SOFTWARE REQUIREMENTS SPECIFICATION

for

<Smart Park Supervisior>

Version 1.0 approved

Prepared by <Yunyan Hong & Shike Zhang
Renjie Gu & Ruohong Xu>

April 1, 2018

Contents

1 Introduction

1.1 Purpose

With the boom of the population and for the convenience of transportation, there exists a growing number of large-scale meetings and activities (e.g. G20). One prerequisite for a successful meeting is to guarantee the absolute security for these occasions considering the occurrence of anomaly-based intrusion.

The old method we use is to manually track video records and find out the activity of particular persons (e.g. lost children, criminals, suspects) which is time, recourse and labor exhausting. Therefore, it's necessary to make supervision for those important areas to grasp the real-time condition, as well as to recognize each person in the meetings. We propose a complete monitoring system for the large population in a park. The definition of park can include any important places that need to be monitored such as residential area, large square, or train stations during the Spring Festival.

1.2 System Overview

In our system, we need to install many HD cameras to form a camera network. Those cameras can clearly capture human faces in the scope, and to compare with the pictures in our database to figure out the person's ID. At the same time, this new pictures will be stored in the corresponding cluster of database, to extend our datasets. Furthermore, according to the different pictures in a time flow for one person, we can know the traveling path for him, thus we can keep track for any person if necessary. When the system identify faces belonging to high-risk population, it will alarm to a department to remind them of him.

Besides, to expand our system, we can bind the picture with the file of corresponding person, and to mark recidivists in the whole system. So when the occur again in any park, we can alarm immediately to maintain the security.

In conclusion, our purpose is to make a real time monitoring system to benefit our clients, policemen or supermarket. For example, to maintain security in their ranges, in which security does not only mean the security for one's life , but also no stealing and other terrible actions.

2 Presupposition

2.1 Requirement

Smart Park Supervisor provides users a powerful tool to administrate and report population mobility in the park or zone where the system is deployed.

2.1.1 Functional Requirement

First of all, it can help region administrators to record all people appear in the regulated zone so that administrators can easily have a command of the safety in this area. By the ways of snapshot all people flash in front of well arranged cameras, the system can recognize and distinguish the identity of passengers. Overall the system should support two types of people. The first case is for residents that live in this area whose dossiers are in the local police office. The other is for nonresidents that do not have any records before. Of course these two definitions are not absolute. Their meanings are flexible. For example, if the system is applied in a large company, residents represent employees and nonresident represent visitors.

Not only distinguishing whether the person appears in the camera is resident, for those that do not have any records before, the system should cluster all records that have not been recognized and retrieved, in order to classify and group all unrecognized records. In the end we hope the system can display all records in the form of dossiers. Some of dossiers have identity and some not.

Further more, the system should have the competence beyond the existing snapshot records. For example, from a person's records the system can track possible traveling of him. The route can be graphically visualized or return back to the system as another input for further analysis. Meanwhile, for some certain conditions for example, "hide by day and come out by night", the system can filter the dossiers that usually appear at night.

These features for analysis are helpful if the administrator wants to find out potential danger and threats. The system will supervise most of the people's movements and dangerous behaviors. On the other hand, it can also detect persons' whereabouts in certain area. More features should be added in different situations.

For further use, the system should be able to share data and connect to each other, if some other areas implement the system as well.

2.1.2 Non-functional Requirement

On one hand, since the cameras need to shoot the passengers and there is peak-hour, if the cameras(slave) keep connecting to the server(agent), it cost a lot of resources. As a result, the system should have the ability of buffering and sending data at regular time.

On the other hand, there are too many pictures taken in one day. What's more, due to the movement of people and accuracy of cameras, some pictures may be fuzzy and low-quality. The system should be able to filter low-quality pictures and clean outdated records in order to save database space.

Last but not least, the system owns information of face records, identity and more personal privacy, special security measurements are needed to make sure these information will not be revealed to the public.

2.2 Condition, Supposition and Limitation

- Minimum life time of system: 2 years
- Longest time that records will survive: 30 days
- Largest number of cameras: 5
- Our system is sponsored directly by software engineering department of Shanghai Jiao Tong University, both technically and financially.
- Conditions of developing/run-time environment in hardware and software

Hardware:

1. Intel I5 or more advanced PC, laptop
2. Minimum runtime memory requirements: 4 G
3. Hard disk space for installation: 100 G

Software:

1. Agent: Ubuntu 16.04
2. Slave: Ubuntu 16.04

2.3 Feasibility Analyzing Method

After finishing our system, we will make a case test to exam our system's feasibility.

1. First, we ask a person go pass our cameras. Then we enter his person file, to exam whether the picture just took is put into his person file.
2. Second, we check his record to check whether trace just now is his true path.
3. Third, we mark this guy as a high-risk person, and ask him to go pass the camera again, to test whether the alarm will work.

3 Proposed System Architecture

3.1 Introduction to the Proposed System

We prefer to use 3-tiers architecture to deploy our system and satisfy the Functional requirements. The brief physical architecture view can be showed as follow:

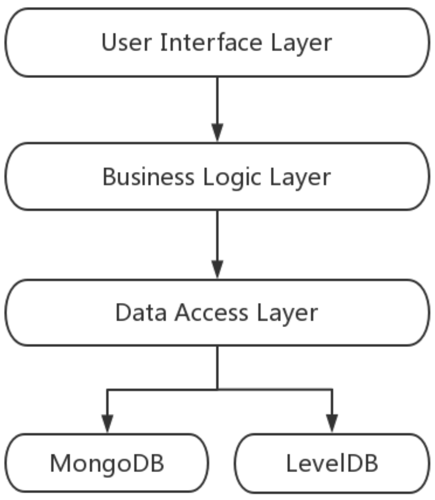


Figure 3.1: 3-tiers architecture

3.2 Brief Architecture

1. We will use two database to store two kinds of data. Firstly, MongoDB is used to store all snapshots taken by cameras. Cameras will take photos of passerbys and pick up face images from the scene image, and will send it back to agent workstation. The agent will store the face image and scene image into MongoDB. When users want to look up some pictures, the system will fetch them in MongoDB. Secondly, we nned a database to store the dossiers that users provide which mean resident. For each dossier, they will get a dossierId and its face feature and personal information will be stored in database. Considering the problem above, we choose LevelDB to store these information because LevelDB supports key-¿value lookup.

We often need to get personal information through someone's dossierId. As a result, LevelDB is the best choice.

2. The core service of our system is the Face Image Compare Service. We need to compare two face image and calculate their similarity. If the score is beyond threshold it means they represent the same guy and can be merged. Actually this service needs artificial intelligence and it is beyond our ability. So we will use the open-sourced algorithm on the internet to accomplish our purpose.

3. the Fllter Service means for images sent back by cameras, we need to estimate them and filter the low-quality ones in order to avoid useless calculation.

4. the Retrieve Service means for images that have passed filter service, we need to call Face Image Compare Service to calculate the similarity between these images and the dossiers in LevelDB. If it is shown that one image belongs to certain dossier, we will tag it and merge them.

5. The Cluster Service means for images that don't belong to any dossier, or we can call them passerbys, we need a service to classify them and mark the clustered dossier nonresident.

6. The Look-up Service is for User Interface. This service will accept the look-up command and search the database and respond. This service is important and contains several modules, for example, search engine, search service, http request handler etc.

7. User Interface is displayed to users. We will show the final result on a website so users will use browers to access our system. Meanwhile we will provide some api for query. For example, users can choose time, place or certain camera to search some guy, and wu will translate the query into sql or mongoDB sentenses and send it to Look-up Service.

From description above, we can conclude our system in the graph shown below:

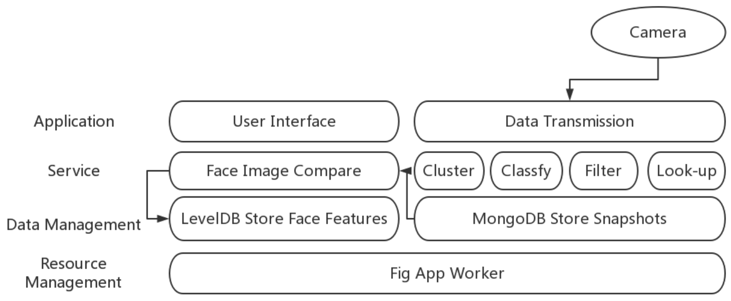


Figure 3.2: System Physical View

3.3 Use Case diagrams

We draw a use case diagrams to show the function of our system and operation authority for general users and administrators. As the picture tells, the administrators can add, delete, update dossier, which all related to the dossier database. While the users just display dossiers, face images and query the information in dossiers and learn the statistics. Besides, whether general users or administrators are all need to login to use our system to guarantee the security.

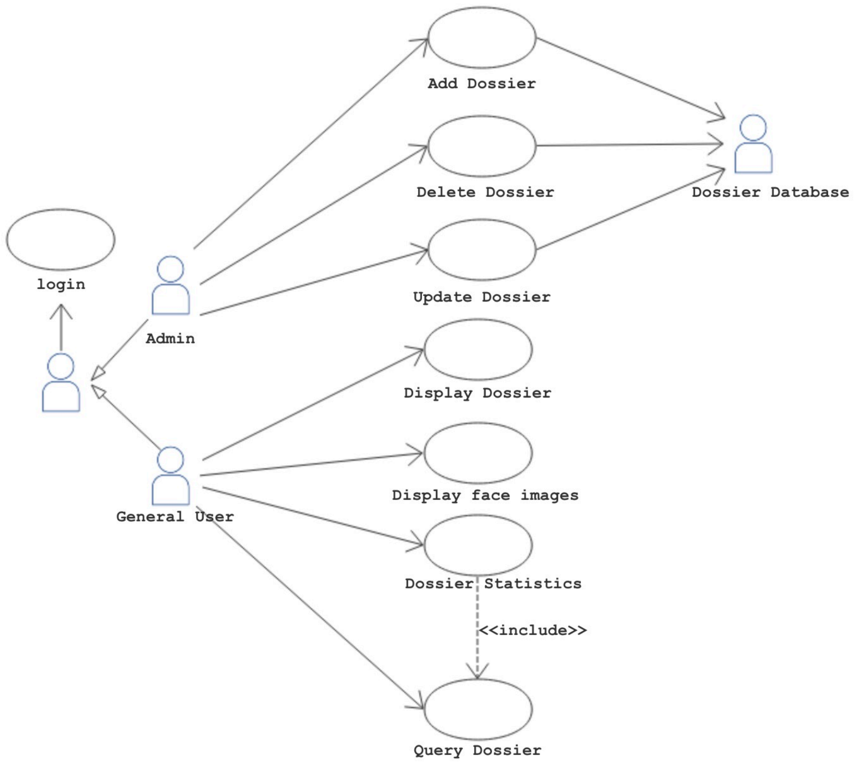


Figure 3.3: use case diagrams

3.4 Object diagrams

This part, we draw the object diagrams for our system. In our system, we design three services, including clusterServices, filterService and retrieveService. Among these three, the clusterServices provide the cluster function for the images which are not belong to one cluster in face image database. Besides, the retrieveService combine with filterService works in query function. The searchEngine is the main platform to accomplish our search function in face database and dossier database.

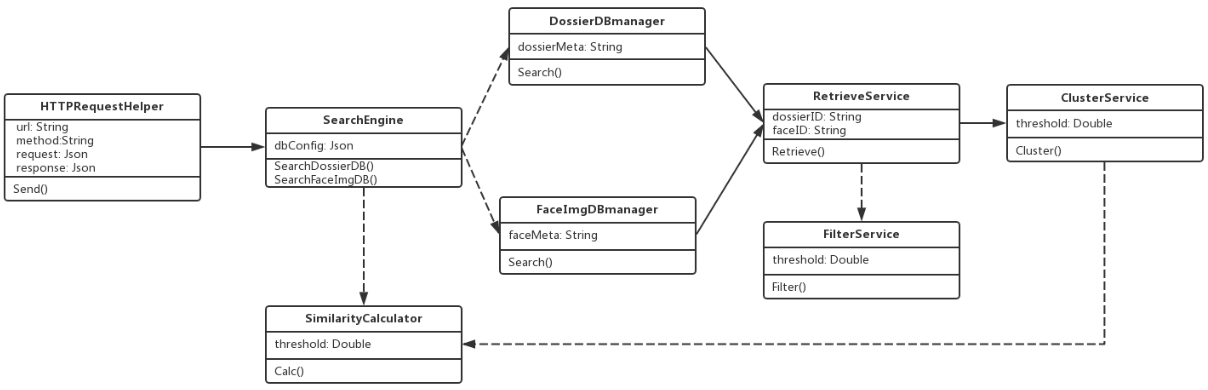


Figure 3.4: object diagrams

3.5 Sequence diagrams

This part, we draw two diagrams to explain our process in detail. The first one tell the whole process as a user for the process for two kinds of function query. The second tells the process after a new picture is taken, including the picture matched in the dossiers and not.

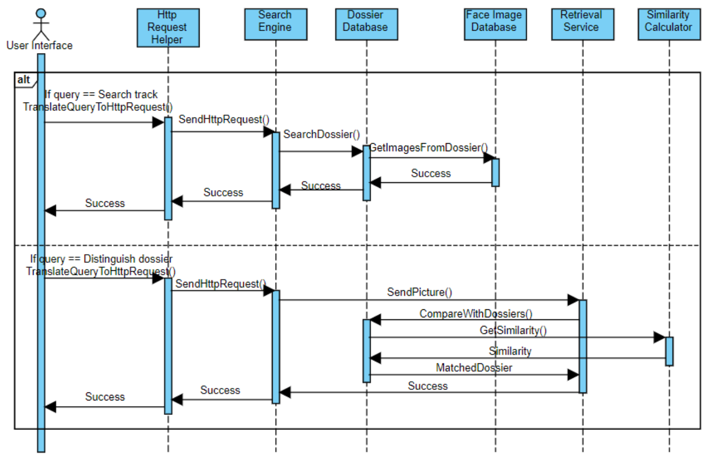


Figure 3.5: Sequence diagram 1

One is search with the information in dossier. For example, user can query person's track, route, pictures and other analysis with person's name. The other is just through a picture to recognize the corresponding person and return his messages.

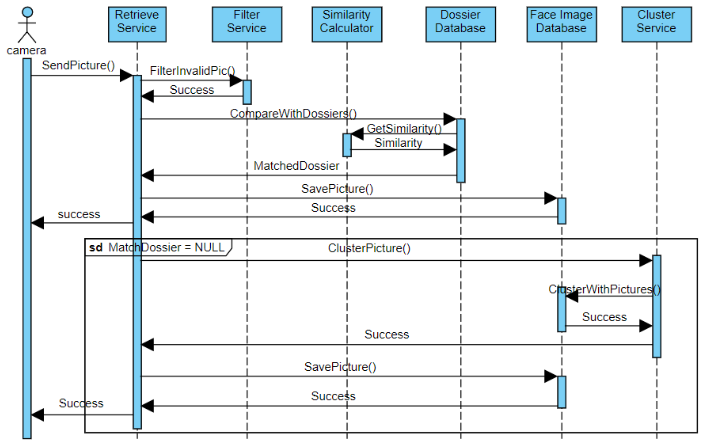
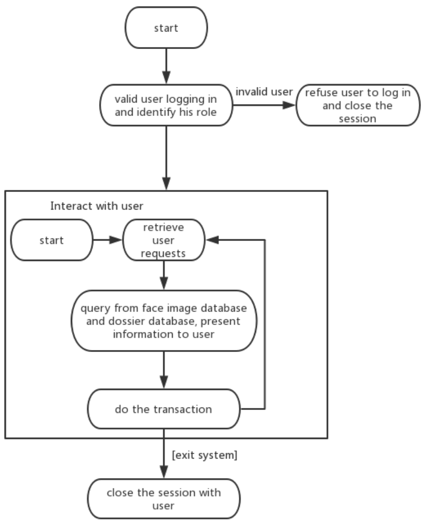


Figure 3.6: Sequence diagram 2

3.6 Workflow

The picture below is our workflow for the whole system, which tells the whole process in a user's view.



start
valid user logging in and identify his role
invalid user
refuse user to log in and close the session
Interact with user
start
retrieve user requests
query from face image database and dossier database, present information to user
do the transaction
[exit system]
close the session with user

Figure 3.7: System Physical View