|  |  |  |
| --- | --- | --- |
| **Name** | **Position** | **Duties** |
| WanHongda | Member | Face recognition algorithm design, API writing and APP Implementation |
| ZhangJiaqing | Member | Information Management Interface and APP Design |
| ZhangWei | Leader | Server design and writing, Information Management writing and database construction |

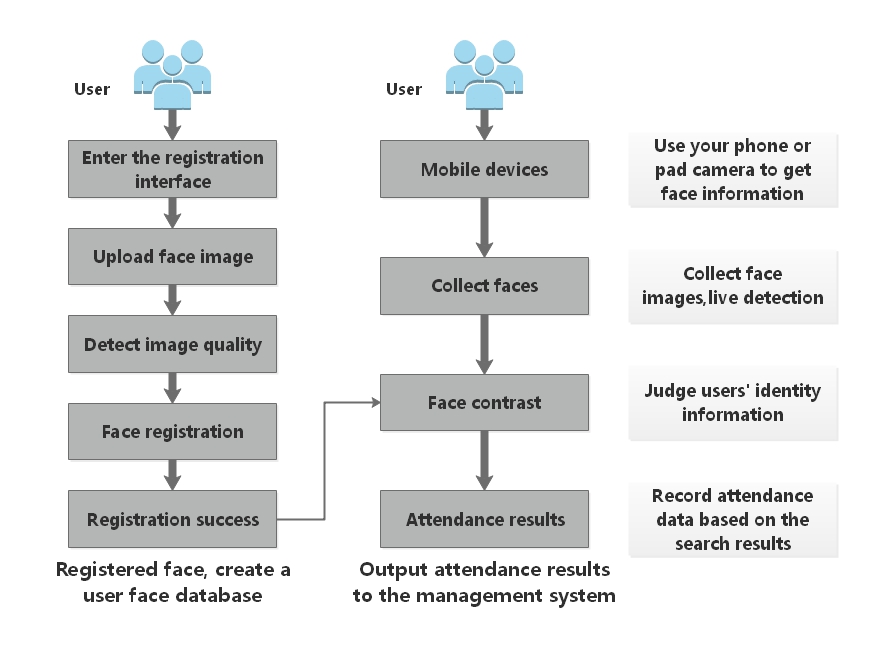
## 1.3 References

[1].T. Ojala, M. Pietikäinen, and D. Harwood (1996), "A Comparative Study of Texture Measures with Classification Based on Feature Distributions", Pattern Recognition, vol. 29, pp. 51-59.

[2].Ding C , Choi J , Tao D , et al. Multi-Directional Multi-Level Dual-Cross Patterns for Robust Face Recognition[J]. IEEE Transactions on Pattern Analysis & Machine Intelligence, 2014, 38(3):518-531.

# 2. Business architecture

The business architecture of software design is as follows:



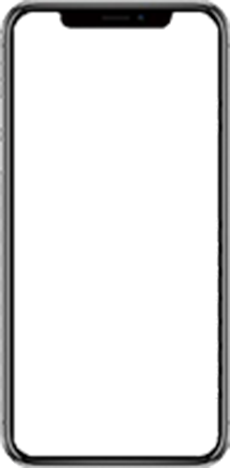
# 3. Module design

## C:\Users\Administrator\Desktop\软件项目管理ppt\结构.png3.1 System structure design

## 3.2 Functional design description

### 3.2.1 User APP module

#### 3.2.1.1 Design diagram



#### 3.2.1.2 Functional description

The main function of this module is to provide users with a friendly check-in and information registration guide. For the check-in function, the camera module of the terminal where the APP is located needs to be called to capture the image of the user's face and transmit it to the server; for the registration function, the user is provided with a form for filling in the basic information and collecting the image of the user's face. Transfer to server-side storage.

#### 3.2.1.3 Input data

(1) User check in

Get an image containing the user's face, the image format must be set to one of (jpg, jpeg, pgm or png). And the current time of the APP terminal Checkin\_time (String type).

(2) User registration

In addition to getting an image containing the user's face, you also need the basic information of the following users:

|  |  |  |
| --- | --- | --- |
| **Variable name** | **Type** | **Description** |
| Name | String | User's real name |
| Gender | String | User's gender |
| Group | String | User's organization or team |

#### 3.2.1.4 Output Data

(1) User check in

|  |  |
| --- | --- |
| **Field** | **Description** |
| UID | The ID assigned automatically by the server to the current user (cannot be changed once the assignment is successful) |
| Name | User's name |
| Gender | User's gender |
| Group | The organization or team the user is in |
| Checkin\_time | The time of the user checked in at the APP terminal |

(2) User registration

|  |  |
| --- | --- |
| **Field** | **Description** |
| Code | Registration result:  0 means failure,  1 means success. |

#### 3.2.1.5 Detail design

In order to provide users with a friendly interface, it is necessary to optimize the display interface of some APPs.

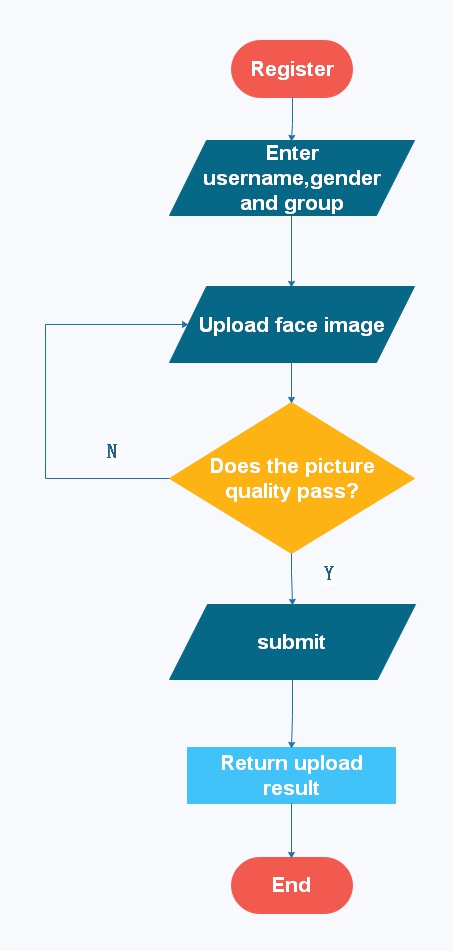
(1) User check in

The software needs to provide a collection window for the user to guide the user to provide facial key information better and more convenient. At the same time, the check-in result will be returned to the user in the form of a pop-up window, as shown in the figure:

(2) User registration

In addition to providing the user with a face collection window, the software will provide a form for the user after the face data is successfully collected, to help the user fill in the registration information, and after the user clicks the submit button, according to whether the relevant information is successfully stored in the server database Whether the user successfully registered, and then pop-up window gives user feedback, as shown:

3.2.1.6 Flow chart

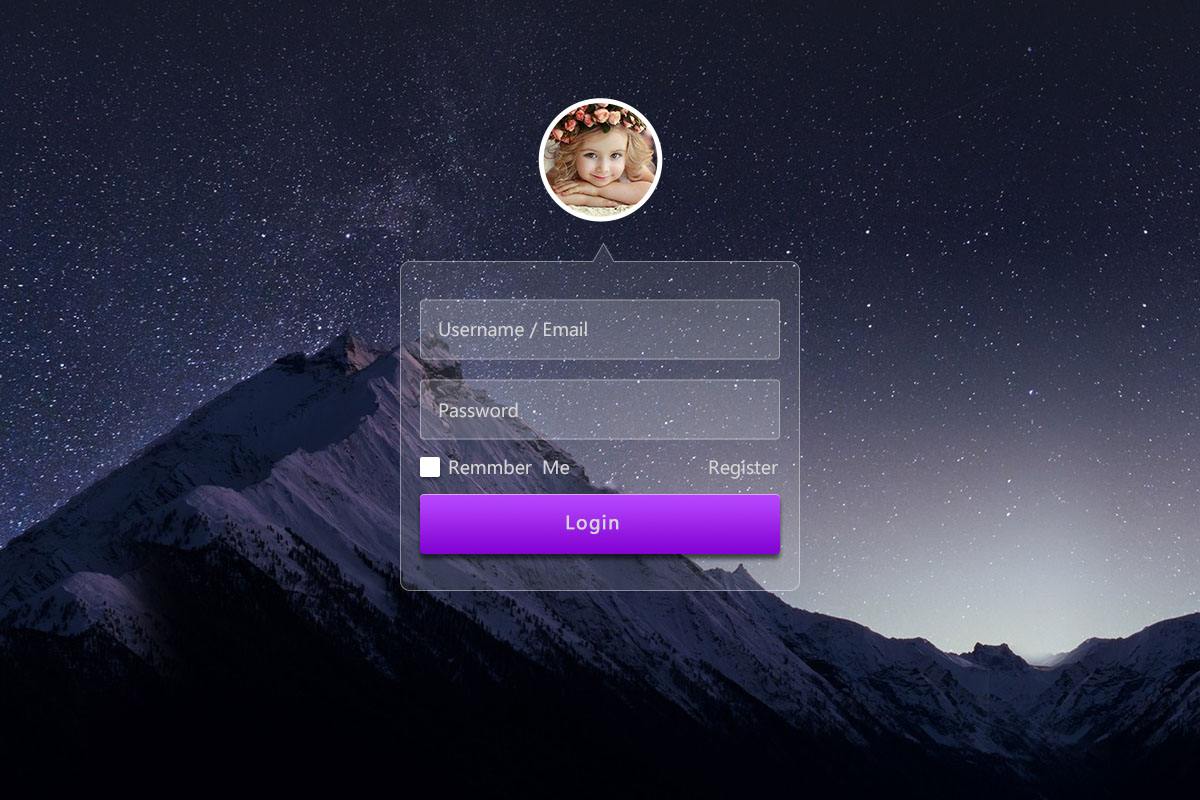
(1) User registration

#### 3.2.1.7 Limitation factor

All functions in the APP need to be used under normal network conditions, and the time required for information verification is related to the network speed.

### 3.2.2 Management system module

#### 3.2.2.1 Design diagram

(1) Login module

(2) Management module

#### 3.2.2.2 Functional description

The system management module provides administrators with a visual and friendly interface for administrator management. This module can be divided into two sub-modules, namely the login module and the management module.

#### 3.2.2.3 Input data

(1) Login module

|  |  |  |
| --- | --- | --- |
| **Variable name** | **Type** | **Description** |
| Admin\_name | String | Administrator's username (Required) |
| Password | String | Password (Required) |

(2) Management module

|  |  |  |
| --- | --- | --- |
| **Variable name** | **Type** | **Description** |
| - | - | - |

#### 3.2.2.4 Output Data

(1) Login module

After the registration module obtains the padding information, if the verification succeeds, it directly jumps to the management module interface, and if it fails, it gives the corresponding popup prompt, so the module does not have output data.

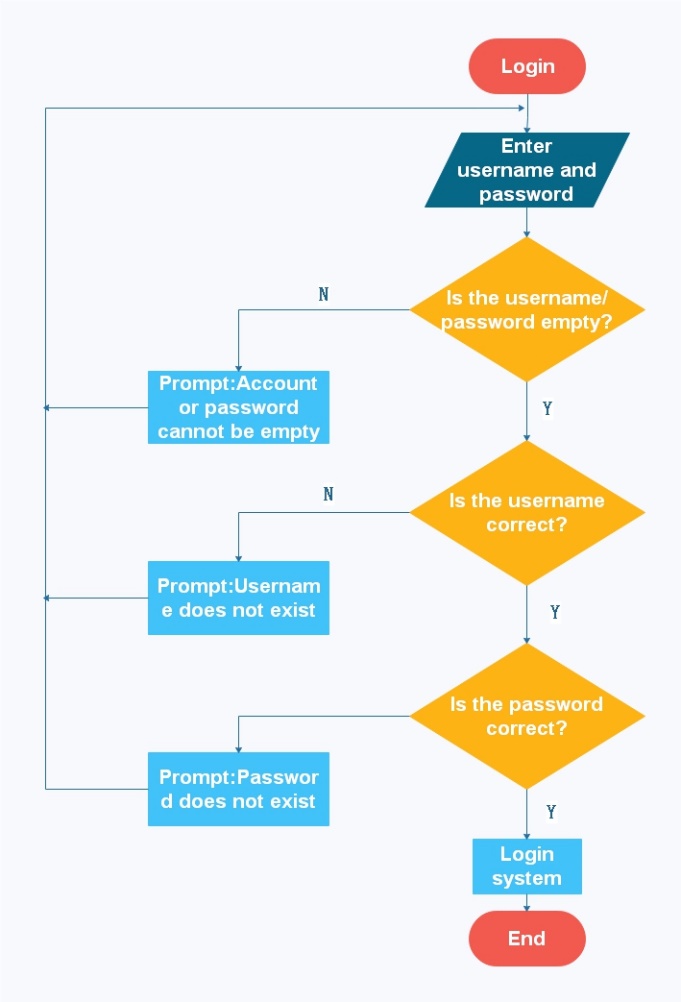
(2) Management module

|  |  |  |
| --- | --- | --- |
| **Variable name** | **Type** | **Description** |
| UID | String | - |
| Name | String | User's real name |
| Gender | int | User's gender |
| Group | String | User's organization or team |
| Checkin\_time | Date | - |

#### 3.2.2.5 Detail design

The management module needs to implement basic CRUD operations and communicate with the server-side database.

#### 3.2.2.6 Flow chart

Login module flow chart：

### 3.2.3 Face recognition algorithm module

#### 3.2.3.1 Functional description

The main function of this module is to provide a complete set of face recognition comparison functions. This module can be divided into two sub-modules, namely the face feature extraction module and the face comparison module.

#### 3.2.3.2 Algorithm detail design

The principle of LBP algorithm used in this module comes from the papers of T. Ojala, M. Pietikäinen, D. Harwood[1] and DCP algorithm comes from the papers of Ding C , Choi J , Tao D , et al.[2](only choose one of the two). The algorithm has a good precision and recall rate after comparison on multiple databases. Certain modifications are applied to the software project. The algorithm extracts and generates 256-dimensional feature vectors(or 512) from the pre-processed face images, and the module stores the feature vectors and the automatically assigned tags in the local xml file(or in the database). When the face features are compared, the module program reads the feature vectors stored in the xml file(or the database) and sequentially calculates the distance between the vectors (the distance calculation algorithm uses one of the Euclidean distance, the chi-square distance or the L1 distance), and the distance is the smallest. If the threshold is less than the preset threshold, it is the face owner, and the output returns the corresponding label.

#### 3.2.3.3 Input data

(1) Face feature extraction

|  |  |  |
| --- | --- | --- |
| **Variable name** | **Type** | **Description** |
| Face\_image | cv::Mat | User's face image |

(2) Face contrast

|  |  |  |
| --- | --- | --- |
| **Variable name** | **Type** | **Description** |
| Face\_image | cv::Mat | Pending detection face image |

#### 3.2.3.4 Output Data

(1) Face feature extraction

|  |  |  |
| --- | --- | --- |
| **Variable name** | **Type** | **Description** |
| Feature | Mat | The facial features of the user extracted by the algorithm |

(2) Face contrast

|  |  |  |
| --- | --- | --- |
| **Variable name** | **Type** | **Description** |
| Tag | int | The identified face owner's label, where -1 means the detected face does not exist in the database |

#### 3.2.3.5 Interface

Developers can create a new faceDetection by referring to the following code:

1. #include <iostream>
2. FaceDetection faceDetection();

Face recognition: Identify the face owner in the picture and return its identity tag:

1. #include <opencv2/opencv.hpp>
2. //Set the default recommended value for Rin/Rex/Grid\_x/Grid\_y
3. cv::Mat image\_test = "test\_train.jpg";
4. cv::Mat image\_predict = "test\_predict.jpg";
5. std::**int** Rin = 1;
6. std::**int** Rex = 4;
7. std::**int** Grid\_x = 1;
8. std::**int** Grid\_y = 1;
9. std::**float** Threshold = 0.85;
10. //Directly call face recognition
11. faceDetection.train(image\_test);
12. std::**int** tag = faceDetection.predict(image\_predict);
14. //If there are optional parameters
15. std::map<std::**int**, std::**int**, std::**int**, std::**int**, std::**float**> options;
16. options["Rin"] = Rin;
17. options["Rex"] = Rex;
18. options["grid\_x"] = Grid\_x;
19. options["grid\_y"] = Grid\_y;
20. options["Threshold"] = Threshold;
22. //Call face recognition with parameters
23. faceDetection.train(image\_test, options);
24. std::**int** tag = faceDetection.predict(image\_predict, options);

**(1)** **Face feature extraction: used to extract feature information of a face in a specified image.**

**Face feature extraction——Request parameter details**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Parameter name** | **Whether it is necessary** | **Type** | **Defaults** | **Description** |
| **Face\_image** | **Yes** | **cv::Mat** | **Null** | Image information containing the face (Data size should be less than 10M), the image format should be one of jpg, jpeg, pgm or png(Recommended jpg format) |
| **Rin** | **No** | **std::int** | **1** | **The inner radius of the characteristic sampling point range (No special case is not recommended)** |
| **Rex** | **No** | **std::int** | **4** | **The outer radius of the characteristic sampling point range (No special case is not recommended)** |
| **Grid\_x** | **No** | **std::int** | **1** | **The number of horizontal segmented regions of the feature image (Modifying to a larger value can improve recognition accuracy, but speed will decrease)** |
| **Grid\_y** | **No** | **std::int** | **1** | **The number of vertical segmentation areas of the feature image (Modifying to a larger value can improve recognition accuracy, but speed will decrease)** |

**Face feature extraction——Return parameter details**

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Whether it is necessary** | **Type** | **Description** |
| **Feature** | **Yes** | **Mat** | Extracted face feature vector information |

**(2) Face contrast: used to determine the identity of the face owner.**

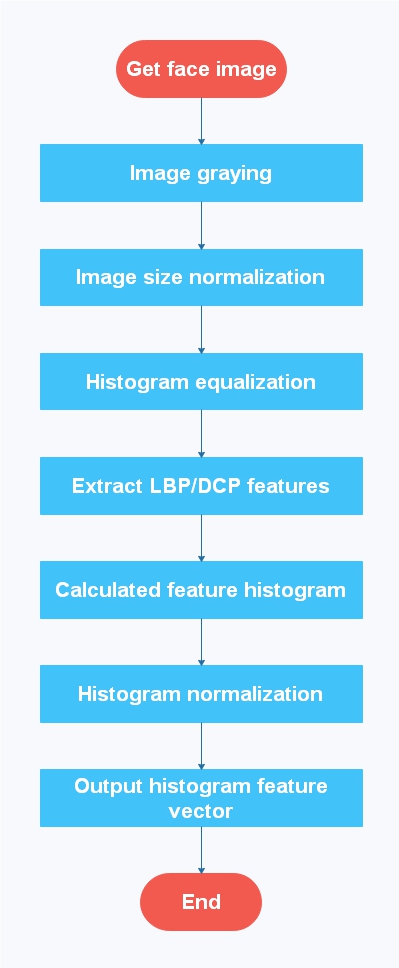
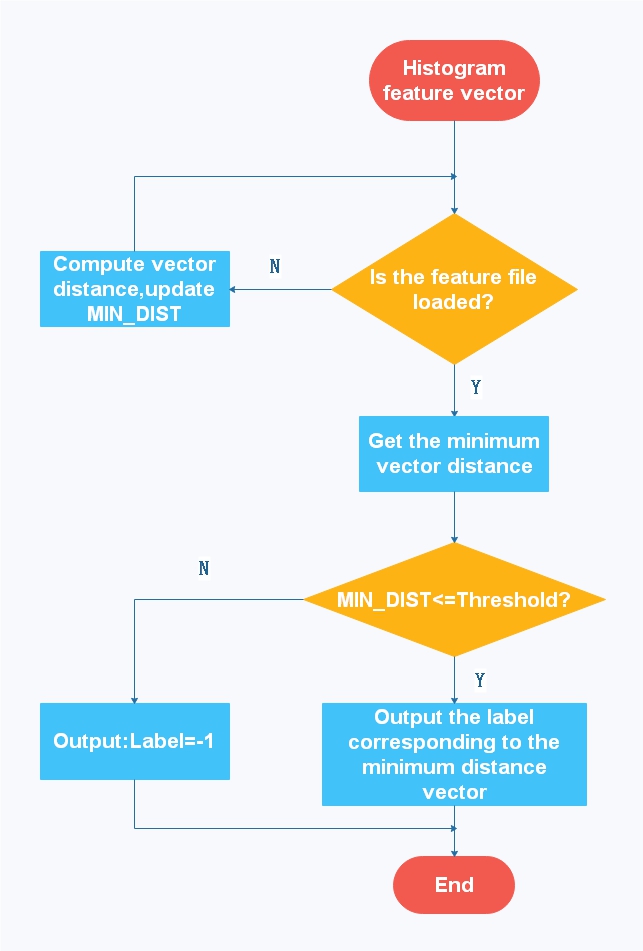
**Face contrast——Request parameter details**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Parameter name** | **Whether it is necessary** | **Type** | **Defaults** | **Description** |
| **Face\_image** | **Yes** | **cv::Mat** | **Null** | Pending detection image(Data size should be less than 10M), the image format should be one of jpg, jpeg, pgm or png(Recommended jpg format) |
| **Threshold** | **No** | **std::float** | **0.85** | **Threshold, either by yourself or by default** |

**Face feature extraction——Return parameter details**

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Whether it is necessary** | **Type** | **Description** |
| Tag | **Yes** | **int** | The identified face owner's tag,  where -1 means the detected face does not exist in the database |

#### 3.2.3.6 Algorithm flow

(1) Face feature extraction && Face contrast

#### 3.2.3.7 Compiler Environment

This part of the algorithm is compiled and implemented in Microsoft Visual Studio 2010 and above, and requires an extension module of OpenCV3.0 and above.

# 4. support conditions

## 4.1 Hardware Support

(1) A terminal capable of running a face sign-in program requires a memory of 512 MB or more, a storage space of 4 GB or more, and a front camera that can operate normally.

(2) One server, Intel® XEON® Cascade Lake baseband/core frequency: 2.6GHz/3.5GHz, 2 core 4G, bandwidth 6Gbps, 100G hard disk storage.

(3) A terminal capable of running an information management interface needs to have more than 2G of memory, 128G or more of storage, 100M of broadband, and a LAN using TCP/IP protocol.

## 4.2 Software Support

The required software support varies according to the different requirements of the module.

(1) OpenCV needs to control the version above 3.0.

(2) Support CX11 when configuring.

(3) Node.js software service,

(4) Apache software services,

(5) MySql database software service,

(6) Operating system of Windows 7 and above.

(7). The terminal operating system needs to select Android system, the version is between Android 5.0~8.1.

(8) Development tools: WebStorm 11, HBuilder, Xshell5, FlashFXP, Navicat Premium, Visual Studio 2010 and above version.

# 5. Schedule

## 5.1 Overall schedule

|  |  |
| --- | --- |
| **Stage** | **Time required** |
| Design | 1-3 weeks |
| Learn | 4-5 weeks |
| Develop | 6-8 weeks |
| Test&Debug | 9-10 weeks |
| Release | 11 weeks |

## 5.2 Learning schedule

|  |  |  |  |
| --- | --- | --- | --- |
| **Learning topic** | **Principal** | **Study-time** | **Deliverables** |
| Algorithm | WanHongda | 2 weeks | Extract feature vectors, Recognize faces. |
| Front-end | ZhangJiaqing | 2 weeks | Review knowledge that solves business requirements |
| Node.js | ZhangWei | 1.5 weeks | Build servers, Business request and response |
| APP | ZhangWei | 0.5 weeks | Android App development and packaging |

## 5.3 Development schedule

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Module** | **Submodule** | **Principal** | **Development time** | **Testing time** | **Test leader** |
| **Login module** | Login interface | ZhangJiaqing | 3 days | 1 days after completion | WanHongda |
| Verification request | ZhangWei | 1 days after completion | ZhangJiaqing |
| **Management interface module** | Management interface | ZhangJiaqing | 3 days | 1 days after completion | WanHongda |
| CRUD interaction | ZhangWei | 1 days after completion | ZhangJiaqing |
| **App face registration** | Registration interface | WanHongda | 5 days | 2 days after completion | ZhangJiaqing |
| Picture transfer function | ZhangWei | 1 days after completion | WanHongda |
| Information reception save and response | ZhangWei | 1 days after completion | ZhangJiaqing |
| **App check-in real time** | Check-in interface | WanHongda | 7 days | 2 days after completion | ZhangWei |
| Face capture and upload | ZhangWei | WanHongda |
| Face contrast return results | ZhangWei | WanHongda |
| **Algorithm module** | Feature extraction module | WanHongda | 7 days | 4 days after completion | ZhangWei |
| Face contrast module | WanHongda | 10 days | 6 days after completion | ZhangWei |

# 6. Risk assessment and solution

Since the algorithm program is written in C++ language, there may be certain risks in the background encapsulation such as functional failure and data loss, so it is necessary to prepare a risk solution to deal with the possible risks. To this end, we have prepared two solutions PLAN A and PLAN B.

## 6.1 PLAN A

Use the SDK provided by Baidu Company to replace the algorithm part. For details on Baidu SDK, please refer to the documentation: <http://ai.baidu.com/docs#/Face-Node-SDK/top>.



## 6.2 PLAN B

It turns out that it is realistic and feasible to write LBP or DCP algorithms using JAVA language. The algorithm program written in JAVA language can be slightly modified or even directly applied to the development of the client, so that the interface parameters of the LBP/DCP algorithm of the JAVA language can be appropriately simplified.

You can create a new FaceDetection with reference to the following code. After the initialization is complete, modify the corresponding parameters as needed. The program will return the identity tag of the face owner:

1. **public** **class** Sample{
2. //Set the output tag of the log
3. //Set the default recommended value for Rin/Rex/Grid\_x/Grid\_y
4. **private** **static** **final** String TAG = "Sample";
5. **private** **static** **int** Rin = 1;
6. **private** **static** **int** Rex = 4;
7. **private** **static** **int** Grid\_x = 1;
8. **private** **static** **int** Grid\_y = 1;
9. **private** **static** **float** Threshold = 0.85;
10. **public** **static** **void** main(String[] args) {
11. //Optional: Reset parameter values according to the developer's needs
12. //If not set, the default configuration is used.
13. **this**.Rin = 1;
14. **this**.Rex = 4;
15. **this**.Grid\_x = 1;
16. **this**.Grid\_y = 1;
17. **this**.Threshold = 0.85;
18. //Initialize a FaceDetection
19. FaceDetection faceDetection = **new** FaceDetection();
21. //Call interface
22. String trainPath = Environment.getExternalStoragePublicDirectory()+"/test\_train.jpg";
23. String predictPath =Environment.getExternalStoragePublicDirectory()+"/test\_predict.jpg";
24. faceDetection.train(trainPath, Rin, Rex, Grid\_x, Grid\_y);
25. **int** label = faceDetection.predict(predictPath, Rin, Rex, Grid\_x, Grid\_y, Threshold);
26. Log.d(TAG，label.toString());
27. }
28. }

**Face feature extraction——Request parameter details**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Parameter name** | **Whether it is necessary** | **Type** | **Defaults** | **Description** |
| **Face\_image\_path** | **Yes** | **String** | **Null** | Target image storage path (Data size should be less than 10M), the image format should be one of jpg, jpeg, pgm or png(Recommended jpg format) |
| **Rin** | **No** | **int** | **1** | **The inner radius of the characteristic sampling point range (This parameter is only used in the DCP algorithm and no special case is not recommended)** |
| **Rex** | **No** | **int** | **4** | **The outer radius of the characteristic sampling point range (This parameter is only used in the DCP algorithm and no special case is not recommended)** |
| **Grid\_x** | **No** | **int** | **1** | **The number of horizontal segmented regions of the feature image (Modifying to a larger value can improve recognition accuracy, but speed will decrease)** |
| **Grid\_y** | **No** | **int** | **1** | **The number of vertical segmentation areas of the feature image (Modifying to a larger value can improve recognition accuracy, but speed will decrease)** |

**Face feature extraction——Return parameter details**

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Whether it is necessary** | **Type** | **Description** |
| **Feature** | **Yes** | **String** | Character vector in string format(The original vector dimension is 1xN-dimensional and all data are separated by spaces) |

**(2) Face contrast: used to determine the identity of the face owner.**

**Face contras——Request parameter details**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Parameter name** | **Whether it is necessary** | **Type** | **Defaults** | **Description** |
| **Face\_image\_path** | **Yes** | **String** | **Null** | Pending detection image storage path (Data size should be less than 10M), the image format should be one of jpg, jpeg, pgm or png(Recommended jpg format) |
| **Threshold** | **No** | **float** | **0.85** | **Threshold, either by yourself or by default** |

**Face feature extraction——Return parameter details**

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Whether it is necessary** | **Type** | **Description** |
| Tag | **Yes** | **int** | The identified face owner's tag, where -1 means the detected face does not exist in the database |

It should be noted that to write algorithm programs in JAVA language, it is necessary to configure OpenCV for Android. The CX11 is not required to be configured, but if the customer wants better compatibility and the resulting installer volume is negligible, then it is recommended to configure.