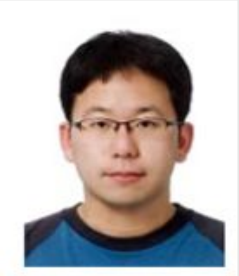
CERTIFIED 100% CERTIFIED

Phase 1 Presentation

Team 3

Team 3 - Member Introduction

								
Name	Kyuwoon Kim	Gyeonghun Ro	Wonyoung Chang	Soohyun Yi	Hyejin Oh	Hyungjin Choi	Vibhanshu Dhote	Cliff Huff
Role & Responsibility	PC App Development, **Requirements, Threat Modeling, Security** Implementation	Project Manager, Test case, Architect, **Requirements, Threat Modeling**	PC App/Embedded Development, Environment, **Requirements, Threat Modeling**	Embedded Development, Static Analysis, **Requirements, Threat Modeling**	Documentation, Security Standards, **Requirements, Threat Modeling,** Static Analysis	Embedded Development, Tool Research, **Requirements, Threat Modeling**	Vulnerability Research, **Requirements, Threat Modeling**	Mentor
LGE Email	kyuwoon.kim@lge.com	gyeonghun.ro@lge.com	wonyoung.jang@lge.com	soohyun.yi@lge.com	hyejin.oh@lge.com	jin0925.choi@lge.com	vibhanshu.dhote@lge.com	
Personal Email (CMU account)	mnik83@gmail.com	gyeonghun.ro@gmail.com	wonyoungjjang@gmail.com	randelx@gmail.com	chaolly007@gmail.com	zzzzdx@gmail.com	dhotevibhanshu@gmail.com	cch@sei.cmu.edu

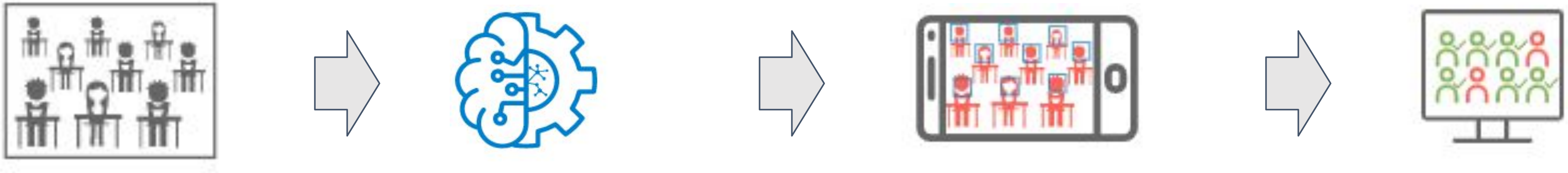
Alias Email of Team 3 :
cmu-team3@lge.com

Project Overview - Redefining for practical implementation

CERTIFIED 100% CERTIFIED

Carnegie Mellon University

[CMU Student Attendance Check System]



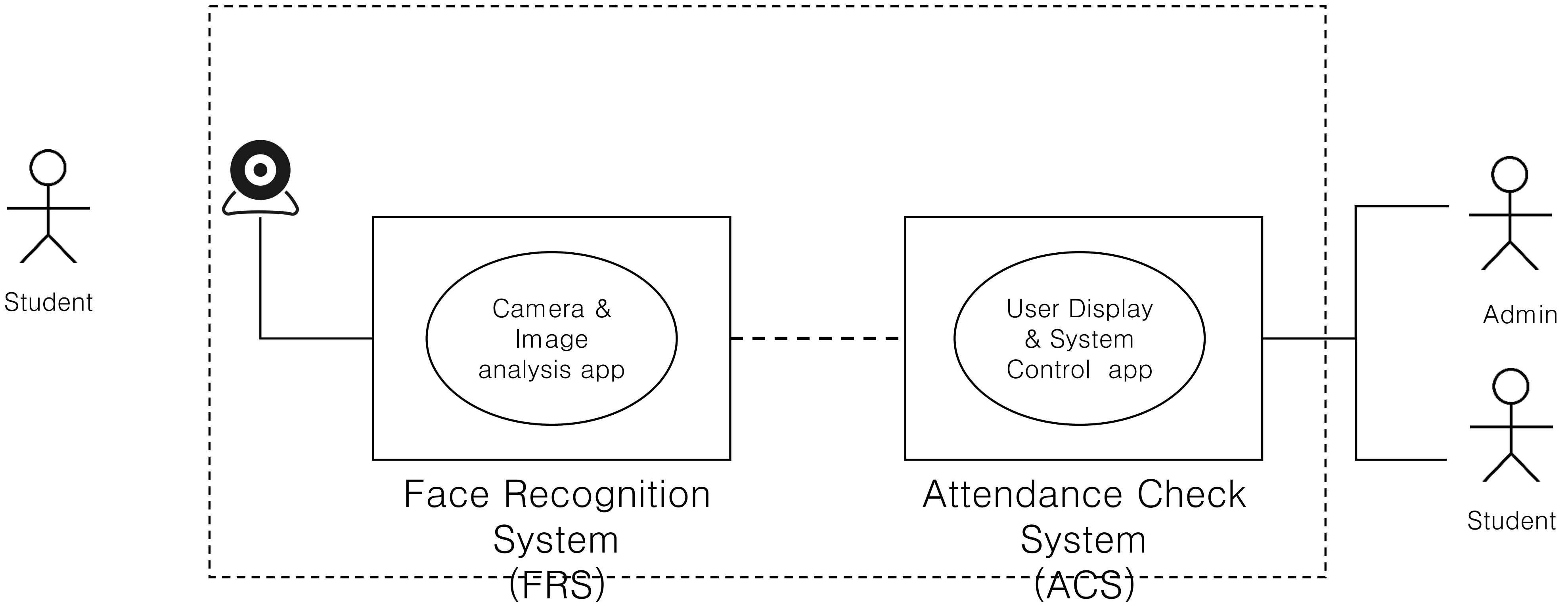
Students in the classroom

“100% recognition engine” find faces and recognize who they are.

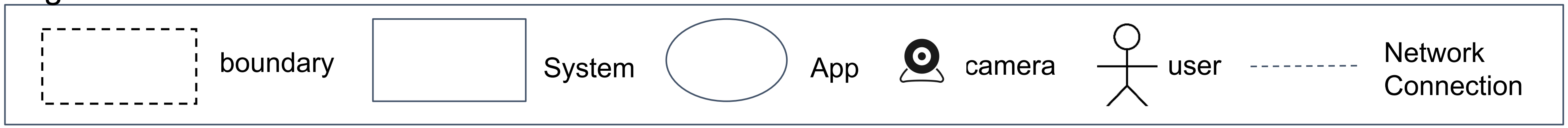
The face and name recognized by the system are displayed.

Student attendance is checked.

Context Diagram



legend



Security Goal

AACS Security Goal :

1. **Confidentiality** : The sensitive data(User DB, Video, Face data ...) in the system should be accessible only by the authorized people.
2. **Integrity** : The sensitive data in the system should not be tampered by an attacker.
3. **Availability** : The attendance system should be available at all times during the semester.

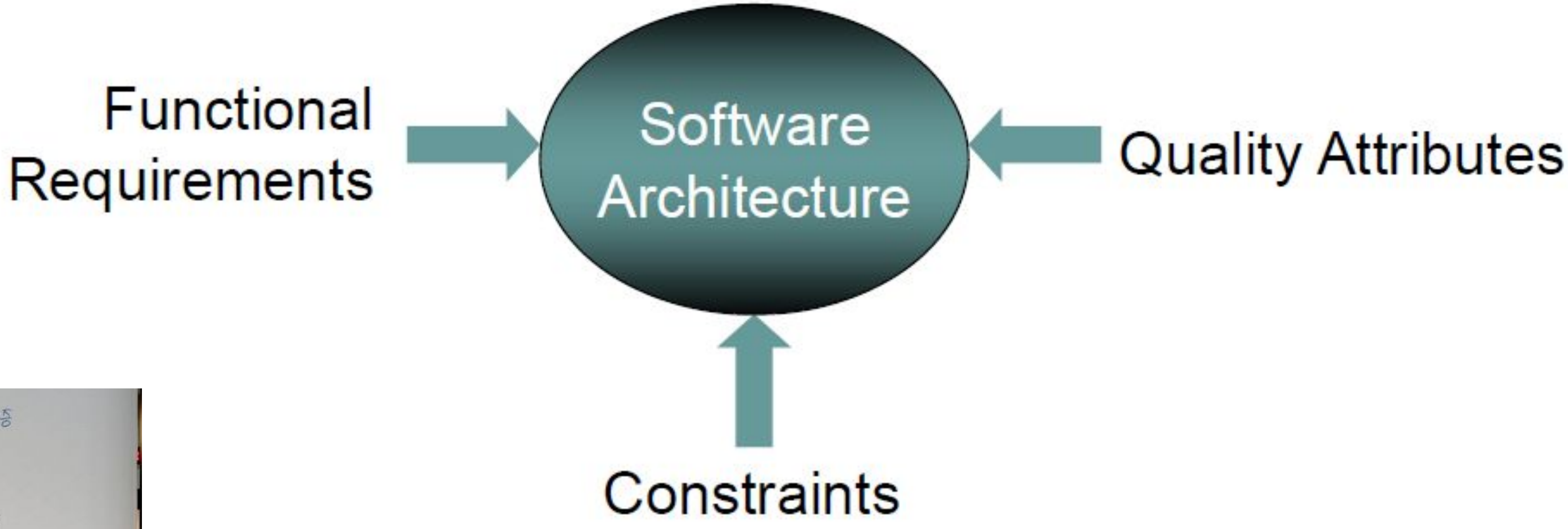
Project Overview

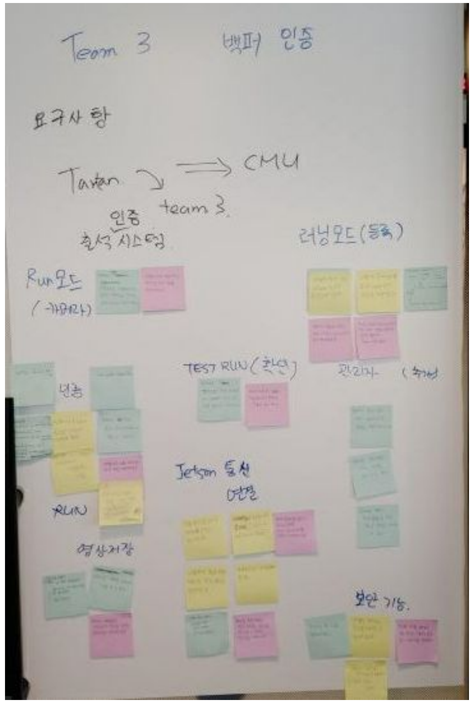
CERTIFIED 100% CERTIFIED

AI Attendance Check System Demo

Enjoy a cup of coffee during the attendance check time

Deriving architecture drivers

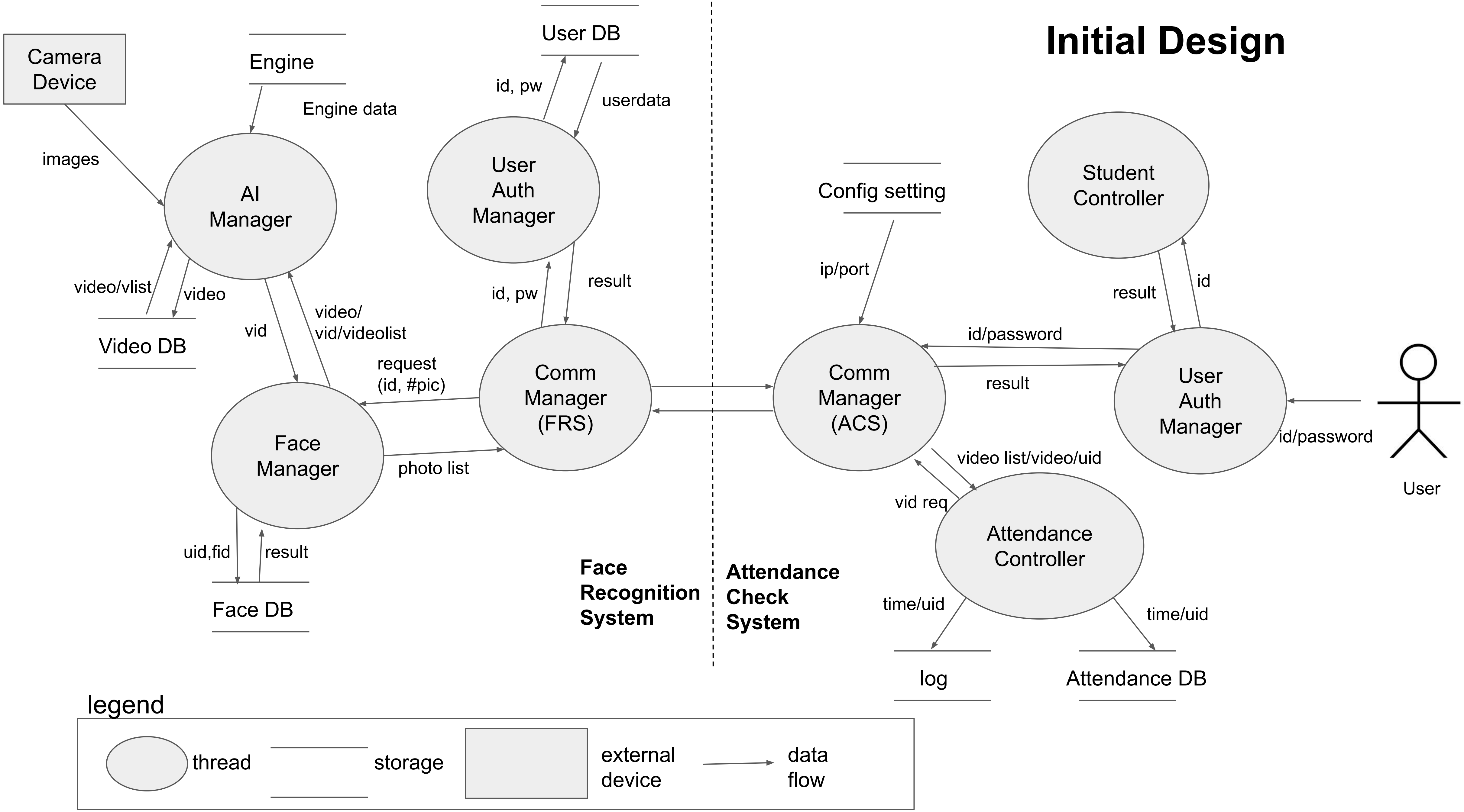




Item	Category	Count
Functional requirements	log in	9
	register of student (Learn mode)	5
	attendance check (Run mode, Test run mode)	8
	security	29
	total	**51**
Quality Attribute		10
Constraints		8

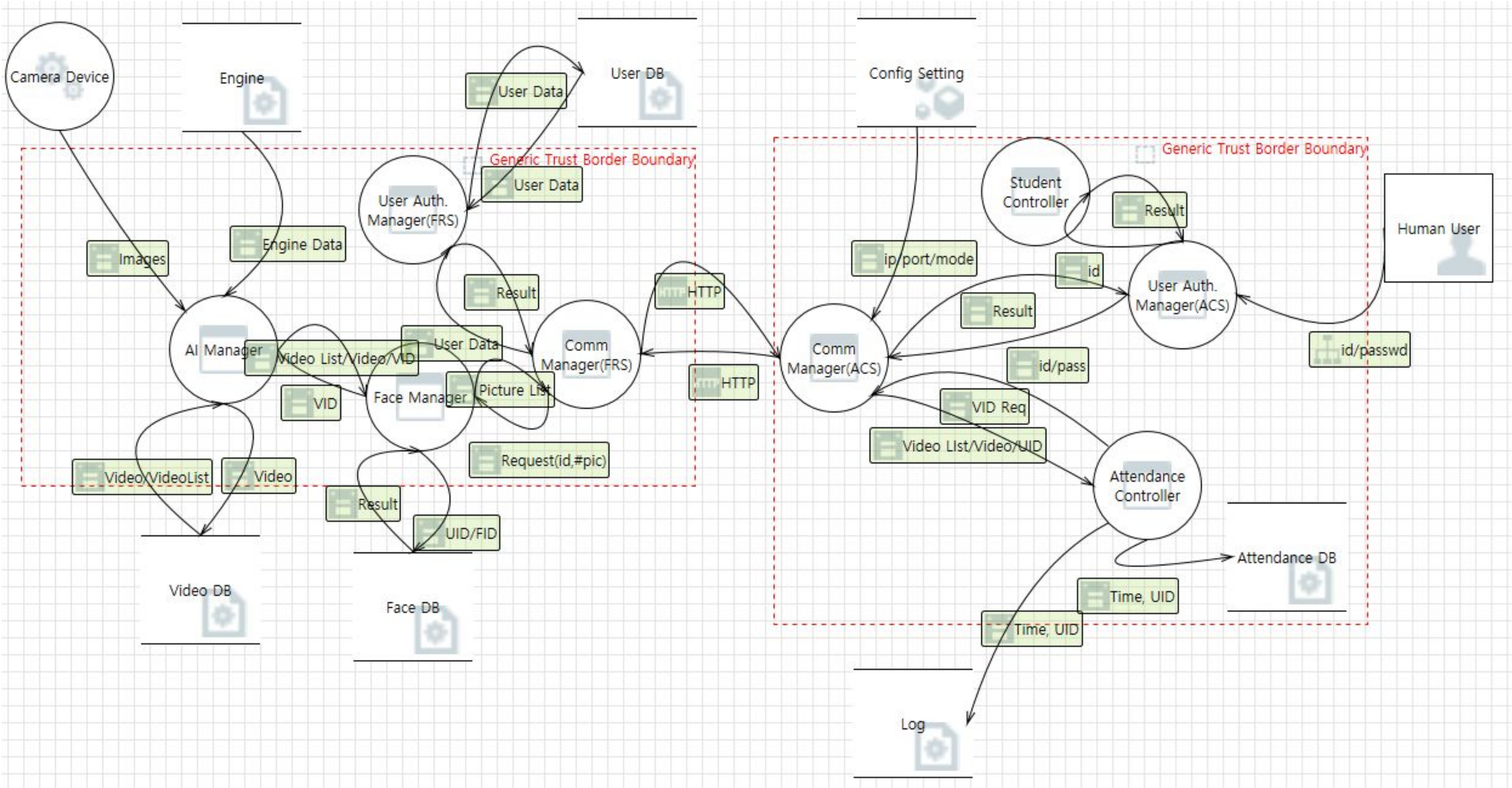
Original requirements : 75%
Additional requirements : 25%

Initial Design



Threat Analysis & Risk Assessment

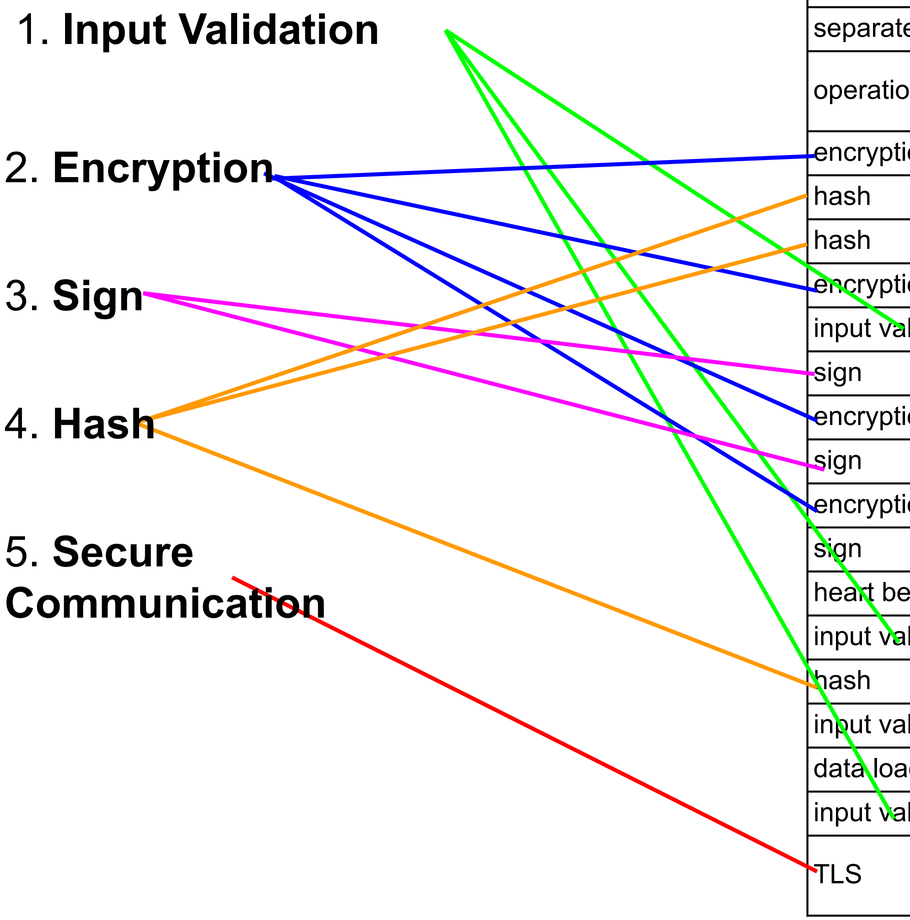
CERTIFIED 100% CERTIFIED



[Voting for risk assessment]

Threat Analysis	Risk assessment for high priority threats	Security Requirements from threat mitigation
138	29	23

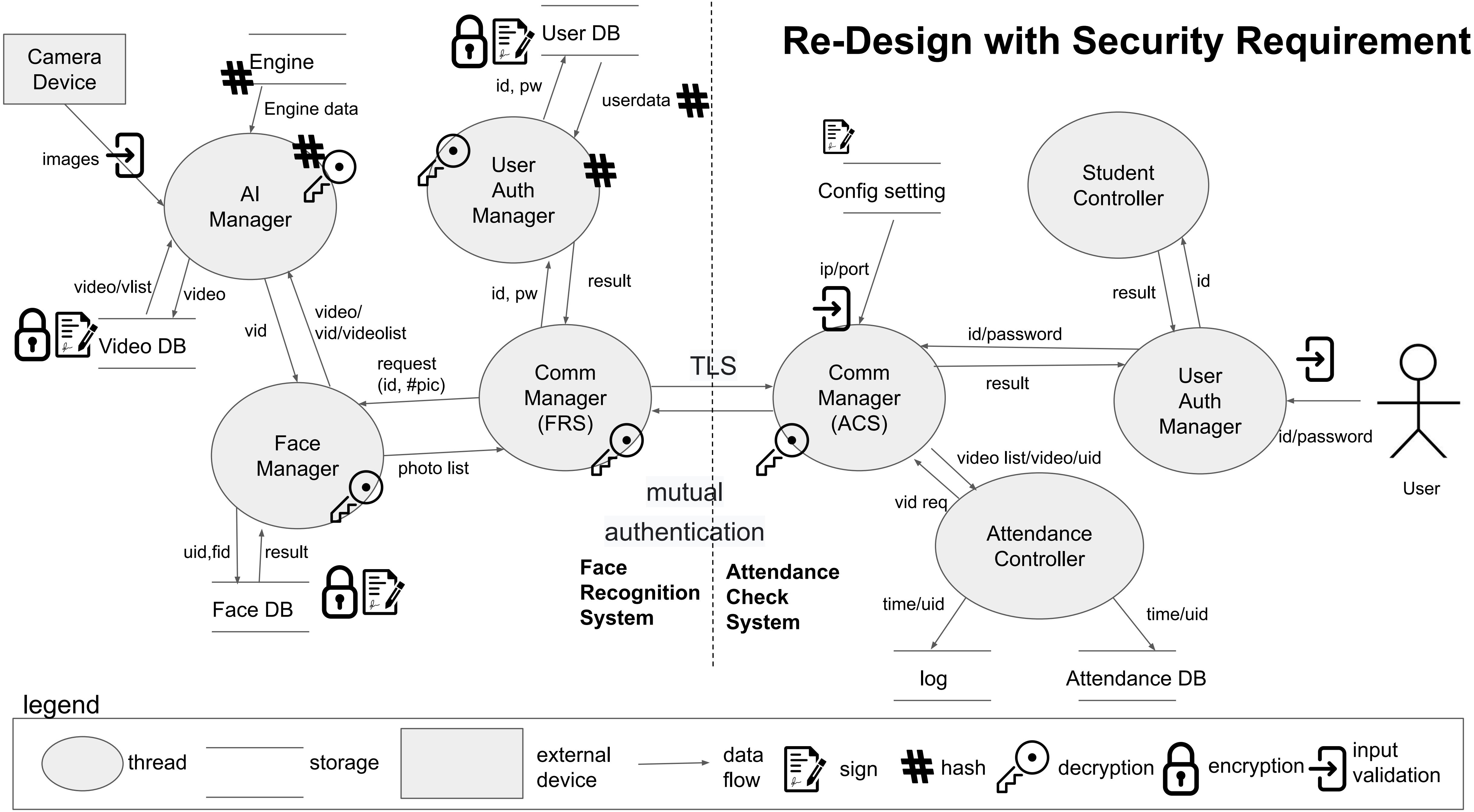
Security Requirements from risk assessment



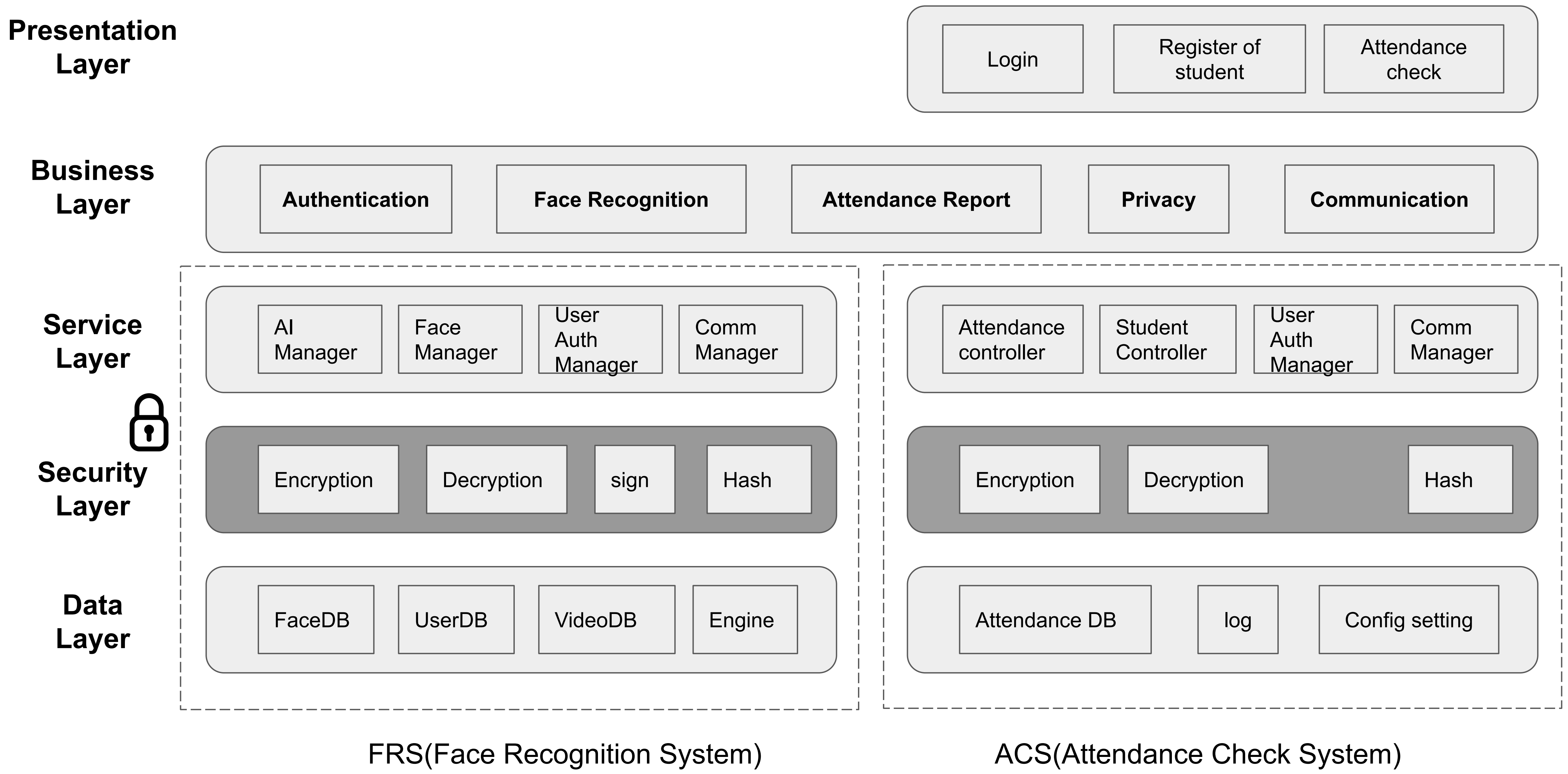
1. **Input Validation**
2. **Encryption**
3. **Sign**
4. **Hash**
5. **Secure Communication**

limited photos	Each student must be able to save up to 5 photos.
check capacity	When saving student photos, this system should be able to check the remaining capacity.
check capacity	When saving a video about attendance, this system should be able to check the remaining capacity.
separate partition	This system should be able to save the video of the attendance in a separate partition.
operation	This system should be able to operate the attendance function even if it is not possible to save the video of the attendance.
encryption	Videos of attendance in this system must be encrypted.
hash	User accounts accessing this system must be hashed.
hash	Config setting file that manages device information stored in the system should be hashed
encryption	Face DB in the system must be encrypted.
input validation	When logging into the system, the input data should be verified.
sign	Face DB data stored in the system must be signed by admin.
encryption	Video DB data stored in the system must be encrypted.
sign	Video DB data stored in the system must be signed by admin.
encryption	User DB data stored in the system must be encrypted.
sign	User DB data stored in the system must be signed by admin.
heart beat	The system's Comm Manager (ACS) must apply a heart beat.
input validation	Input verification for the Config setting in the system should be done.
hash	AI Engine data shall be hashed.
input validation	Input verification for engine data in the system should be done.
data loading	It is necessary to check the loading completion of the engine data of the system.
input validation	The Comm Manager (FRS) in the system should verify the input.
TLS	TLS version 1.2 and above is needed must be applied for communication between FRS and ACS in the system.

Re-Design with Security Requirement



AACS Static view

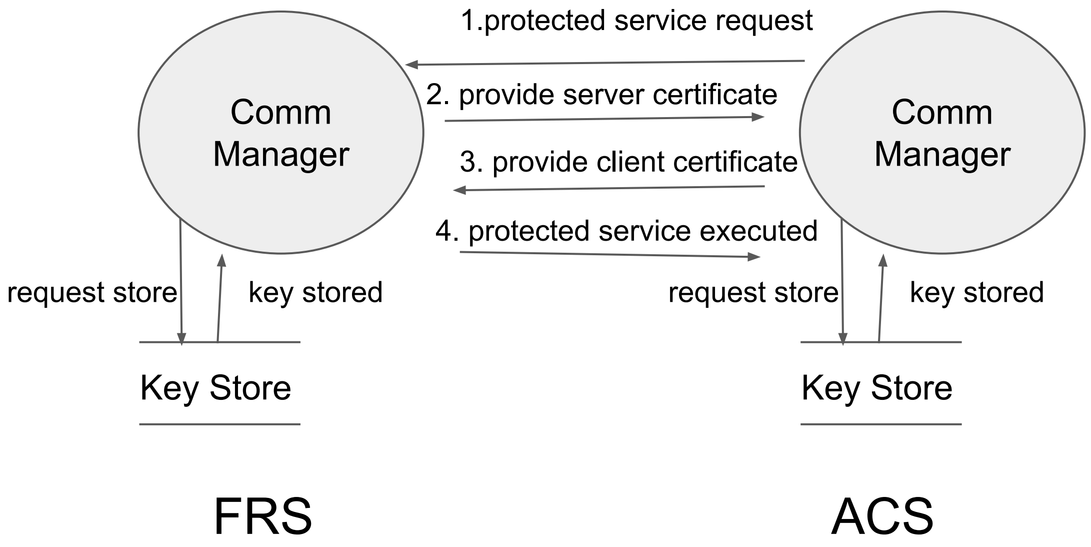


AACS Key Management

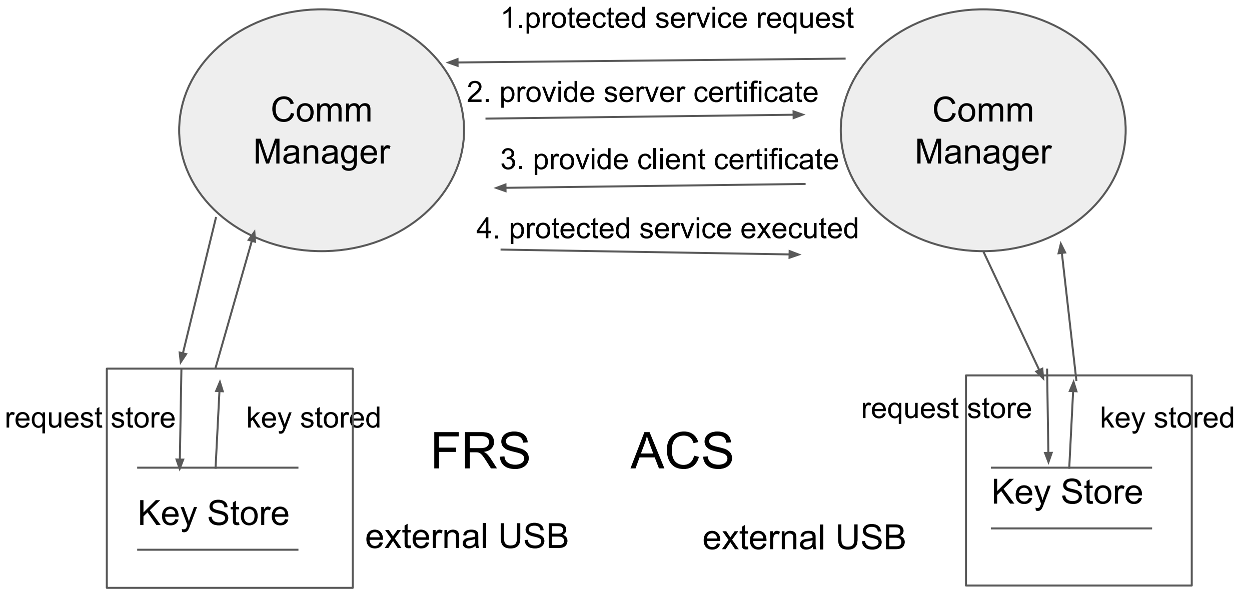
Category	Recommendations
Key lifecycle	Cryptographic key management is critical to the security of a cryptosystem. This includes the generation, exchange, storage, use, destruction and replacement of keys.
Key distribution	The generated keys shall be transported (when necessary) using secure channels and shall be used by their associated cryptographic algorithm within at least FIPS 140-2 compliant cryptographic modules.
Key storage	Ensure all keys are stored in cryptographic vault, such as a hardware security module (HSM) or isolated cryptographic service

AACS Key Management Architectural Alternatives

AACS-QA-010) Key management for system must be secure.


1.protected service request
2. provide server certificate
3. provide client certificate
4. protected service executed
Comm Manager
Comm Manager
request store
key stored
request store
key stored
Key Store
Key Store
FRS
ACS

pre-install key management



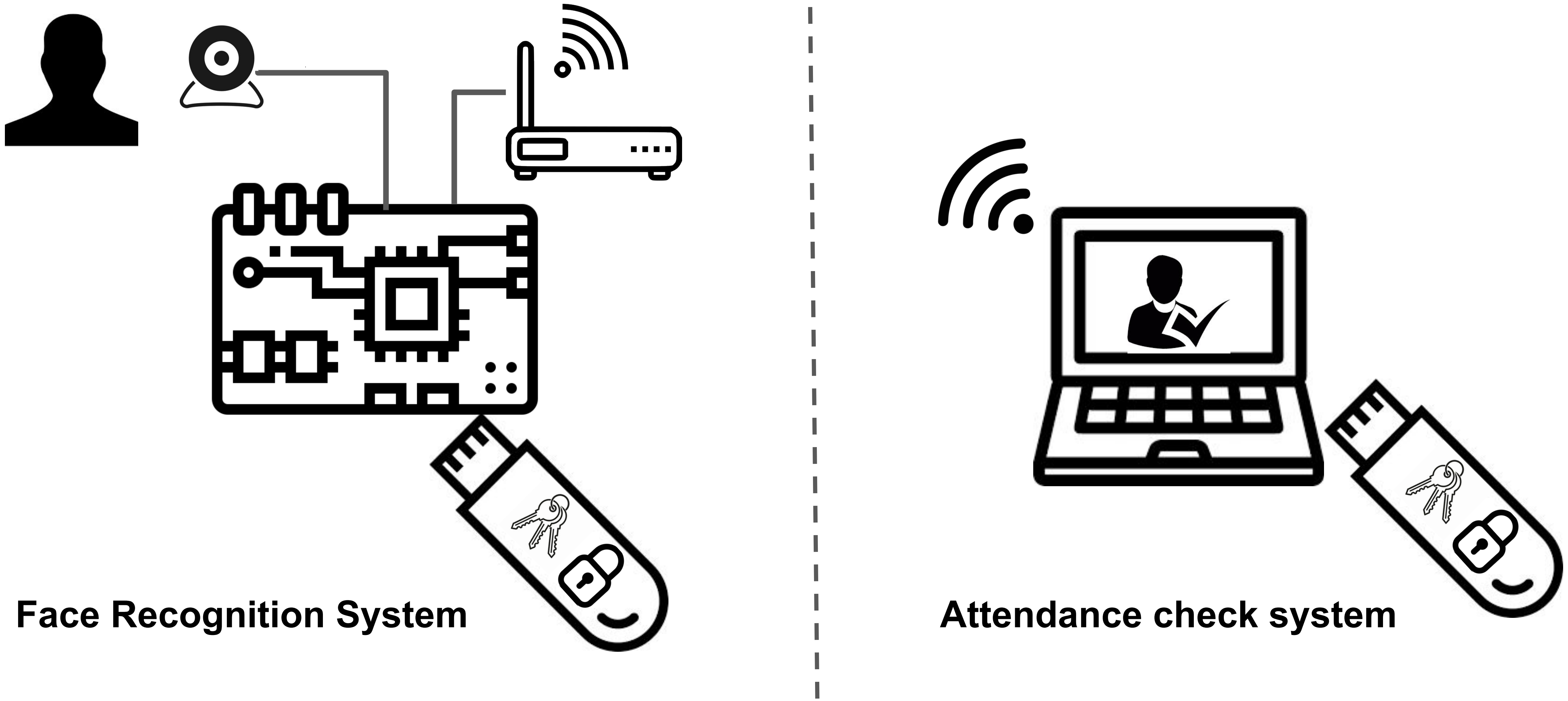
external device key management

	Pre-installed key management	**External Device key management**
Pros	Easy to distribution Easy to key management	Update usb to update key Securely distribution Secure key storage
Cons	Update the whole program to update the key The key is easy to be exposed to risk	Physical usb key management Difficult to distribution

Architectural Decision :

Selected as an **external device key management** for key lifecycle management and secure key management

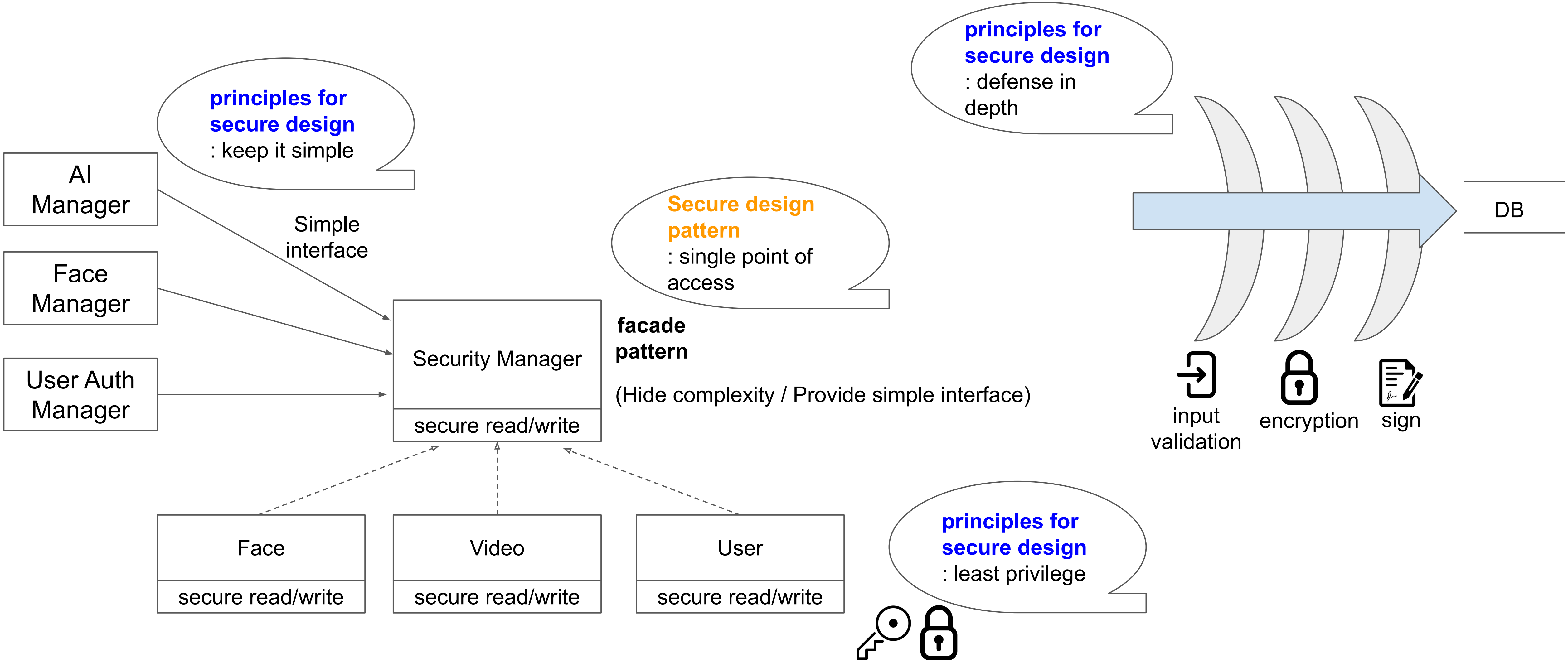
AACS Physical view



Face Recognition System

Attendance check system

AACS Secure Design Pattern



principles for secure design
: keep it simple

AI Manager

Face Manager

User Auth Manager

Simple interface

Security Manager

secure read/write

Secure design pattern
: single point of access

facade pattern

(Hide complexity / Provide simple interface)

Face

secure read/write

Video

secure read/write

User

secure read/write

principles for secure design
: defense in depth

DB

input validation

encryption

sign

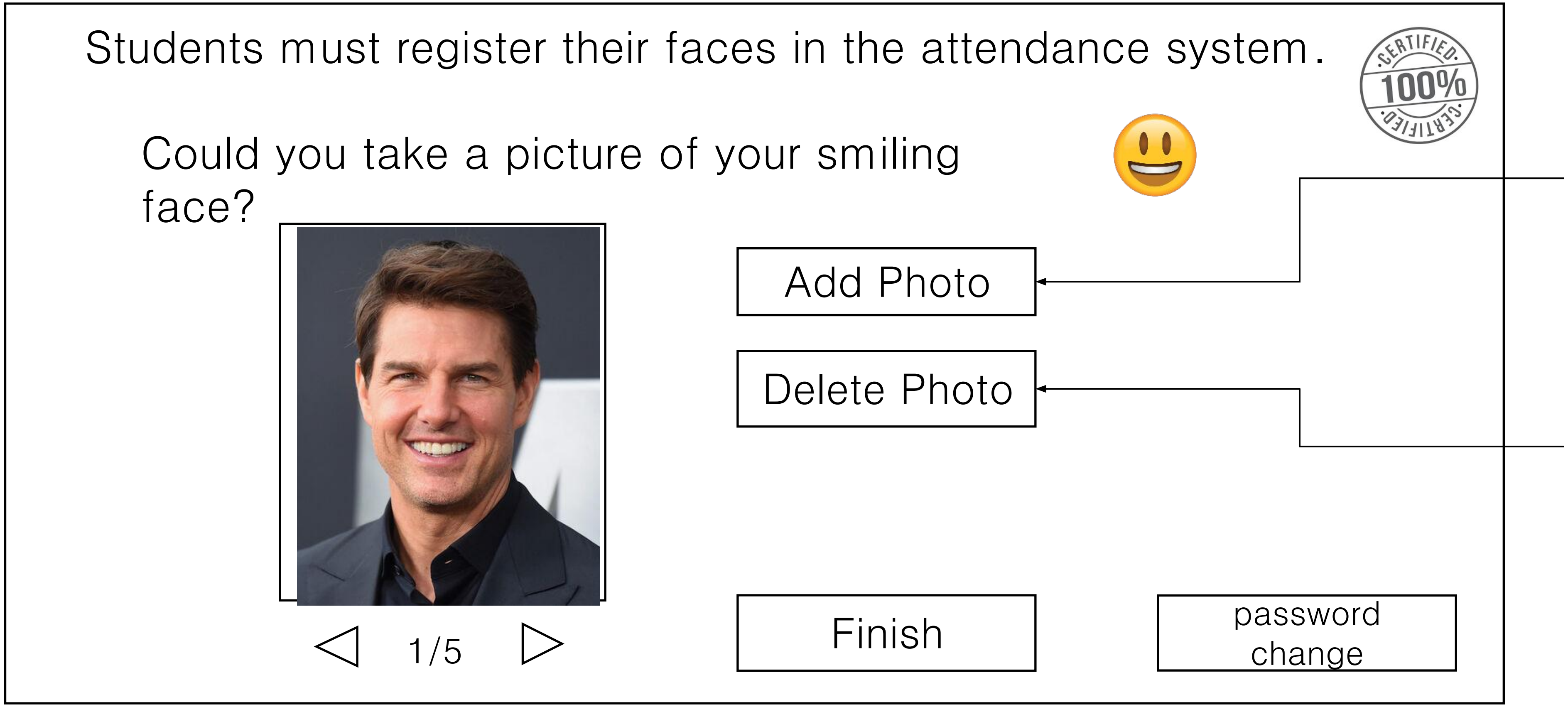
principles for secure design
: least privilege

: Each class performs encryption/decryption using a different key.

AACS Secure UX

Secure design pattern
: Limited view

Each student has a permission to view only their own photo.



Students must register their faces in the attendance system.

Could you take a picture of your smiling face? 😃

Add Picture

This function is designed to register the face of a registered student. Students can register up to 5 photos.

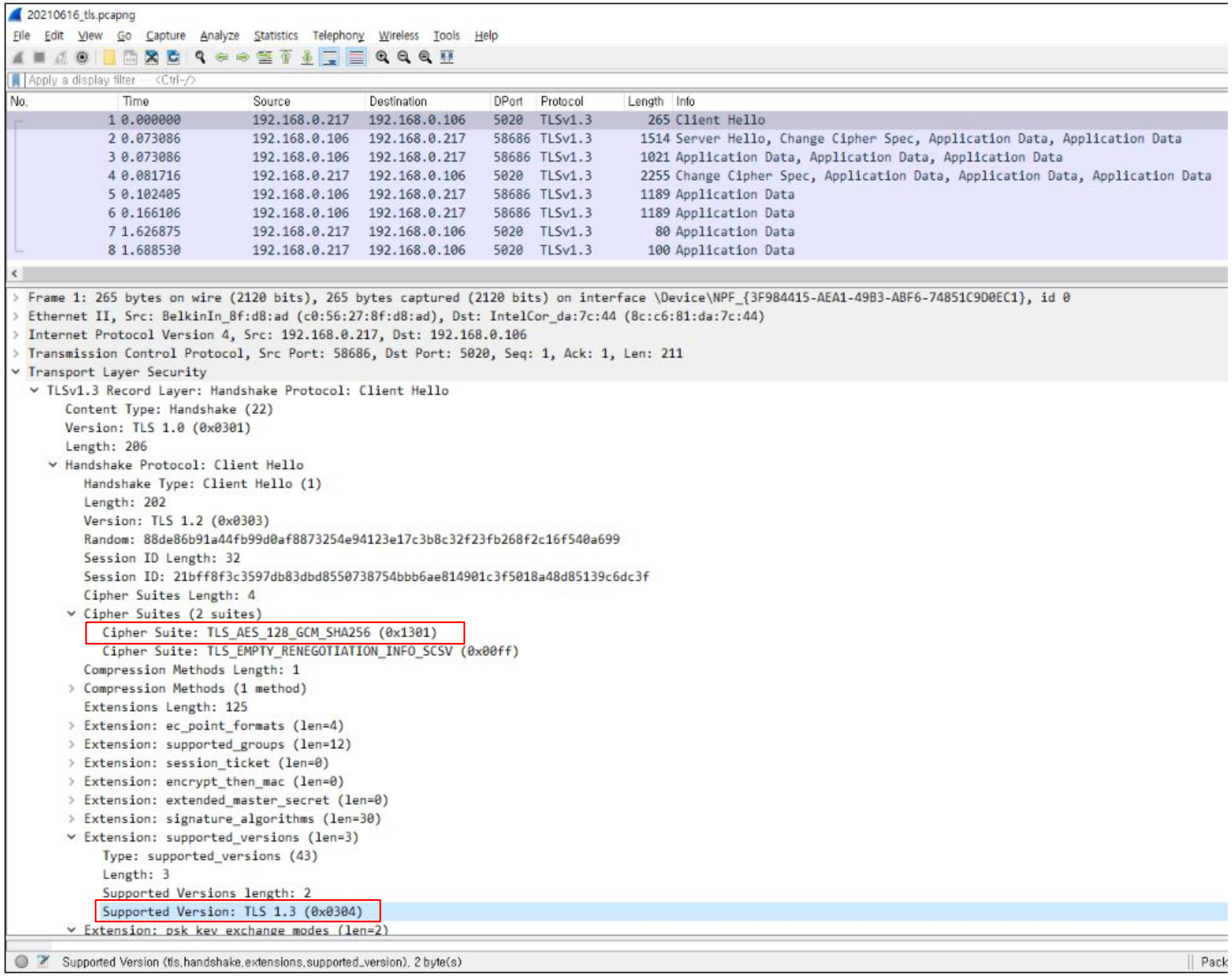
Delete Picture

This is a function to delete a registered student's face photo.

Secure communication with TLS 1.3

Category	Recommendations	Applied to our project	Reason for selection
SSL/TLS version	TLS 1.2 or TLS 1.3	TLS 1.3	The latest version of TLS Don’t need to consider backward compatibility The highest level of security
Cryptographic library for TLS	GnuTLS OpenSSL wolfSSL	OpenSSL v1.1.1 - v1.1.1k (ACS) - v1.1.1 (FRS)	Widely used in industries for the commercial products
Cipher suites	Recommended for TLS 1.3 : TLS_AES_128_GCM_SHA256 TLS_AES_256_GCM_SHA384 TLS_AES_128_CCM_SHA256	TLS_AES_128_GCM_SHA256	AES-128 for performance
Signature keys for certificates	RSA : 2000 bits ~ (~2023) 3000 bits ~ (2024~2027+)	RSA-2048 bits	RSA-2048 bits for performance

Secure communication with TLS 1.3



20210616_tls.pcapng
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ··· <Ctrl-/>

```
No.     Time        Source          Destination     DPort  Protocol  Length Info
      1 0.000000    192.168.0.217   192.168.0.106   5020   TLSv1.3     265 Client Hello
      2 0.073086    192.168.0.106   192.168.0.217   58686  TLSv1.3    1514 Server Hello, Change Cipher Spec, Application Data, Application Data
      3 0.073086    192.168.0.106   192.168.0.217   58686  TLSv1.3    1021 Application Data, Application Data, Application Data
      4 0.081716    192.168.0.217   192.168.0.106   5020   TLSv1.3    2255 Change Cipher Spec, Application Data, Application Data, Application Data
      5 0.102405    192.168.0.106   192.168.0.217   58686  TLSv1.3    1189 Application Data
      6 0.166106    192.168.0.106   192.168.0.217   58686  TLSv1.3    1189 Application Data
      7 1.626875    192.168.0.217   192.168.0.106   5020   TLSv1.3      80 Application Data
      8 1.688530    192.168.0.217   192.168.0.106   5020   TLSv1.3     100 Application Data
```

```
> Frame 1: 265 bytes on wire (2120 bits), 265 bytes captured (2120 bits) on interface \Device\NPF_{3F984415-AEA1-49B3-ABF6-74851C9D0EC1}, id 0
> Ethernet II, Src: BelkinIn_8f:d8:ad (c0:56:27:8f:d8:ad), Dst: IntelCor_da:7c:44 (8c:c6:81:da:7c:44)
> Internet Protocol Version 4, Src: 192.168.0.217, Dst: 192.168.0.106
> Transmission Control Protocol, Src Port: 58686, Dst Port: 5020, Seq: 1, Ack: 1, Len: 211
v Transport Layer Security
  v TLSv1.3 Record Layer: Handshake Protocol: Client Hello
      Content Type: Handshake (22)
      Version: TLS 1.0 (0x0301)
      Length: 206
    v Handshake Protocol: Client Hello
        Handshake Type: Client Hello (1)
        Length: 202
        Version: TLS 1.2 (0x0303)
        Random: 88de86b91a44fb99d0af8873254e94123e17c3b8c32f23fb268f2c16f540a699
        Session ID Length: 32
        Session ID: 21bff8f3c3597db83dbd8550738754bbb6ae814901c3f5018a48d85139c6dc3f
        Cipher Suites Length: 4
      v Cipher Suites (2 suites)
          Cipher Suite: TLS_AES_128_GCM_SHA256 (0x1301)
          Cipher Suite: TLS_EMPTY_RENEGOTIATION_INFO_SCSV (0x00ff)
        Compression Methods Length: 1
      > Compression Methods (1 method)
        Extensions Length: 125
      > Extension: ec_point_formats (len=4)
      > Extension: supported_groups (len=12)
      > Extension: session_ticket (len=0)
      > Extension: encrypt_then_mac (len=0)
      > Extension: extended_master_secret (len=0)
      > Extension: signature_algorithms (len=30)
      v Extension: supported_versions (len=3)
          Type: supported_versions (43)
          Length: 3
          Supported Versions length: 2
          Supported Version: TLS 1.3 (0x0304)
      v Extension: psk_key_exchange_modes (len=2)
```

Supported Version (tls.handshake.extensions.supported_version), 2 byte(s) | Pack

Cryptographic Algorithms

Category	Recommendations	Applied to our project	Reason for selection
Block ciphers	AES-128, AES-192, AES-256	AES-128	Choose the shortest key length for performance
Mode of operation for block ciphers	CCM (Counter with Cipher Block Chaining Message Authentication) GCM (Galois/Counter Mode) CBC (Cipher Block Chaining) CTR (Counter Mode)	CBC	The most secure mode among the block cipher modes
Hash functions	SHA-256, SHA-512/256, SHA-384 and SHA-512 SHA3-256, SHA3-384, SHA3-512	SHA-256	Most widely used. Fast and strong enough for most purposes.
Digital signatures	RSA, DSA, ECDSA, ECKDSA, ECGDSA, XMSS+ or LMS	RSA-2048 bits	Most widely used.

Cryptoperiod

No.	Key Type	Cryptographic keys stored in FRS USB (Server-side)	Cryptographic keys stored in ACS USB (Client-side)	Cryptoperiod Recommended	
				Originator-Usage Period (OUP)	Recipient-Usage Period
1	Private signature key	Private signature key for signing Video DB Private signature key for signing User DB Private signature key for signing Face DB		1 to 3 years	-
2	Public signature-verification key	Public signature-verification key for Video DB Public signature-verification key for User DB Public signature-verification key for Face DB Root CA certificate	Client certificate Root CA certificate (self-signed certificate)	Several years (depends on key size)	
4	Private authentication key	Server key	Client key	1 to 2 years	
5	Public authentication key	Server certificate Root CA certificate	Client certificate Root CA certificate (self-signed certificate)	1 to 2 years	
6	Symmetric data encryption key	Symmetric data encryption key for encrypting Video DB Symmetric data encryption key for encrypting User DB Symmetric data encryption key for encrypting Face DB		Up to 2 years	Up to OUP + 3 years

NIST SP 800-57

Applied to our project

- Root CA certificate cryptoperiod : 1 year
- Client/Server certificate cryptoperiod : 3 months

** Root CA private key, Private signature key for signing client configuration file are located in Admin PC

Security Assessment Reports

1) Static Analysis Report based on RATS

Folder	High	Low	Medium	Total
ControlAndDisplay(ACS)	6	2		8
LgFaceRecDemoTCP_Jetson_NanoV2(FRS)	6	9	6	21
Total	**12**	**11**	**6**	**29**

Issue	High	Low	Medium	Total
EVP_DecryptUpdate			1	1
EVP_EncryptUpdate			1	1
fixed size global buffer	10			10
fixed size local buffer		7		7
memcpy		4		4
read			4	4
wsprintf	2			2
Total	**12**	**11**	**6**	**29**

https://security.web.cern.ch/recommendations/en/codetools/rats.shtml

2) Open Source Vulnerability Report

OSS Name / Version	High	Low	Medium	Total
NVIDIA Tegra kernel v4.9	1		2	3
OpenCV v4.1.1			1	1
OpenSSL v1.1.1	2	2	8	12
Total	**3**	**2**	**11**	**16**

AACS Functional Test Results

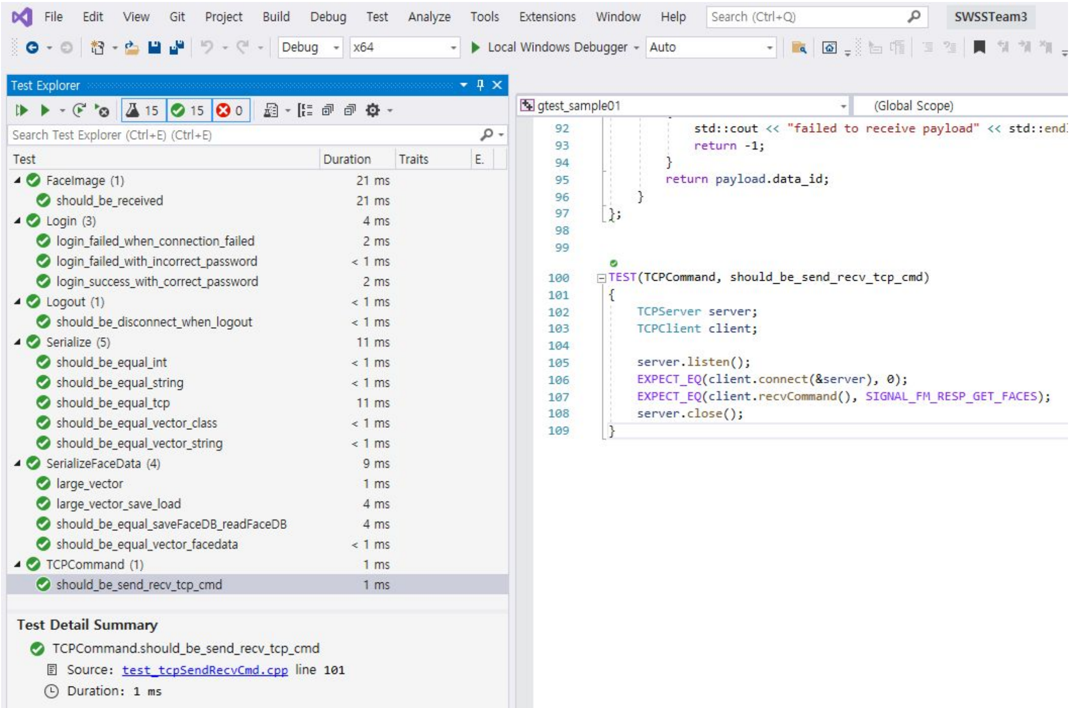
Manual Test

- Functional test cases developed based on S/W requirements
- Some test cases developed from Given- When-Then pattern

Item	Description
Test Scope	AACS Requirements
Requirement Test Coverage	**80%**
Test Level	**System Test**
Test Method	Manual Test
Number of Test Cases	49
Pass	45
Fail	4
Pass Rate	**91.8 %**

Automatic Test (gTest)

- The unit test cases and the basic scenario tests developed
- 4 issues were detected and fixed based on automated tests
- 15 test cases developed



```
Test                                        Duration   Traits   E.
◢ ✓ FaceImage (1)                              21 ms
    ✓ should_be_received                       21 ms
◢ ✓ Login (3)                                   4 ms
    ✓ login_failed_when_connection_failed       2 ms
    ✓ login_failed_with_incorrect_password    < 1 ms
    ✓ login_success_with_correct_password       2 ms
◢ ✓ Logout (1)                                < 1 ms
    ✓ should_be_disconnect_when_logout        < 1 ms
◢ ✓ Serialize (5)                              11 ms
    ✓ should_be_equal_int                     < 1 ms
    ✓ should_be_equal_string                  < 1 ms
    ✓ should_be_equal_tcp                      11 ms
    ✓ should_be_equal_vector_class            < 1 ms
    ✓ should_be_equal_vector_string           < 1 ms
◢ ✓ SerializeFaceData (4)                       9 ms
    ✓ large_vector                              1 ms
    ✓ large_vector_save_load                    4 ms
    ✓ should_be_equal_saveFaceDB_readFaceDB     4 ms
    ✓ should_be_equal_vector_facedata         < 1 ms
◢ ✓ TCPCommand (1)                              1 ms
    ✓ should_be_send_recv_tcp_cmd               1 ms
```

Test Detail Summary

✓ TCPCommand.should_be_send_recv_tcp_cmd

Source: test_tcpSendRecvCmd.cpp line 101

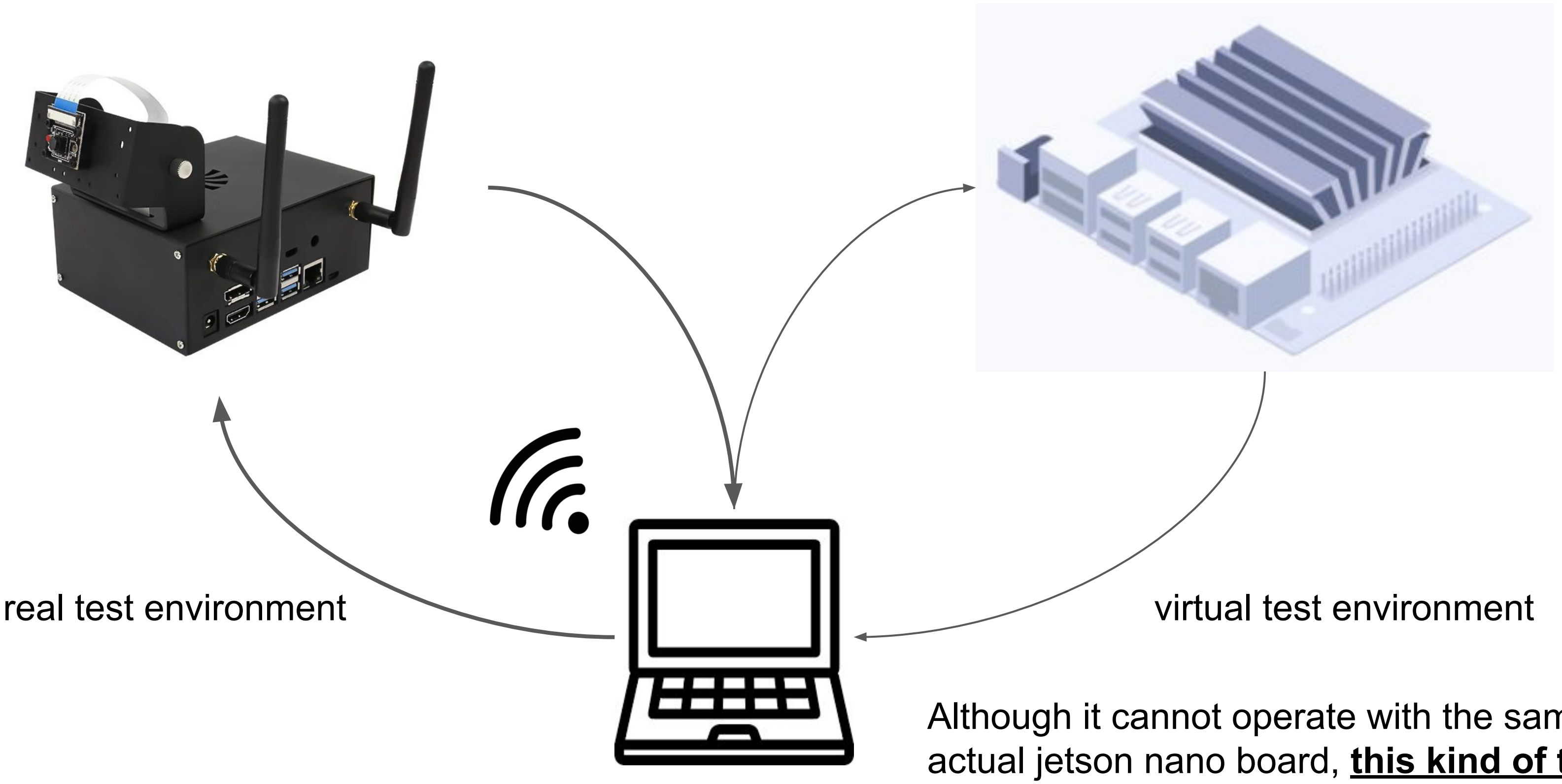
Duration: 1 ms

```
 92                 std::cout << "failed to receive payload" << std::end
 93                 return -1;
 94             }
 95             return payload.data_id;
 96         }
 97     };
 98
 99
100     TEST(TCPCommand, should_be_send_recv_tcp_cmd)
101     {
102         TCPServer server;
103         TCPClient client;
104
105         server.listen();
106         EXPECT_EQ(client.connect(&server), 0);
107         EXPECT_EQ(client.recvCommand(), SIGNAL_FM_RESP_GET_FACES);
108         server.close();
109     }
```

AACS Test Environment Improvements

The development environment was difficult to develop with one test board.
In some cases, performance was slow or the system crashed when multiple people connected to the board and tested it.

To solve this problem, a virtual jetson nano board was developed and used for testing.



Although it cannot operate with the same function as the actual jetson nano board, **this kind of test simulator is useful for data communication testing using the protocol.**

Lesson Learned !

1. It was a good opportunity to learn about the process of enforcing security in architecture.
2. The security was reflected in the design from the beginning, so the implementation had been well.
3. Lack of time was the biggest constraint, but we were able to overcome it through teamwork.
4. When faced with a problem in the project, I was able to learn how to apply to this project and to make a decision with discussion to solve the problem.
5. It's good to know the usage of openssl and learn from team mates about security manager implementation.
6. From design to deployment, I felt that secured software had a lot to consider and took time.
7. There were too many requirements to implement, but proper decision on what can be done in within given time help a lot to reach to a good shape.

We learned a lot from phase 1 and are looking forward to phase 2 !

Security Response Contact : cmu-team3@lge.com