CS350 Safehome Project
Software Design Specification (SDS)
Team 3

2025-11-14

20240397	Hichan Shin
20240782	Youngdo Han
20240905	Bumgyu Suh
20240951	Mohamed Elnakeeb

Korea Advanced Institute of
Science and Technology

Table of Contents

I. Overview

1. Introduction

This document describes the design model of SafeHome system proposed in the previous report. Since the design phase of the product is directly connected to the implementation phase, we focus on the well-formed and concrete design of the system. Architectural structure, class diagram, CRC cards, and state diagrams are presented to portray the design of the system. The diagrams were made with PlantUML, and their corresponding PlantUML shorthand code are provided at the end of this document referring to their figure number (Fig. X) and name.

Please refer to the SRS document for the correct definitions of terms such as the Glossary. The assumptions made in the SRS are valid, but some are overridden in this document in the I.4. Assumptions section.

2. Goal

The goal is to create a modular and secure architecture that supports surveillance and security features from multiple user interfaces.
1) Follow the SRS but making changes where inconsistencies are found.
2) Achieve low coupling, high cohesion, and modularity in design.
3) Pursue testability, integrity, efficiency, maintainability, and reliability.
4) Minimize complexity and consider reusability and flexibility.

3. How the Design Work Proceeded

1) **Creation of conceptual diagram:** The SRS's overall design of the system was improved upon and fixed by creating a conceptual diagram in order to make an overall structure of the system.
2) **Extraction of classes:** Nouns and verbs from use case scenarios were used to extract classes based on each subsystem categorized in the SRS, matched with the conceptual diagram.
3) **Creation of architectural structures:** The architectural structure of subsystems was derived from the organization of classes from step 2 and conceptual diagram from step 1.
4) **Creation of class diagrams:** Extracted classes from step 2 and architectural structure from step 3 were used to make class diagrams directly considering the implementation.
5) **Creation of REST API:** Methods provided by subsystem classes listed in the class diagrams were filtered to define the REST API to be used for the system.
6) **Creation of CRC cards:** The class diagrams from step 4 were used to specify each class in CRC cards.
7) **Creation of state diagram:** Based on the class diagram and CRC descriptions of classes, appropriate state diagrams were created.
8) **Review of relations:** Relationships between the classes of the subsystems were reviewed, and class diagrams, and architecture were modified accordingly.
9) **Evaluation:** The design was evaluated using several metrics, final needed adjustments were made.

4. Assumptions

1) The SafeHome Cloud mentioned in the SRS does not store logs, recordings, and settings, user accounts which will be stored locally. Instead, the SafeHome Cloud is only used for connecting a remote user to the local SafeHome Hub via hole-punching.
2) When the SafeHome owner buys the product, the company sets up the floor plan (including the location of the SafeHome devices), so that this is set up already and is not changed through the SafeHome Software. Floor plan configuration and hardware deployment is complete and out of the scope of our project, therefore, the SRS's Use Case VI. 3.1.1 Add and Configure New Devices is out of the scope of our project. Reconfiguring the floor plan or relocating/adding sensors or cameras are not in the scope of our project. The floor plan is given and sensor locations static and predetermined. Hence, the SafeHome software is specialized for a certain floor plan.
3) The SafeHome System can be accessed through the physical control panel installed that can be interacted with physically, or online, through the web. Both methods will share the same front-end for convenience and intuitive user experience.
4) The initial admin user already exists and so its ID and PW (password) is known by the appropriate person using SafeHome.
5) Cameras are IP-Based and may or may not support movement(zoom and pan)
6) SafeHome software runs on dedicated hardware installed in home.
7) Any issues related to the hardware, such as networking, power, or temperature settings, are outside of the scope of our project.
8) Security zones are rectangular boxes drawn over the Floor Plan that include all the sensors represented by dots.
9) Security zones may overlap sensors, and these sensors will be toggled first-come-first-serve. This is handled in the security subsystem.
10) Cameras and motion sensors are separate objects, but cameras detecting movement may trigger actions in the security subsystem as mentioned in the SRS.

II. Architectural Structure

1. Conceptual Diagram

SafeHome consists of:

1) The main **SafeHome Hub Server** that has the SafeHome Hub software installed in the physically hardware installed at home, which

a) **stores** persistent data such as surveillance camera recordings, SafeHome configuration, and user authentication/authorization settings in the **Database(hub)**,

b) **interacts** with physical **Devices** that include: surveillance cameras, motion sensors, and alarms, by **controlling** the devices and **receiving** signals from the devices,

c) **receives** input from and **displays** info on the **control panel** installed in the home, which is used by the **users**.

d) **allows** external access (from outside the home) to SafeHome Hub Server functions through the **SafeHome Cloud Server**, which allows a P2P connection directly from the **users**.

2) The **SafeHome Cloud Server**, which allows external access to the **SafeHome Hub Server** through the internet from the users' mobile app or browser app.

Note that the software design specification pertains to the design of the main Safehome program running in the SafeHome Hub Server shown in the diagram below. The rest of the document is therefore in the context of the SafeHome Hub Server unless noted otherwise.

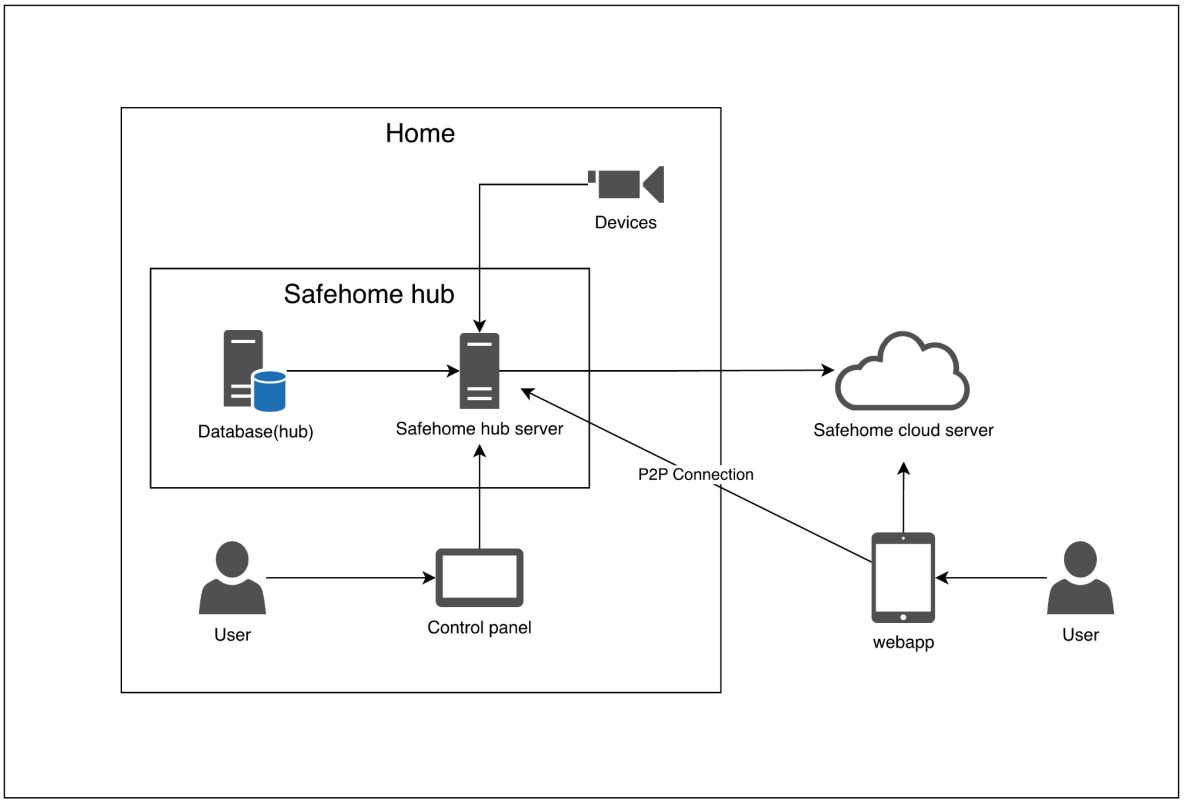

Fig. 1 - Conceptual Diagram

2. Overall SafeHome Hub Architecture

As made clear by the conceptual diagram, the system follows a **layered architecture**: presentation, subsystems, and database/devices. The REST API facilitates the communication between the presentation layer and the subsystems. The ER diagram below specifies the details of the database needed by the subsystems to ensure persistence of important configurations and states.

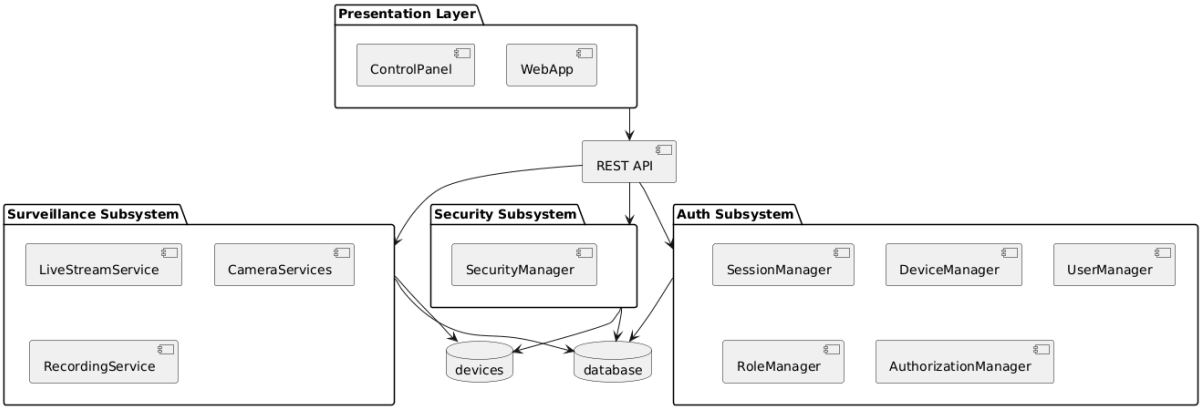

Fig. 2 - SafeHome Hub Architecture

3. Database (ER Diagram)

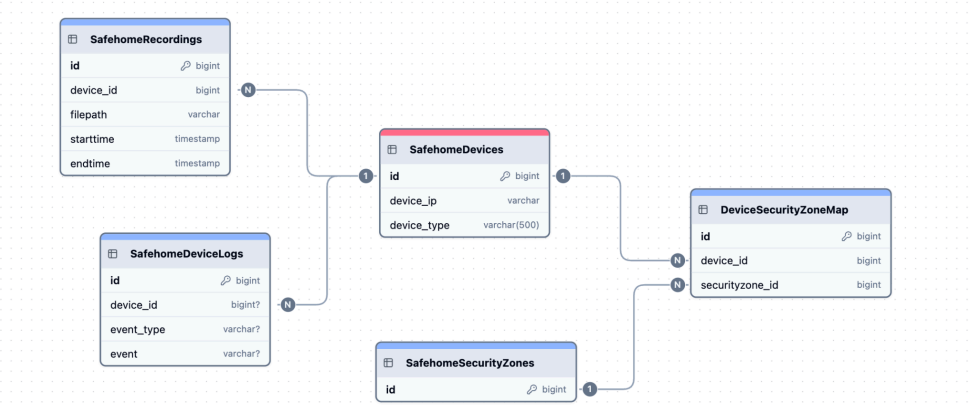

Fig. 3 - Surveillance & Security Subsystem ER Diagrams

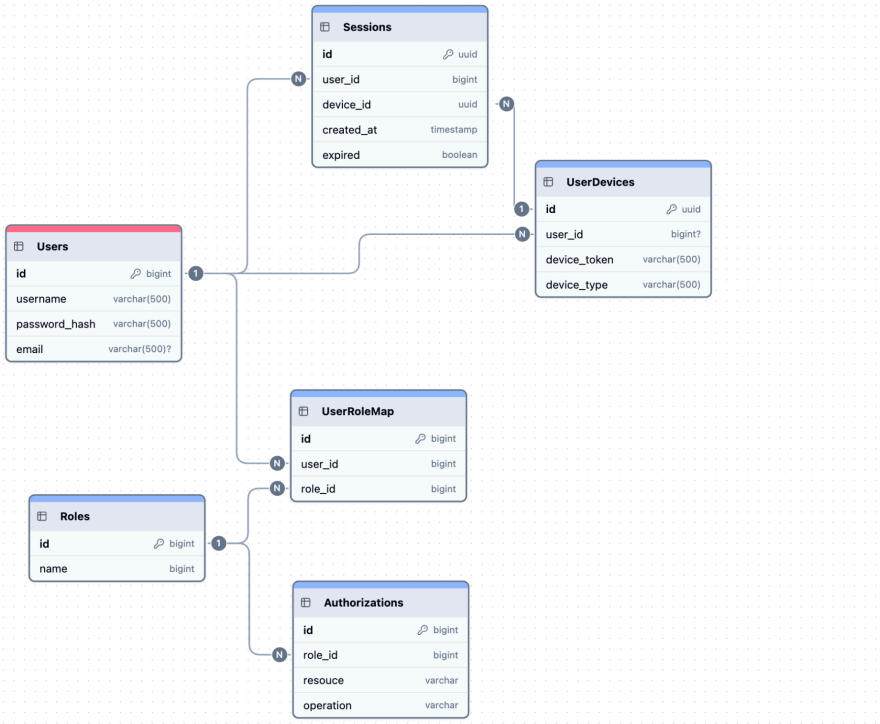

Fig. 4 - Auth Subsystem ER Diagram

4. REST API

REST API is an interface between safehome hub and webapp.

1. Surveillance Subsystem REST API

ID	Method	URI	Description
PCS	PATCH	/camera/{id}	Patch state of {id} camera
GRI	GET	/camera/{id}/record	Get list of camera record id
GCR	GET	/record/{id}	Get URL of {id} record
GCS	GET	/camera/{id}/snapshot	Get camera snapshot
GLC	GET	/cameras	Get list of cameras
GCL	GET	/camera/{id}/live	Get {id} camera live stream
PCL	PATCH	/camera/{id}/live	Patch state of {id} camera live stream

2. Security Subsystem REST API

ID	Method	URI	Description
GLM	GET	/modes	Get list of modes
SSM	PATCH	/mode	Patch state of security mode
GSE	GET	/event/{id}	Get information of {id} security event
PES	PATCH	/event/{id}	Patch state of {id} event
CSE	POST	/event	Post security event created by user
GLS	GET	/sensors	Get list of sensors
GSI	GET	/sensor/{id}	Get information of {id} sensor
PSS	PATCH	/sensor/{id}	Patch state of {id} sensor
CSZ	POST	/securityzone	Create security zone
DSZ	DELETE	/securityzone/{id}	Delete {id} security zone
GIZ	GET	/securityzone/{id}	Get information of {id} security zone

3. Auth Subsystem REST API

ID	Method	URI	Description
LIN	POST	/session	Create new session
LOT	DELETE	/session/{id}	Delete {id} session
CNR	POST	/role	Create new role
GLR	GET	/roles	Get list of roles
CRA	POST	/authority	Create new authority
DRA	DELETE	/authority/{id}	Delete {id} authority
GUI	GET	/user/{id}	Get information of {id} user
GLU	GET	/users	Get list of users
CNU	POST	/user	Create new user
PUS	PATCH	/user/{id}	Patch state of {id} user

5. Surveillance Subsystem Component Diagram

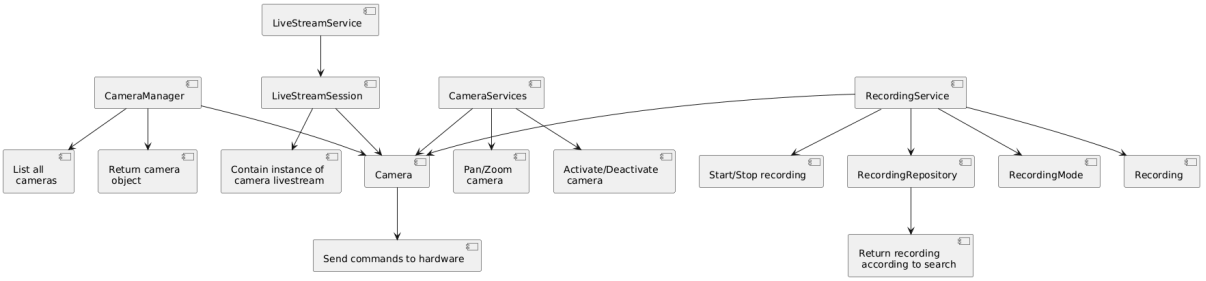

Fig. 5 - Surveillance Component Diagram

6. Security Subsystem Component Diagram

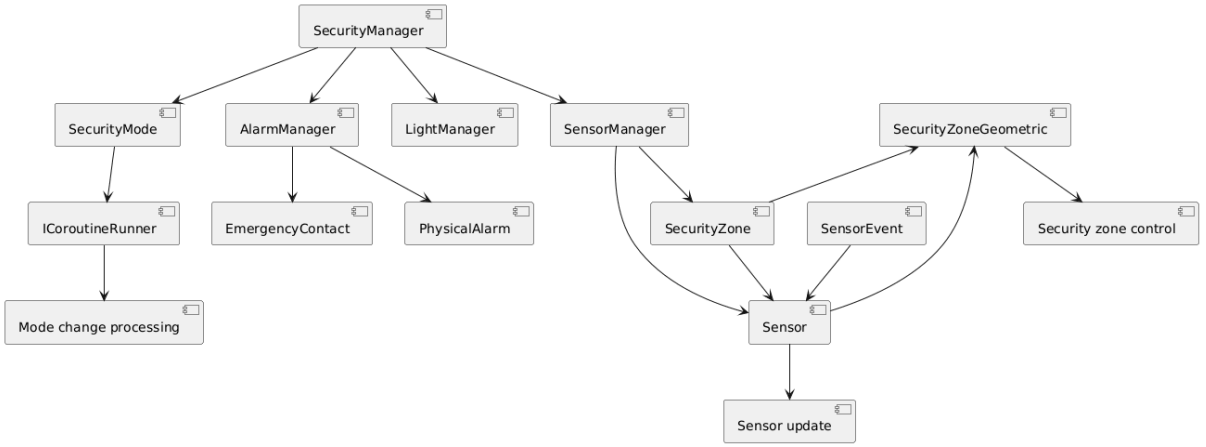

Fig. 6 - Security Component Diagram

7. Auth Subsystem Component Diagram

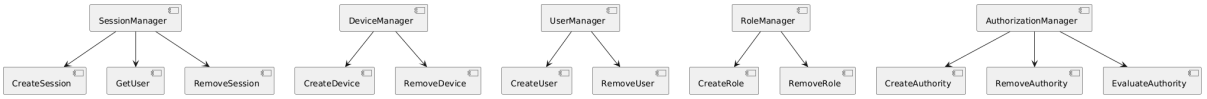

Fig 7. - Auth Subsystem Component Diagram

III. Class Diagrams

The architectural-level view of the whole system is shown below first, then the rest are detailed class-level view of classes labeled in the overview.
Class diagrams were made using PlantUML: arrows indicate dependency (dependent class → dependency class), and for clarification, classes that depend on a specific class will have the dependency class as a property. Dataclasses are used to clarify what type of data are stored in the databases used by the classes, and what other datatypes are used throughout the class diagrams.

1. Class Diagram - Whole System Overview

A comprehensive UML class diagram showing all subsystems and how they interconnect. This shows the big picture — how classes from different modules relate.

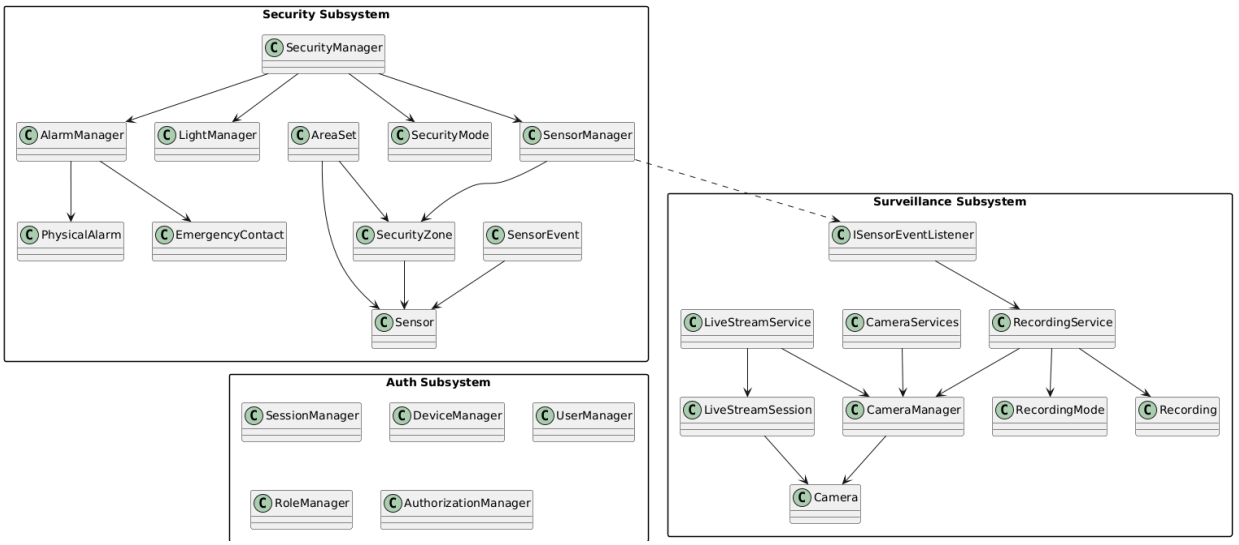

Fig. 8 - Overview Class Diagram

2. Class Diagram - Surveillance Subsystem

The surveillance subsystem pertains to the surveillance cameras and related functions that the users may use, such as viewing the current livestream feed of a specific camera or viewing past recordings. All camera functions access the interface of the physical cameras.

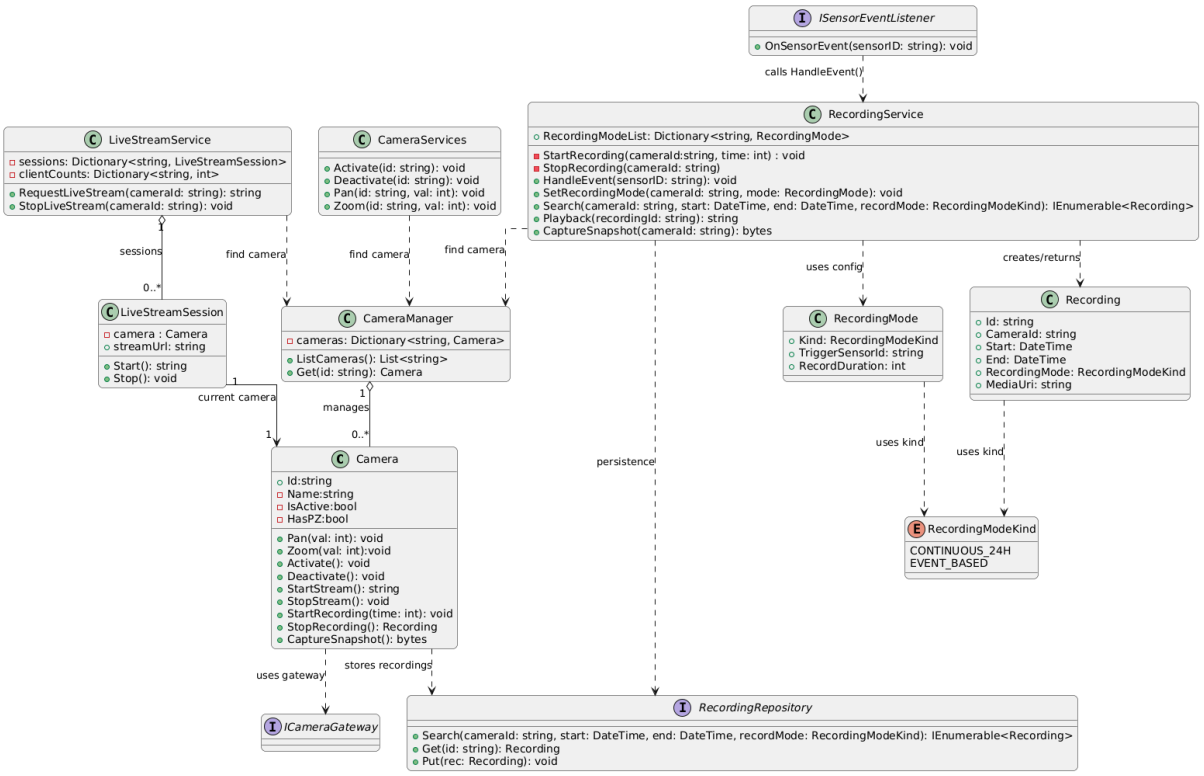

Fig. 9 - Surveillance Subsystem Class Diagram

See also: Camera state diagram and LiveStreamService state diagram

3. Class Diagram - Security Subsystem

The security subsystem deals with receiving signals from the sensors (motion sensors, window/door sensors, and other sensors that can be triggered) and reacting accordingly to configuration, and turning the sensors on and off.

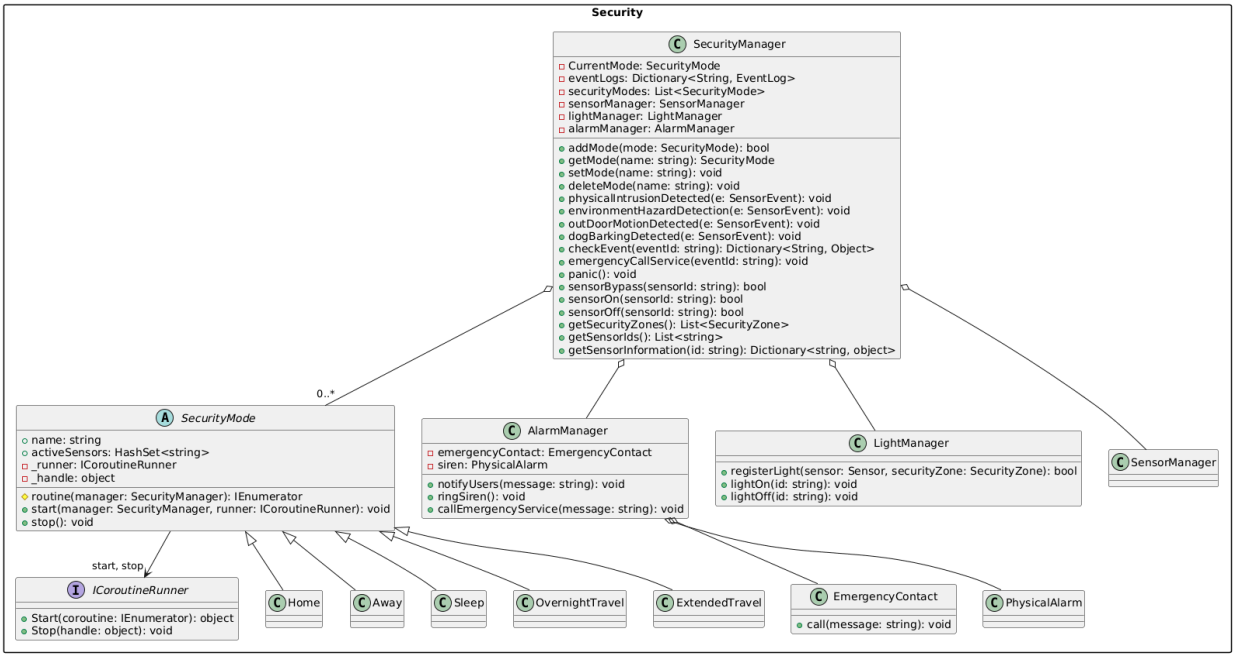

Fig. 10 - Security Subsystem Class Diagram

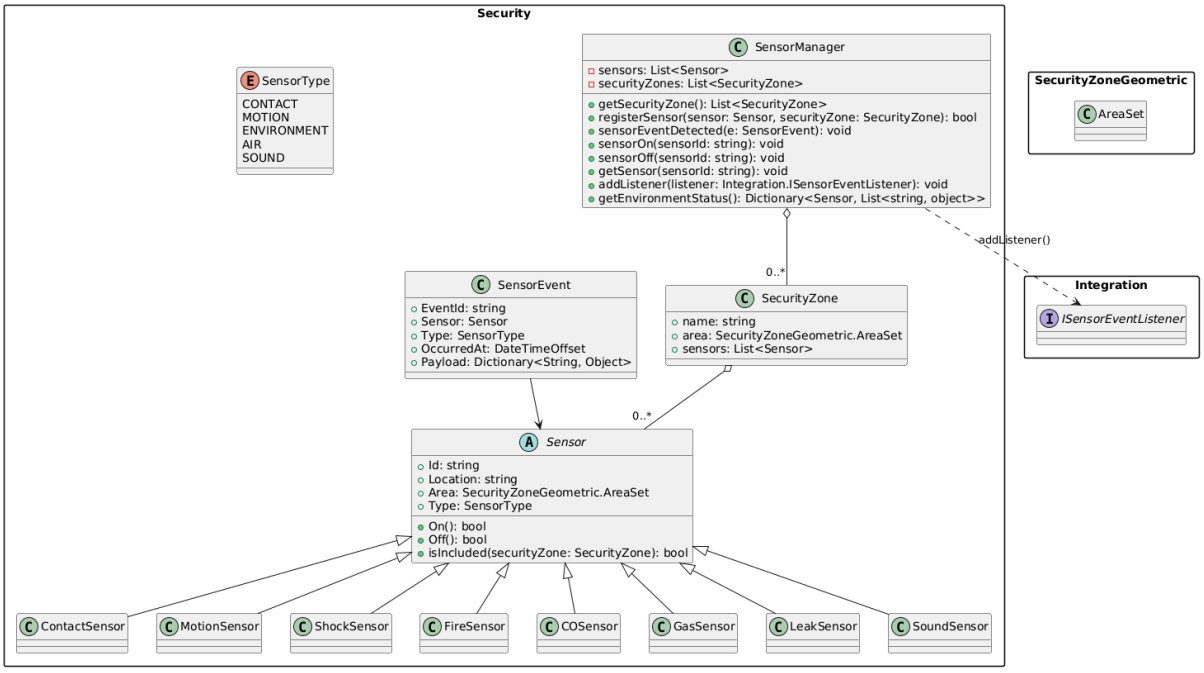

Fig. 11 - SensorManager Class Diagram

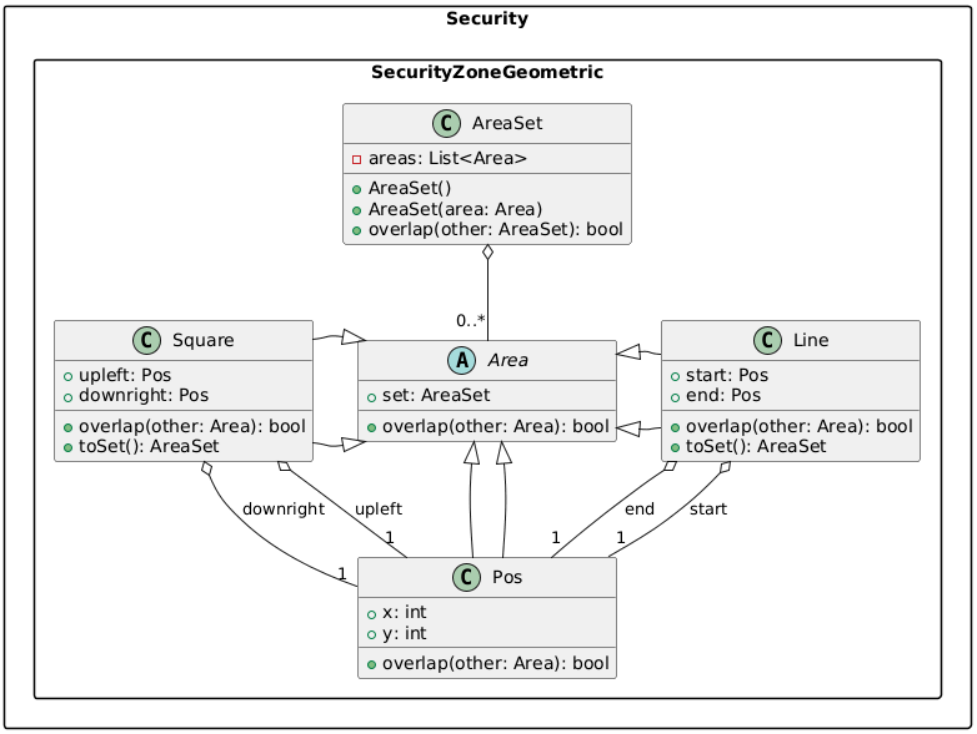

Fig. 12 - Security Subsystem AreaSet Class Diagram

See also: Sensor state diagram, Security mode state diagram, and Security state diagram

4. Class Diagram - Authentication/Authorization Subsystem

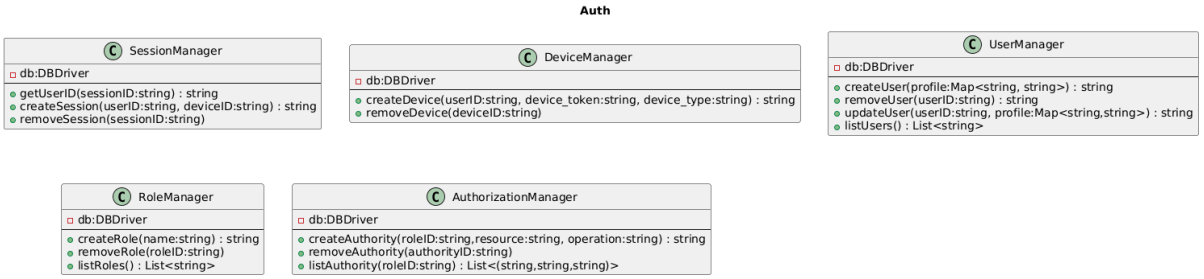

Fig 13. Auth Subsystem Class Diagram

IV. CRC Cards

Each card represents one class and lists: Class Name, Responsibilities (what the class does), Collaborators (other classes it works with). Multiple collaborators may be involved in different responsibilities. All the classes are labeled with a number from here on. These numbers will be used to indicate which classes are which in the VI. Design Evaluation Section.

1. Surveillance Subsystem CRC Cards

1. Class: CameraManager	
Manages the various camera objects stored in memory	
Responsibilities	**Collaborators**
Maintain collection of cameras	2- Camera
List available cameras	
Look up a camera by id	

2. Class: Camera	
Class for managing cameras as instantiable object representations	
Responsibilities	**Collaborators**
Represent a physical camera (id, name, state)	4- ICameraGateway
Control PTZ (Pan/Zoom) if supported	
Start/stop live stream	
Start/stop recording	
Capture snapshots	
Log recording start/stop	

3. Class: CameraServices	
Main controller that coordinates camera based functions provided to the user	
Responsibilities	**Collaborators**
Provide high-level API for camera control by id	1- CameraManager
Activate/deactivate cameras	2- Camera
Control PTZ operations via CameraManager and Camera	

4. Class: ICameraGateway	
Abstraction for the camera device driver	
Responsibilities	**Collaborators**
Abstract access to physical camera hardware	2- Camera
Provide low-level operations (e.g., open/close stream, control PTZ/recording)	

5. Class: LiveStreamSession	
An instance of a live stream from a certain camera	
Responsibilities	**Collaborators**
Represent an active live stream for a single camera	2- Camera
Start stream for the bound camera	
Stop stream and clear stream URL	

6. Class: LiveStreamService	
A service available to the API that manages live stream requests	
Responsibilities	**Collaborators**
Manage live streaming sessions per camera	1- CameraManager
Handle live stream requests from clients	2- Camera
Reuse existing stream URL if a session already exists	5- LiveStreamSession
Track client reference counts per camera	
Stop stream only when last client stops	

7. Class: RecordingMode	
A dataclass for recording settings	
Responsibilities	**Collaborators**
Represent per-camera recording configuration	
Distinguish continuous vs event-based recording	
For event-based mode, store trigger sensor id and record duration	

8. Class: Recording	
A dataclass for recordings	
Responsibilities	**Collaborators**
Represent a recorded video clip	7- RecordingMode
Store camera id, time span and media URI	
Indicate recording mode used for this clip	

9. Class: RecordingRepository	
A repository of the recordings in the database	
Responsibilities	**Collaborators**
Represent a recorded video clip	8- Recording
Store camera id, time span and media URI	
Indicate recording mode used for this clip	

10. Class: RecordingService	
A class available to the API that is used for recording functions	
Responsibilities	**Collaborators**
Maintain per-camera RecordingMode configuration	1- CameraManager
Handle sensor events and decide when to record	2- Camera
Coordinate camera recording start/stop	7- RecordingMode
Search and return recordings based on criteria	8- Recording
Provide playback entry (e.g., URL) for a recording	9- RecordingRepository
Capture snapshots through camera management	11- ISensorEventListener

11. Interface: ISensorEventListener	
An listener to the sensor events	
Responsibilities	**Collaborators**
Define a callback for incoming sensor events	10- RecordingService
Allow security subsystem to notify the recording logic	19- SensorManager (external, from security side)

2. Security Subsystem CRC Cards

12. Class: SecurityManager	
Main controller that coordinates modes, events, and subsystem managers.	
Responsibilities	**Collaborators**
Requests to turn sensors on or off, or requests to check information are connected to sub-managers.	13- SecurityMode
When an emergency call service or panic request comes in, it connects to the sub-manager.	15- AlarmManager
Provides information about security mode or switches to a specified security mode.	18- LightManager
	19- SensorManager
	22- SensorEvent

13. Class: SecurityMode	
Executes security mode-related data and routines	
Responsibilities	**Collaborators**
Changes the monitoring status of the sensor. Run automatic residence routines	12- SecurityManager
	14- ICoroutineRunner

14. Interface: ICoroutineRunner	
Abstraction for starting and stopping coroutines.	
Responsibilities	**Collaborators**
Start, run coroutine with handle	13- SecurityMode

15. Class: AlarmManager	
Emergency call and ring siren.	
Responsibilities	**Collaborators**
Ring siren	16- PhysicalAlarm
Call emergency service	17- EmergencyContact

16. Class: PhysicalAlarm	
Handle siren.	
Responsibilities	**Collaborators**
Ring siren	15-AlarmManager

17. Class: EmergencyContact	
Handle connection to company which manages user’s home	
Responsibilities	**Collaborators**
Send a message to the company that manages the user's home.	15- AlarmManager

18. Class: LightManager	
Manages lights in relation to security zones and sensors.	
Responsibilities	**Collaborators**
Process routine in security mode by security manager	12- SecurityManager

19. Class: SensorManager	
Manage sensors, security zones	
Responsibilities	**Collaborators**
Add/delete security zone	11- ISensorEventListener
Turn on/off sensor	20- SecurityZone
Manage sensor detection	21- Sensor
Get one sensor's information	22- SensorEvent
Manage other sensor listeners	

20. Class: SecurityZone	
Just store informations of security zone	
Responsibilities	**Collaborators**
Stores information about the area of the security zone and verifies whether the sensor is included.	21- Sensor
	23- AreaSet

21. Class: Sensor	
Abstract class for managing sensor devices	
Responsibilities	**Collaborators**
Pass sensor events to other classes	19- SensorManager
Stores a lot of information, including whether it is on/off	22- SensorEvent

22. Class: SensorEvent	
Abstract class for managing sensor devices	
Responsibilities	**Collaborators**
Store Sensor informations and detected informations	21- Sensor

23. Class: AreaSet	
Store, check sensor's area, security zone's area	
Responsibilities	**Collaborators**
Store informations	20- SecurityZone
Check overlap	21- Sensor

24. Class: Area	
Abstract representation of a geometric area.	
Responsibilities	**Collaborators**
Determine overlap with other area, area set	23- AreaSet

25. Class: Pos	
Abstracts information from area(most sensors)	
Responsibilities	**Collaborators**
Store x, y	26- Square
	27- Line

26. Class: Square	
Abstracts information from area(security zones)	
Responsibilities	**Collaborators**
Store square area information	25- Pos

27. Class: Line	
Abstracts information from area(motion sensors)	
Responsibilities	**Collaborators**
Store line area information	25- Pos

28. Class: Home	
Extend SecurityMode	
Responsibilities	**Collaborators**
Store mode information	13- SecurityMode

29. Class: Away	
Extend SecurityMode	
Responsibilities	**Collaborators**
Store mode information	13- SecurityMode

30. Class: Sleep	
Extend SecurityMode	
Responsibilities	**Collaborators**
Store mode information	13- SecurityMode

31. Class: OvernightTravel	
Extend SecurityMode	
Responsibilities	**Collaborators**
Store mode information	13- SecurityMode

32. Class: ExtendedTravel	
Extend SecurityMode	
Responsibilities	**Collaborators**
Store mode information	13- SecurityMode

33. Class: ContactSensor	
Extend Sensor	
Responsibilities	**Collaborators**
Override sensor detection methods	21- Sensor

34. Class: MotionSensor	
Extend Sensor	
Responsibilities	**Collaborators**
Override sensor detection methods	21- Sensor

35. Class: ShockSensor	
Extend Sensor	
Responsibilities	**Collaborators**
Override sensor detection methods	21- Sensor

36. Class: FireSensor	
Extend Sensor	
Responsibilities	**Collaborators**
Override sensor detection methods	21- Sensor

37. Class: COSensor	
Extend Sensor	
Responsibilities	**Collaborators**
Override sensor detection methods	21- Sensor

38. Class: GasSensor	
Extend Sensor	
Responsibilities	**Collaborators**
Override sensor detection methods	21- Sensor

39. Class: LeakSensor	
Extend Sensor	
Responsibilities	**Collaborators**
Override sensor detection methods	21- Sensor

40. Class: SoundSensor	
Extend Sensor	
Responsibilities	**Collaborators**
Override sensor detection methods	21- Sensor

3. Auth Subsystem CRC Cards

41. Class: SessionManager	
Read/Write session table of the database	
Responsibilities	**Collaborators**
Create session	
Delete session	
Get user id from session	

42. Class: DeviceManager	
Read/Write user device table of the database	
Responsibilities	**Collaborators**
Register device	
Remove registered device	

43. Class: UserManager	
Read/Write user table of the database	
Responsibilities	**Collaborators**
Create user	
Remove user	
Update user information	
List all the users	

44. Class: RoleManager	
Read/Write role table of the database	
Responsibilities	**Collaborators**
Create role	
Remove role	
List all the roles	

45. Class: AuthorizationManager	
Create/Remove/Evaluate authority of the role	
Responsibilities	**Collaborators**
Create authority	
Remove authority	
List all authorities of the role	

V. State Diagrams

Each subsection here describes the state transitions for classes that involve different states using UML state machine diagrams. They show how an object of a class may change state in response to events. Not every class has states and therefore are not listed here.

1. Surveillance Subsystem State Diagrams

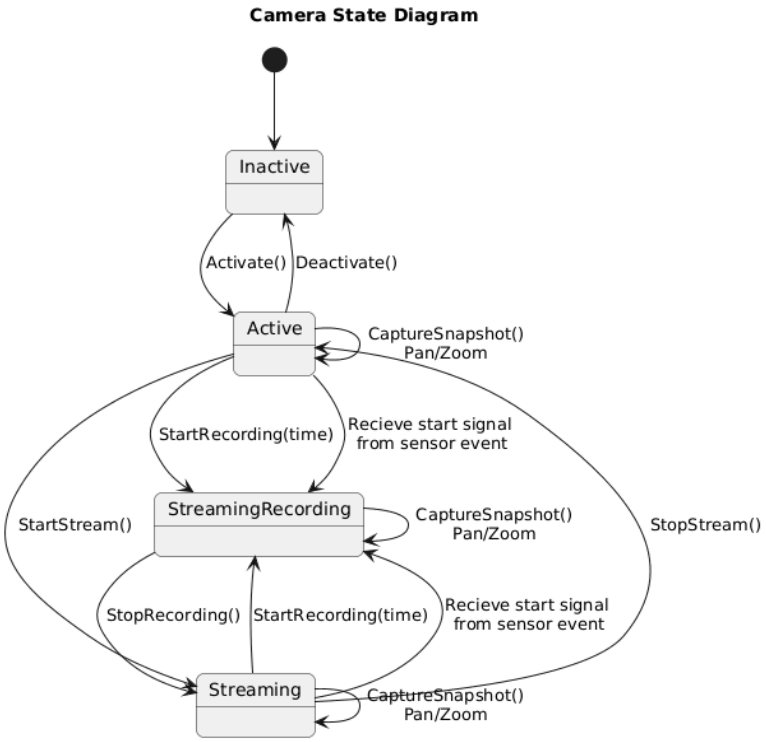

Fig. 14 - Camera State Diagram

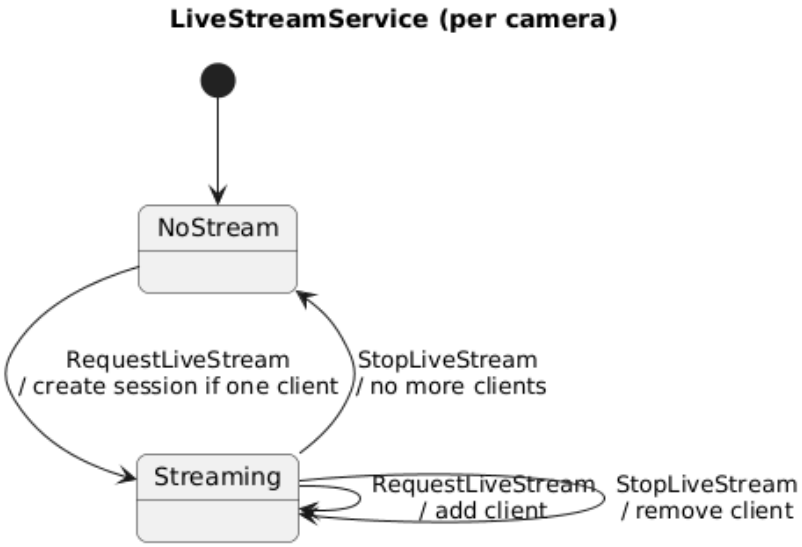

Fig. 15 - LiveStreamService State Diagram

2. Security Subsystem State Diagrams

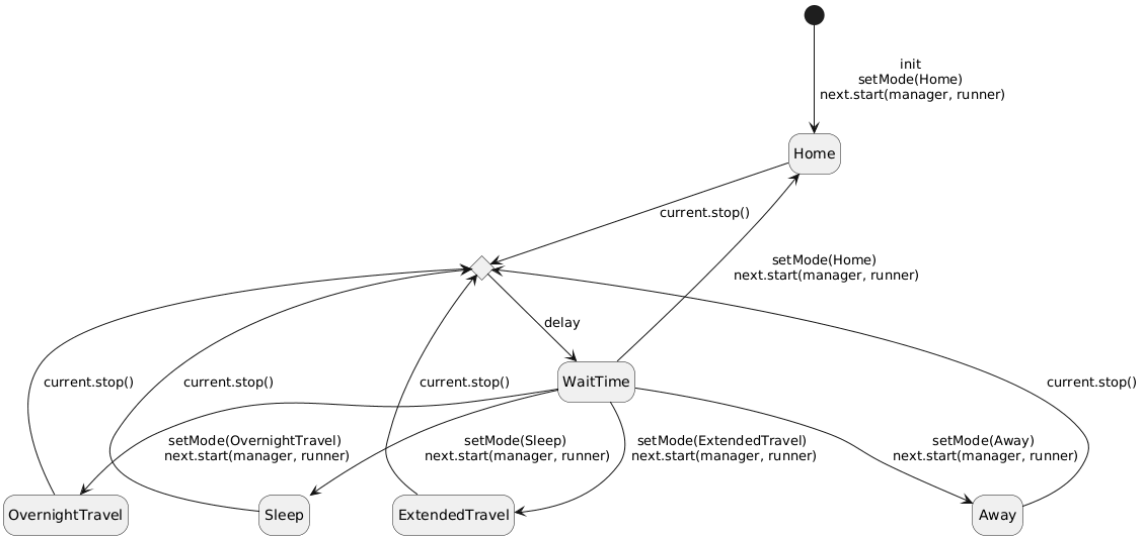

Fig. 16 Security Mode State Diagram

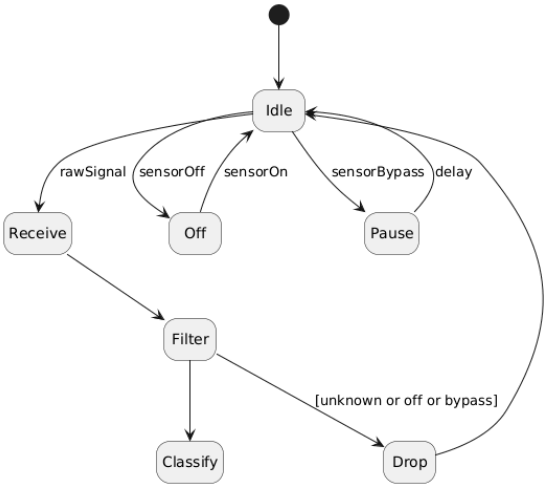

Fig. 17 Sensor State Diagram

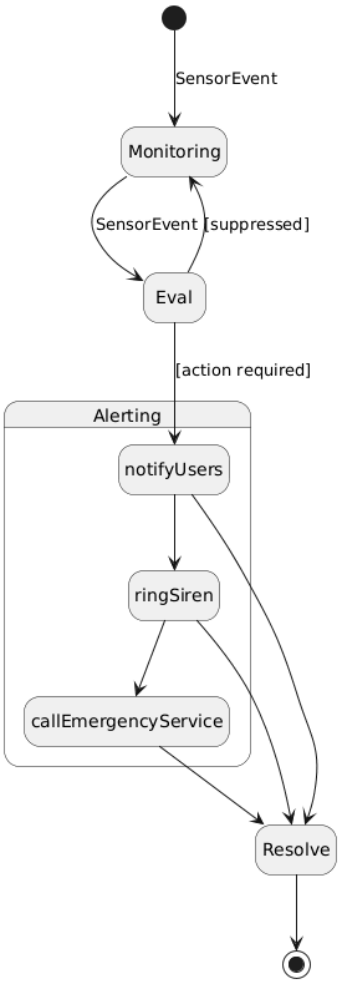

Fig. 18 Security State Diagram

VI. Design Evaluation

This section assesses how good our design is, using object-oriented metrics.

1. Architectural Design Metric

We use Fenton's simply morphology metrics:

node	45
arc	61
size (node + arc)	106
depth	5 (from S
width	
arc-to-node ratio	1.356

2. CK Metrics

Depth of the inheritance tree	1
Maximum Number of Children	8 (at Sensor class)
Average Number of Children	13/45 = **0.289**
Maximum Coupling Between Object classes	6 (RecordingService)
Average Coupling Between Object classes	61/45 = **1.356**

3. MOOD Metric

All the numbers indicate class number IDs listed in the CRC Cards section.

a. MIF (Method Inheritance Factor)

	Md(Ci)	Mi(Ci)	Ma(Ci)
1	0	0	2
2	0	0	9
3	0	0	4
4	0	0	0
5	0	0	2
6	0	0	2
7	0	0	0
8	0	0	0
9	0	0	3
10	0	0	7
11	0	0	1
12	0	0	14
13	0	0	2
14	0	0	2
15	0	0	3
16	0	0	2
17	0	0	1
18	0	0	3
19	0	0	7
20	0	0	0
21	0	0	4
22	0	0	0
23	0	0	1
24	0	0	2
25	1	2	2
26	1	2	2
27	1	2	2
28	1	2	2
29	1	2	2
30	1	2	2
31	1	2	2
32	1	2	2
33	1	4	4
34	1	4	4
35	1	4	4
36	1	4	4
37	1	4	4
38	1	4	4
39	1	4	4
40	1	4	4
41	0	0	3
42	0	0	2
43	0	0	4
44	0	0	3
45	0	0	3
total	16	48	134

MIF = 48/134 = 0.3582

b. CF (Coupling Factor)

The coupling table was made by going through the class number in each column, and for each column if it had a collaborator, the corresponding row for the collaborator class for that column was set to 1. The rest was set to 0. Remember, class numbering follows the CRC card numbers.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45
1	0	0	1	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	1	0	1	1	1	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
26	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
27	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
28	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
29	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
30	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
31	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
33	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
34	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
35	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
36	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
37	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
38	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
39	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
40	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
41	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
42	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
43	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
44	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
45	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CF = 61 / (45 * 45 - 45) = 61 / 1980 = 0.031

VI. Who Did What

All the team members were involved in adding to the sections: I.4. Assumptions and VIII. PlantUML while working on the rest of the document. Team members were also involved in deciding conventions and better approaches together, which included going over each other's works, then suggesting and/or applying fixes.

20240397 Hichan Shin
III.3. Class Diagram - Security Subsystem IV.2 Security Subsystem CRC Cards V.2. Security Subsystem State Diagrams II.7. Security Component Diagram VI. Design Evaluation
20240782 Youngdo Han
II.1. Conceptual Diagram II.3. Database(ER Diagram) II.4. REST API II.7. Auth Subsystem Component Diagram III.4. Class Diagram - Authentication/Authorization Subsystem IV.3. Auth Subsystem CRC Cards
20240905 Bumgyu Suh
I. Overview II.2. Overall SafeHome Hub Architecture III. Class Diagram - Whole System Overview VI. Design Evaluation VII. Meeting Logs
20240951 Mohamed Elnakeeb
II.5. Surveillance Subsystem Component Diagram III.2. Class Diagram - Surveillance Subsystem IV.1. Surveillance Subsystem CRC Cards V.1. Surveillance Subsystem State Diagrams VI. Design Evaluation

VII. Meeting Logs

Meeting Logs specify the attendees by the first letter of their first names: Hichan (H), Youngdo (Y), Bumgyu (B), Nakeeb (N). Tasks assignments are to be due before the next meeting unless specified otherwise. They are mostly based on and justified by previous responsibilities in the SRS homework or past meeting assignments. They follow a chronological order.

1. Meeting #1

Date: 2025-11-04
Time: 16:00~
Location: N10 Group Meeting Room 3
Objective: to review the SRS and go over what work needs to be done for the DDS
Decisions Made:

1. PlantUML codes should be shared for easier editing of diagrams.
2. The classes should be decided on first

Task Assignments Until Next Meeting:

- H: work on VI.1. Intelligence Security
- Y: work on VI.3. System and User Management, VI.4.
- B: work on VI.5. Indoor Monitoring and Device Control
- N: work on VI.2. Live Surveillance
- All: decide on classes and architecture of assigned subsystems

2. Meeting #2

Date: 2025-11-10
Time: 18:00~
Location: N10 Group Meeting Room 5
Objective: decide on the architectural design
Decisions Made:

1. Follow conceptual diagram draft
2. Safety Zone → Security Zone
3. Don't make ER diagrams for database
4. Class Diagrams:
    a. Use the default plantUML formats (C for class, - and + for properties and methods)
    b. Words in arrows to clarify what the arrows mean
5. Bottom-up approach for fixing architecture
6. SafeHome cloud server does p2p connection, user system is placed in SafeHome Hub, Control Panel uses user system.
7. User Permissions, Security & Surveillance currently assumes that the user directly calls the services without permission checking
8. Camera uses user authentication instead of password

Task Assignments Until Next Meeting:

- H: Fix Security Zones
- Y: Move user permissions from SafeHome Cloud to Hub, work on user permission system
- B: Work on how user interface uses the Surveillance and Security systems
- N: Fix Surveillance class diagrams & CRCs to match
- All: make CRC cards while making Class Diagrams

3. Meeting #3

Date: 2025-11-11
Time: 16:00~
Location: N10 Group Meeting Room 2B
Objective: Progress check and review tasks
Decisions Made:

1. Get rid of CameraAccessPolicy and related things from Surveillance Subsystem
2. User Subsystem may change User Database information on which notifications are enabled etc, and the AlarmManager of Security Subsystem will send the notifications accordingly (by looking at the database)
3. Access to database in diagrams will just be marked as box “Database”

Task Assignments Until Next Meeting:

- H: Class Diagrams, CRC, State Diagrams, Security Zones
- Y: finalize User Class Diagrams update conceptual diagram, design database (ER diagrams), Session State Diagram
- B: Architectures, connect lower level Subsystems to main SafeHome Hub, and Presentation Layer to SafeHome Hub
- N: Remake stream/camera system
- All: List which classes has STATES and requires state diagram

4. Meeting #4

Date: 2025-11-13
Time: 15:00~
Location: N10 Group Meeting Room 2B
Objective: Rethink the class diagrams and architectures
Decisions Made:

1. Database will be abstracted out
2. Architecture diagrams will be changed again after class diagram edits

Task Assignments Until Next Meeting:

- H: Finalize Security Subsystem
- Y: Finish the REST API
- B: Finalize
- N: Finalize Surveillance Subsystem

5. Meeting #5

Date: 2025-11-14
Time: 14:00~
Location: N10 Group Meeting Room 2C
Objective: Finalize the SDS
Decisions Made:

1. No more changes in architecture and class diagrams
2. REST API will be connected directly to the subsystem classes

VIII. PlantUML

Almost every diagram (marked as Fig. X) in this document was made with PlantUML, and so we will provide the respective PlantUML shorthand codes that can be decoded at the bottom of **editor.plantuml.com**. Each “Fig. X” is linked here, in the order of X.

1. Conceptual diagram
   Made with draw.io office icon sets
2. SafeHome Hub Architecture
   VL91QiCm4BmR_0VXUxvGOj8U2YcOg2M7qiFQNb9HMJ8hsU2K_hsoKjUHvUI
   sZ6PdTcUrDHR8TerCapHfeVg24xAi83IeB5YX5Tl3W9InxpHXx7sdbIKj2r0eFpnon
   39lMrV_11Ayenw5bA0gPBmhpM0iDfFCNlJ8BI4q7AaN5LwqTj0WmHzaBjW1Aqs
   rKAUPkN32gYDXXzXbUda1vPWKTUQT_LphCiRjFJSvx157wm1wCqW1SD0oP8
   pociJPnnZCi1hmPlFWH9xuAykBPyURQ_zqAz4LOXn8J5hiFkPIWuKIZ0jcgh80z1a
   OJxrvNgzzrRvwdOYqF4IiSjVkVx0xRbV3Rr7L_iV_0W00
3. Surveillance & Security ER Diagram
   Made with ChartDB DBML
4. Auth ER Diagram
   Made with ChartDB DBML
5. Surveillance Component Diagram
   XPFDReCm3CVlF4LUeE8DfaxTSPMcSXlbu0MlpKJiod6PzlOB0rBQGdSBziz_Vx
   8E2g9dtsHPkG6F0jNus0B10QNARWrcjNeoYSxALzTYeOBW2mp1CLNNDccTnN
   dmMcecgWTcCcJb3YrBxUYmvJhANxvJ1uD8k4qpwtq3oj-P_PwCVGW-MtKjAEOl
   2CDp6fFAc3H-P-kL5ZjOm9QHpMBstbzqMyuBvPEH_zMIUOSd3avPVgRmn3hW
   UXQw2CTfW1tUHja41B77wi7-cHGS6KVn6icYuSyqKTF4eF0NT7UPut44DT0qAI
   GiW4ETFHHf_lX2gmaUHuLK6ylU0zMXA_y8Kd-3T8jSHr_yLtu1
6. Security Component Diagram
   VPDTQiCm3CVV2xs3Bv0RZD5HoXYiCDgtZJmOHpY6H2goFCXEloJEg9suUuka
   dpxybtBmebb2t-rt-ztN5KnW9yEP6gX7-zG3Mq0p70b56vby7-tWdT7TIwUvdnppnrc
   Zji2J_UviAubz1VJ4cMFficPEtu2I4f-4S0_NounfELLLpwh09_OVifJ8UgrY3ylVS_F
   Hq_W_BupOLfWIjPRpKSjyKrbMH4hBAs2RVQRYlGBr8EnChPwgggH4cIglOnus
   Tdqx4bCGXt09YF6b6rz4fonbMeqMr8t9WFSERN8MwH9Kk3LQDaUGJ9Z9ftx6e3
   AZG4pTV007m6Ry7Nu1
7. Auth Component Diagram
   VLDB2i8m4DqNUeTUe5UGYeehDughuYBee84qWMHIgASthR59cp7BzvaPyXvf3
   3ADjXDbKHRj1OpXIfwOP4_GjudRQM08Nd5KUmIycabjpz2fFj8SjOUUtu7Oy1N
   2r8mTC-uZHfPdF1gn4jI5aGvsZnpUMdmfpTyCwRTv7OSmlA0EFHDsvQFYgkfwK
   wLvaQvrShGhH9qa6nyBqLAUVDpndb_UgPvh8DumLv3egrB-rH8lMvIGSHELPTn
   qcmt8n_GBFW00
8. Overview Class Diagram
   bLJBQiCm4BmR_0yYz-GJGaAQGo614jDBRrjbSKHbAKYomItzz_eX9rf9bnurEpix
   -x1tre5nRIFBmdu9TGS33Ri3_u8QAzTBP0Qv0rLBB8koy16sshVknghsq_RMORD
   YFsN169TWBQlGMg7L6TJ0D4_y1Jl1CO7VBPe4l6YPCiUQseXlSA7yB_5LjQP38
   IKeZf6_ifXb3j2W0KQV3xco801RhxUUybGuYMuOZa5eVA-vW1cx7NCIDj5CKgP
   efaAU6Bb_hL1PROuTAdSIG-lAjt11hirLg3emvaUzHCxwYWb8f-IJRK8BL2apPpB
   EeWPd6rdEltH6Bz5j86-DS7rq5ylLpi5m2iUXN8sAzmUj772tu6-ttWeESY_1D0iuFH
   xUbdkeRox2vWqbeA-UuqxRZ9aVMY6bZRqIyMaUb1--V7XE3IeHF9RSRBOiTsH
   FQLgSz9gZq17b67Im-JBXxaB-54xNc52YZUw7pwn2EWcAXKEThco7wZhzHly0

9. Surveillance Subsystem Class Diagram
nLTBSzis4BxpL-puqCYrPJUPdZYf9wciYJNrQooxXrmoCBcIC0895e3aSZh-xrsG80
cGbDrRTN68VUz--yWNRPWokqn4UilpWYcMWU0vch90aCfif4TGSfUdCwboLF3
naqSeMB9bQroQKY0eJ0pBrm8Z-ds0tmO_cB4C5OCRbfDGXhaPOOeImRHkMF-9
W7udi4XZRHJFrzNt56w97BuizDV4y3t6pr8AztR5zFttxk44xba-sJCH0y_DSGnxoL
D7-Ivbvf4yIgMM6Po40fV8nWbBczYbKSWoI_6Sj3HPTAI-q0CcKgN4FJ6SmXiwQ
SKx9gAr7uvXnWgpKxZCMQ4tqbYMvzAWZbuX2X9Rbq2v_4uXgLvr39USef8vK-
NdslDJ9t7XJ5npRUeNRTNRJyVPS7n3C-5f4pdnrEpHgqL5Q5w-vFeygpuqC4Ya45Z
P1xboh3eAHOu-7e6SJk7ebxEpduyQdCJWj1o2NfrryWScUvwY19cBylWm_gu9IhN
C4hMs7eMfWZQiEcXTyJufCQWu5NOK1oq2A2zKQLGhbY0iQgtV25Slh2IZNjP
WfobDPeEmTcGZmJ2nhVxISx4fDK-OW1JtF46NEJWxk-ZhZMjTJeyrCMV9PYHg
hk7PThyrWNyJv4L9gSCc-A60hSzqUb7zROkJx9IoxPuqAVC2NcAkfPhlYMx1X7a
BonEuoptgH5V_Nbnwq7BDyHe1XR14PSEcL33Ju05qxPU6dsX0hP3S88w4GZ8Q2
ZBbN1fMlTRLLBrHCzeXWonSD5WHd1oVLTP69IaFJTiyu5yxrARJDQbTM_YjD4
3Hs-mDlZwCHCRMQuKM05wE61LQIBdLi2kQcG1xpiAsZORAUbqvX4NS6XcHj
W0Dkpo65ITVN74EDNKx_gYu5KJqM4D7cE-oZlT6flY7LLwNTdPt-xYuVRfxMlx
uzEjLzJJ_Stxx-EFthylvfKrU3Or0lWMeLHGFTJlge-9hYgR6S5UOjk1MwdAdc4L5
DUxhMeMcg0CJcQ_ucWPxIkCSjjP0o6CJE0pGDNd5tu_2twoXMxCUZFn1HXWc
m3pITdAlSnj-xormv71oRZ3bx4dnrauOUpViCXkX3G88Z1aQbZB3kWJyb-2xSTCo
Fc0XDJTIbMqgbiXKiXdqqIdeCFPJG3_iKtlvaD83uL8RBkQ4GDBvBF1pIp-uBljB_
Nvd9cJ18tYTxIzrrqww3QWzNVXWA0oEc4CizcJYNYxUE9_UOXlsKJXXUx2ncs1
qO0PSpPZjNKzj0HkXxfm2p-jfxvuQp4_ds1LDHO7lhQ0QG2RmRmHAsH0aVHt_1
mZl1IkVwPYVg8PoUEtqZyoHlUEEpkvqQQ5PDMKuw-coe6L0rqSI77ojoBlR8RoF
hkLwRSU2tMLUaxziaAuPKwTHdsiglOOsMktVIcCpmWOd5GOY_hXgq9p8whjQ
goqh55BmfAGjHHU7heuSkiu6ieD1rWonUlw_8n0u-wwTjnBwjXrxpb6zAC1pXGG
4skYZDnm9zHN5T9iA7JJKSHHz8I3R_uZ-2m00
10. Security Subsystem Class Diagram
ZLPBRzGm4BvNwd-mbWCfxLQSeshLfzILYee8NBWWDvdDcYRsolOk18I4AiG9
YJlYm1M90nBygh8_Wh6JxDf9zj7BgZFVFFpDV7QEbQPIJ_DiTqVTCZwXakPaG
kDRcaAaYmo8X5XJdcQmkxExKtb81F5KCbsGzyPA2ECQv8Z6G8PdGegfPXoUJJ
a7QH04__OZKoY8Qsz8XXTyceEaMiYza8YRrrXeWHMJO4nvaa7jGSXCiCG0FfG
rwOtIaiQQn1bLQj7I4v70eYYdEMP07EDfPS88DeC8k19IXUIIgd44-gZ434fG_vMq
pOUjuzH-lxdI-g121ZdbI196Bteg3NlUeQj-x01MXXoGLPquXDWio5YRevAQIv43
WRSQUEBpj8ISlA751aYK0KmsOAvd83bBn_gvf3F8Dg0lh1UIjM07ZiIbmEFYJ71j
1bzFEQPP5kIW539MJtiL5q_7XM8OS99HcRiEQwXOhpFteL4oR3LH0HMJmCDc
zd87XMQZueK2gTOqYK1ZYKoUmBURuotAHY1dB8PjZtjbXj4urRw4b2aKh7K6
oah1R9xvFI2gel-bC2gDdF-mo8qGMPKbCz7NF619zt5A_sZK3V2wB4jxRJRydYW
MupcRIgHBcqK9MpjcnZT3xvL8KUVd37KlE9N5KMHxEI0NbRSMlN8IOCGLSdJ
a9btYd6v3l_aAaZcqXzuGAW1rjYtqTg--ffB4r0porjdyAQHWZn-uDzrU5ndxgXlfJ2o
133Hi04sg5HzoBQSAAJt7c5X34a3DXINMZm8-OrBm71sNz1sLIHc5uMl3yC8x5q
YkqDiMIaHwIYK-PUbMy7WCyQsr1tPPXkwkTkpCjRtl1_Mf5fS1IhHMPZkFonxb
B6xekzodqsA2grwfqGjtfbrwKNHRmL1xwt6uFAw-bMdCMtjZ7RZWCjiIw3sR3eIF
XCofdIzRGQcgA1LBIYlHFoJppr_ctt-GVx-_aldTh_ctd-J-xyVxFv-CjtaPY7w_hS4k
K5E6NPYc4himlSU7XuzwBObv3uZ1TJqLBL3hnV2-O_hz0UdPBmIyb_69xtLya5Z
IZl7bj5zn_m40

11. SensorManager Class Diagram
ZPNTRjem5CVlaNW7gpUZeqIxZYfK1AoA1CdKi5tipZY7B2EnaUsWeQdIBlP6srF
rAMR7Jh1RiZPNpZa_-tpuVv8x8J6NTLKE1s9Vq0Fck487JFOuXrIUIa0SYCGqBs4
u60wi1wL0Qbx84_gfhGW1hIjbf8BnpUa0sepKCq_YpMo-CI_hP1CbiLaluo_HGnAl
bx5rpg87iqYJp_52Bn_DoQJ4GjYZbqUWiZrxtBn5MOY4v0NDhTMGeTrXZJgdqCd
FcXEYgk2GpMI85bZ2fgWWsUq4I0jymgUIOHLYKH1PC8hvwJPjejsWPFjTTMPw
8LNJcg-CTdqOKrp1iqGn1nnwz3sm2XH2WfdofLqMeibRX6XL27bhghWKLzlNcA
hxuMtWIU_UndUEVIREDikDSv2kUNJz7vP3hbp0JQIHy6xZ1haX_UBLeLl6Ig_av
eON85MV8HjrDzYO5NzaHURn2RK1pwguYErsht1DnKu1VHpECjq9eC17fLs4AA
8ISewrO8B8IRb5NyHQqcF16QqKaqeiQw5Rx4hETg_fkh3wOqP_hX3mLZanaPuY
EWs-69KL8qsIpyopDmkpRwpeoBzDtN3VKeY8ahBErBM-IHAUqkUCIbMbBGz-g
APcmXbuGwsPhir2lLJwZP5z2_LI7miE7THBpPCpqq_TO-5WVTGAyFxrY2chQTQ
RlURUeQU_lvx-_DPh_ml19XDqzI48tbzvJJUWymdpk3Qy-psUJAQTv_7IJyAJa96Z
IRzJaxVD6Hy35yh0TvWeruLfihlRO85VPH1CNvr7529taA-7WplLn-QV-0y0
12. Security Subsystem AreaSet Class Diagram
hLEnJiCm5DmZvH-Ukj0MfO8rGbMPM3eWPMDpqqThDRM3_GAjW04XiQ5E_0
C30_z5-m_OJfesAIqCH5byTpcVxpazJKnHDaryJq-uI9bYKqXPF64ZZ6YU82YCY
ObHWhxdUmK34SQPuZI7Uzy3gADNKk05oYcIuh6LM1406sXIB2Q84wOrd2ja-V
VsEPQtg1AMDYMDKOMEROKma38f9Hef9oAa77nSUsyi3RTn3PX1T0XzhkdCijt
IhD0sMpk8_QO8S2YTaLO2rh9SIWqu8nJ3dQFEGk22okMyklojYDe-qKrc0k_RAai
Jl3QjcJGbDfHtGl7Hk0R_OG0aNME7Xz3dOcySTzkgsnhPV-SuWjNYO_Ns3b-V2r
W-ForVNdDYVJza443ZfDDfDnoKatPWmKDt2ocghOBkU3_gdEjfmnc5a3T_G52E
OSktgd1jxMLDSPOpfpPlpwpSN_mD
13. Auth Subsystem Class Diagram
bLFBReCm4BpxAtnqrF03g8hIYakavDAgvyg5BRKADbeRfBJglnS_W18UJJXWOy
_CpgpDNXkEfYuBOeGfW3xKveEGjE1QqsVGMYXvufBdWFIRqFQ9QFOM9uy9
YWQGH95RlACvc1SDU4YOzgn34ck3GkOR6bC_2z0KWHi8ugnsh83TqWmQaS8
o6Q5KJK-Ur28_dVd42LtbtHloX7a_hqPzWfmidYjOznaqBqCDBctFRl1euQn2zIuA
Y4-ykkyS-N6tOCFHHi4cmBhAElsB3ioKMwfM2ErkWMPswzX-1SPkIFoaYbjEnS
APvEK_ZNOmR5z_cZnuihjhdkobLoY-kBdskdjpWMVEuz9R1ArgJA7leQe0dVXw
Ya6FTxFPF0jr-u1iT4vXsBZ0P0yoix_xBm00
14. Camera State Diagram
dLDDJyCm3BtdL_X6XpGXE4u23ROB9n2zmJYOrkiY6YTAt07_ddGTwSQqKP6J5
RyFFyMP1a4lZQbLUDViqAC1RnikPzOpURYyKag9bffWXeOyGYue17EDLSGgz
NB-2kFn3Tmp5gBNfDGPNBS7xZ3e0fpzY3AX9OMkeNwW6zvjLqww8gAoaRhjs
maxWJbX3qakkNX2exc2MY-f-2hYf9tDZaeFchJpU-akeaxVsHic5-iI9dazKM5zkU-L
6UG6wz49qvwsTKyNcMX3luVu1wz77gQrhgVzbONMn39-mq0bh3K2VGfvnXe2S
R1-QCXOwoW5czs2eAjMOi5BRyrM2JPEGu8Frah97bnSKz6M0yWA1KeBR0MA5
N95tIOUvaWlFqCdZQUSqOMLbMoqu4VaYsThpR5nXxAElD9ksbCwQafSjZ_r6m
00
15. LiveStreamService State Diagram
ZP4nJyD038Lt_mgF1QcgO6H0bReYXcO41YldmgavExMTyFUv4hLfaP2w-Vdvyv
FbjGTP3AK3tsVfoQYWwI1feoPi-F1uOVWN9VtEyeaxwfm18aV7-997RiAOIiCsvf
PnqLUqfS96Tm1lzn-uN3xZgqvJ03KqUDQJd4JT3d1oVh5p7vzmouU1FUR4TrbXM
ulAExjd5SmxL65ikymI03D-qp9AwGQq2UslEUEYut-fvtl-adX4HR6edLW7MBEauq
j-0000

16. Security Mode State Diagram
    bLJ1IiD04Bq7yWzpinNh1vHIwa7me7XemOFri2I3MKmsOREjpQtIwa4A2XQ4qeB
    UH1I2Kkd1BsgIVp0RJMcAOfFJPjwRDtblbjHSGRXeMQQgR40TjHQdmZkmTOH
    zuY5N5OF6PxGSuO6EhiQf8wZD9DizeymXd5XWKeR2Sn1iBWnRLLJbUFC4IgK
    gxDaMGXaeewB9N1HIkI2BnIO3XXsndSmlM8IHK-HRm5kC8IzAZHWHc2Yavvr
    pugNdkedef2-7RUICdXgYmKaRpRIysn78TDJJQbgNNr574pLf1IeLpR2fXjLgYXy
    HAXfKpfK-W-Udu6uCOVyXwaqW70t3jzTmqeNv-pIwca9qyn9EPj7rB7WSGp3v34
    T-DBXKbSIwp20xhGnnnXoPT6uxXTYfTBKMCR6y5dCbZxLwic6jrQ8gMORiIRE
    BwJgQSbyomxdVZJCCxsyX6FfnI10Ckc7luYVR9BcF_dpcgygYVt5zbfyoGAvRb55
    BWlv3JMBvrBwtyRjQWkMJMrtPijmAbazuPQyPuIpshsuj9YU_YYy0
17. Sensor State Diagram
    LL0nJWD13EnNsZ-uGqAw3wH0KK1852Y8b54Aa_C6AnllQNSlqNKq_8EA0b7Iy
    2Nk4_WsEX8QMnxFs6DFKyQOsvstvY9pzWGBaXJYFGfkA485B1ctPCqprmIqQt
    87DQLrv2Pp46kiMLwkeAgkuAxsoXjYAHzfJRmdc434mu8tWhfa10V2BVjCqPfZB
    j1D38qAbgri9Hm4GeJWt92UkWPJMlrZNtk5s7NM5Dt9Wnu3Fryl_Vidz1-l_TktDN
    CTCn9qGofNpkNCxy3uQpXtrdd0Dfr4i-B4cYCwpfpeLpogbIb9NVxv2m00
18. Security State Diagram
    RPA_QiCm4CPtWTxXja0XBv2Xf4F6B0sTGWPZNnE18xkIR32ba92az1-qTCaIIcaf
    T3Jj6tMqdNUepcbg9qI3CD_zzDsdizhQECf4Ov-pXX76H-YX6obXaguZdI4gQ88C
    Z3XFuEVw6PIGm_B3TNmVt94Z96SZuI7WE3G9UAXT9K8Z0gjprZyQGBDv3Dr
    02XFGuPRrbpfGdHYb8QG1nSiYV_tWh8BACvtOyU40ldwypT89PsLzrnpArOA-Zi
    9GeTRe3MhWvXvd6fNUWevBKK7XHIHKINDc9s8GJdnKndfUSWRGWFp-C_kU
    Gp6RhcShaZPKpR8IAKhFsibAed5rngY6ADsaXoeMBb9rOrxSh8hv2efbcdsb9DMJ
    Kl9_HwWMSPLErA4cn75sHHiDBqkdVrSfbazbpxT5_lvWzKb-P__1VkDJr84VOwt
    fRilzEX47OzHWIh8zHx9z99orKNhbA_m5