Software Modeling and Design

CS5373 – Model and Software Architecture **Spring 2022**

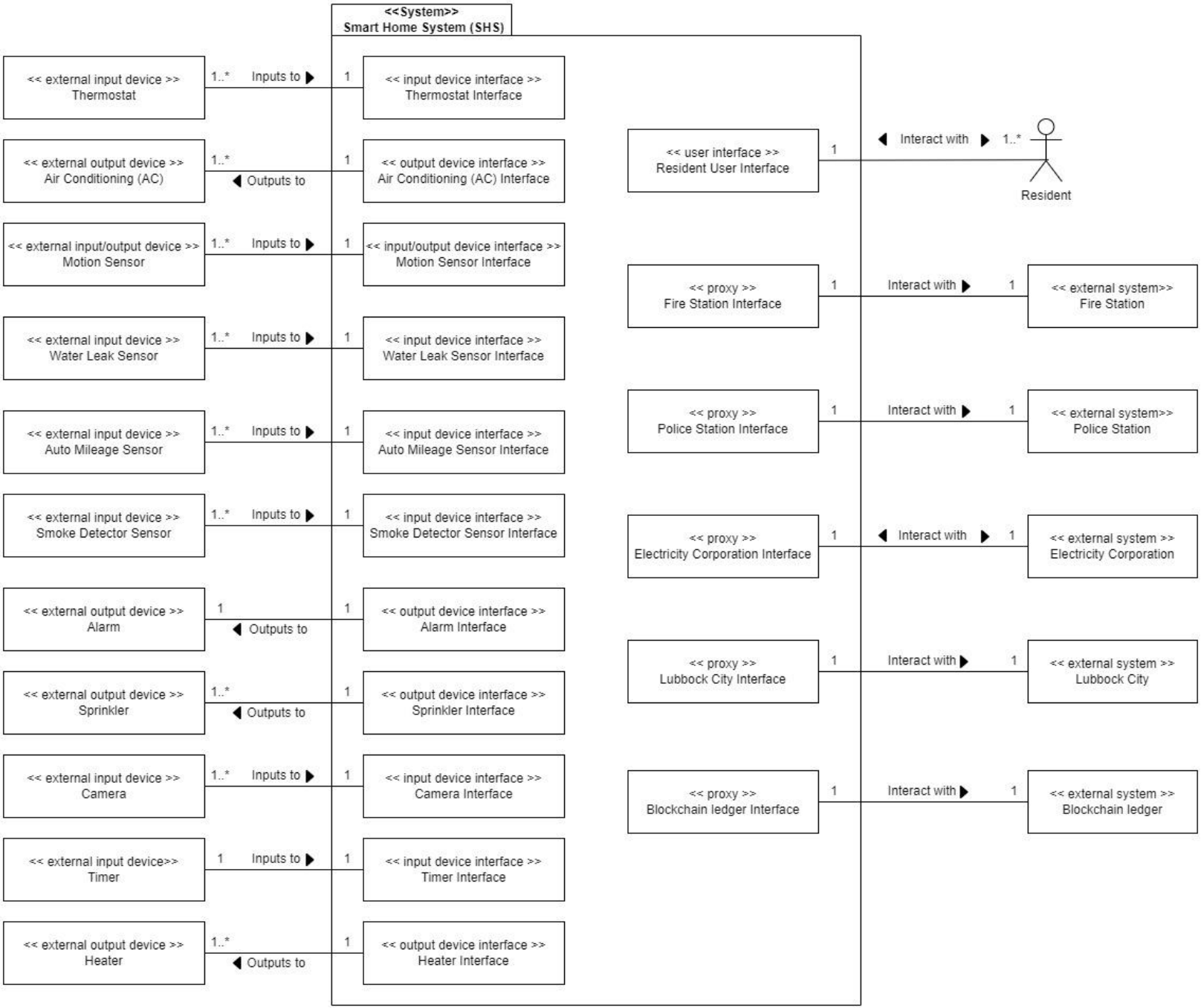
Term Project: Phase I

You are required to develop an Analysis model of the Smart Home System (SHS), which includes:

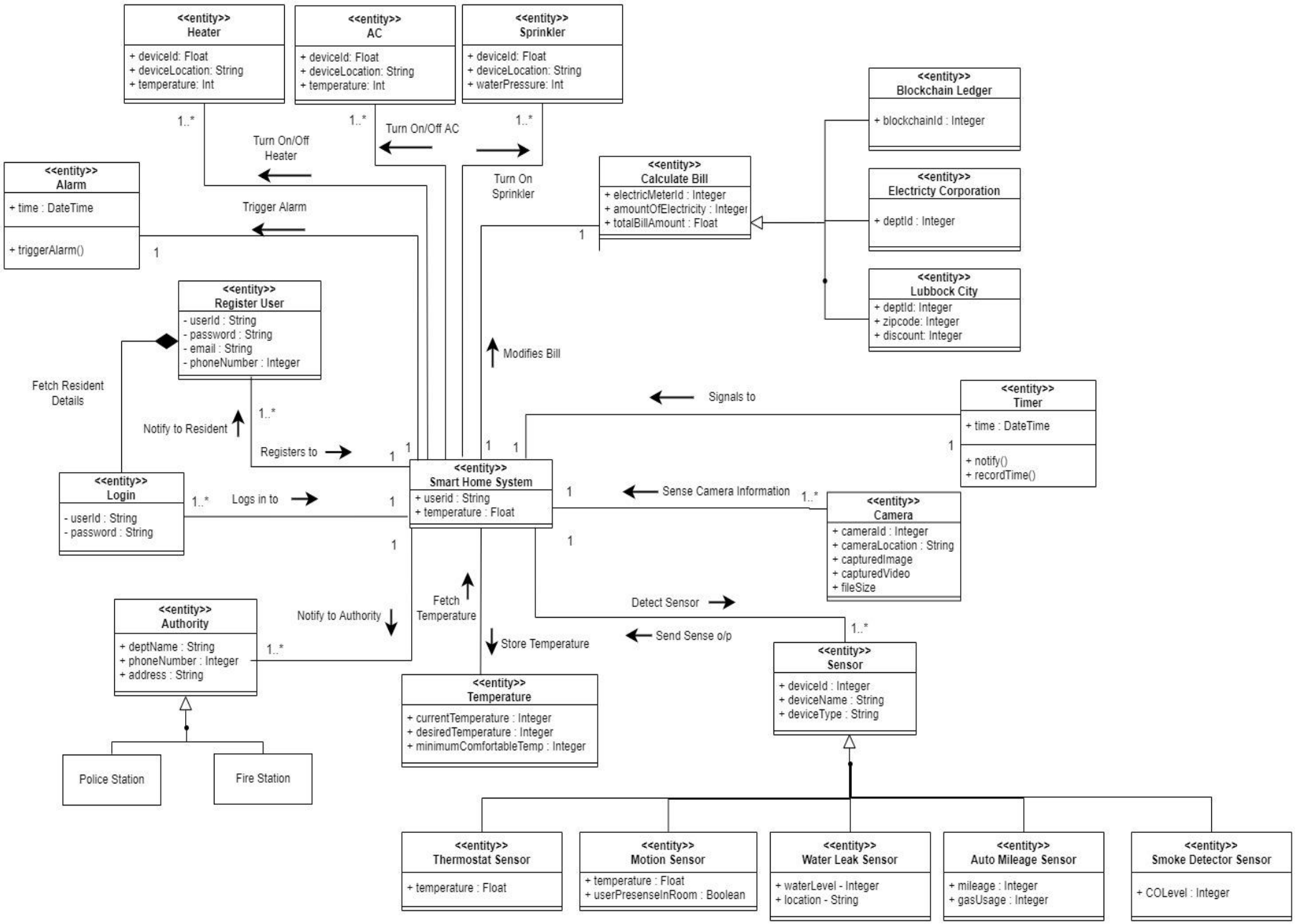
a) **Develop a System Context Class Model depicted on a class diagram showing how the system interfaces to the external environment. (5 pts)**

Assumptions:

- We have segregated input/output (left side) and external proxy (right side) to showcase diagrams properly. All Interface and proxy entities are mentioned inside Smart Home system.
- We have considered Many to one relationship for Sensor input devices because there can be multiple input devices for one interface. But for proxy, we are considering one to one relationship, because there could only one electric corporation is available in that area which belongs to that house.
- For external input device, we are considering << input device interface >> and for external output device, we use << output device interface >>. For external system, we are considering, << proxy >>.


<<System>>
Smart Home System (SHS)
<< external input device >>
Thermostat
1..*
Inputs to
1
<< input device interface >>
Thermostat Interface
<< external output device >>
Air Conditioning (AC)
1..*
Outputs to
1
<< output device interface >>
Air Conditioning (AC) Interface
<< external input/output device >>
Motion Sensor
1..*
Inputs to
1
<< input/output device interface >>
Motion Sensor Interface
<< external input device >>
Water Leak Sensor
1..*
Inputs to
1
<< input device interface >>
Water Leak Sensor Interface
<< external input device >>
Auto Mileage Sensor
1..*
Inputs to
1
<< input device interface >>
Auto Mileage Sensor Interface
<< external input device >>
Smoke Detector Sensor
1..*
Inputs to
1
<< input device interface >>
Smoke Detector Sensor Interface
<< external output device >>
Alarm
1
Outputs to
1
<< output device interface >>
Alarm Interface
<< external output device >>
Sprinkler
1..*
Outputs to
1
<< output device interface >>
Sprinkler Interface
<< external input device >>
Camera
1..*
Inputs to
1
<< input device interface >>
Camera Interface
<< external input device>>
Timer
1
Inputs to
1
<< input device interface >>
Timer Interface
<< external output device >>
Heater
1..*
Outputs to
1
<< output device interface >>
Heater Interface
<< user interface >>
Resident User Interface
1
Interact with
1..*
Resident
<< proxy >>
Fire Station Interface
1
Interact with
1
<< external system>>
Fire Station
<< proxy >>
Police Station Interface
1
Interact with
1
<< external system>>
Police Station
<< proxy >>
Electricity Corporation Interface
1
Interact with
1
<< external system >>
Electricity Corporation
<< proxy >>
Lubbock City Interface
1
Interact with
1
<< external system >>
Lubbock City
<< proxy >>
Blockchain ledger Interface
1
Interact with
1
<< external system >>
Blockchain ledger

b) Develop a static model that describes entity classes, attributes, and relationships between entity classes. (5 pts)

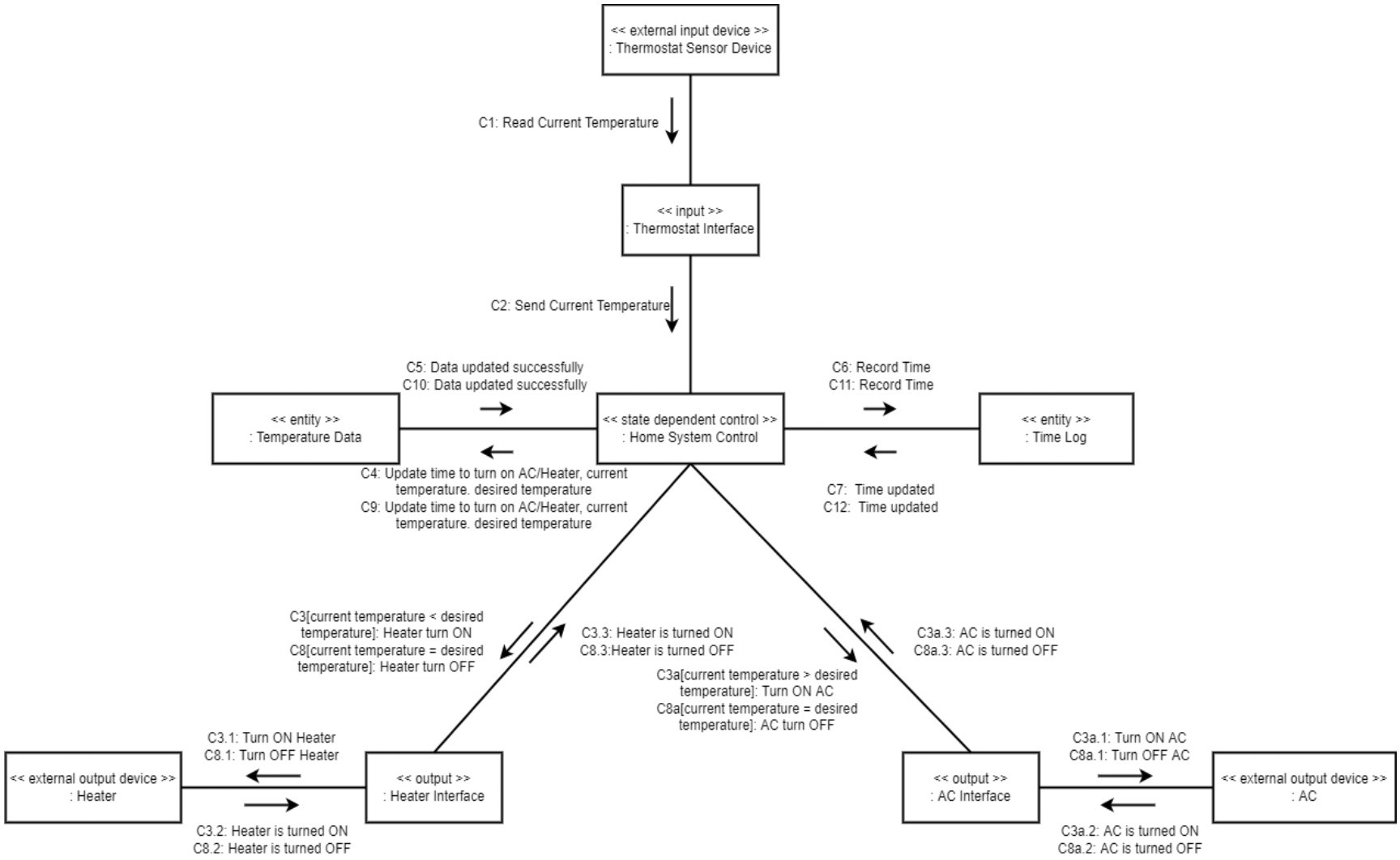


Assumptions:

- For this, we have considered Smart Home System Control as centralized entity which will call all other entity's available in the system.
- We have used Register user details for storing information and processing resident information. And we give composition relationship for Resident login for fetching information.
- We used multi branch generalization for showcasing inheritance for police and fire station with name <<entity>> Authority and other generalization for all sensor devices with name << entity >> Sensor and other one with Calculate Electricity for inheritance of Blockchain ledger, Electric Corporation, and Lubbock city.
- We have considered few Associations for Many to one for Register user, login and sensor devices section. And, considered one to one association for Timer, calculate electric information.

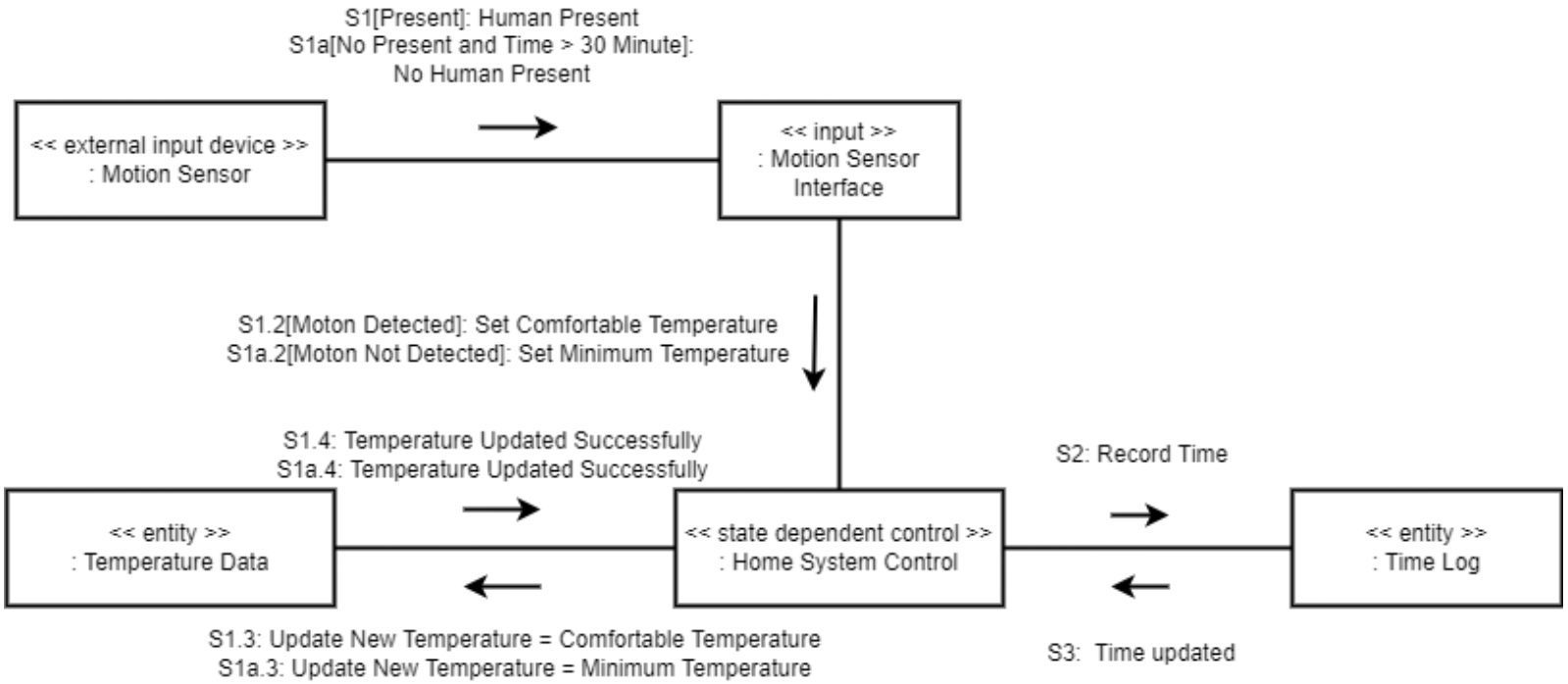
c) Develop sequence (or communication) diagrams (one for each use case), depicting the sequence of interactions among the objects participating in each use case. Define the object structuring criteria used. (12 pts)

Use Case Name Control Temperature:



Assumptions – Home System Control (state dependent control) will check current temperature with desired temperature and take decision accordingly to turn on/of AC and Heater.

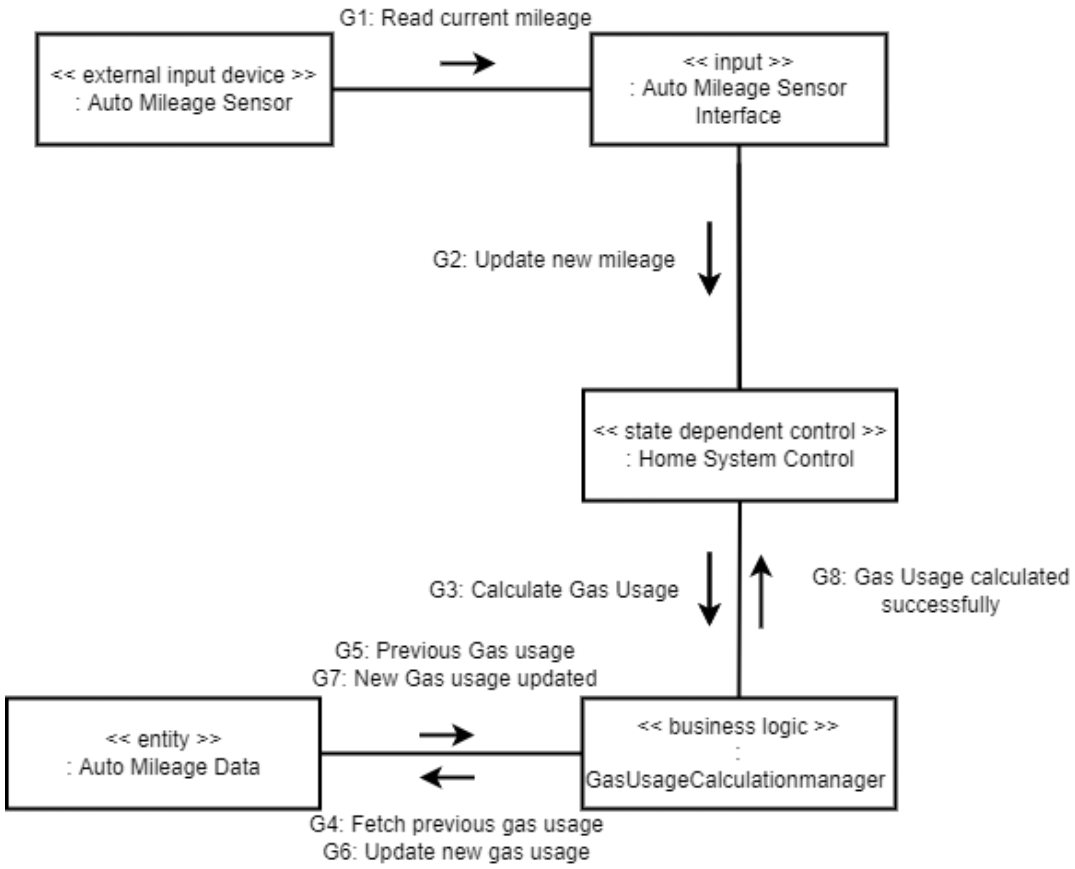
Use Case Name Set Desired Temperature:



Assumption –

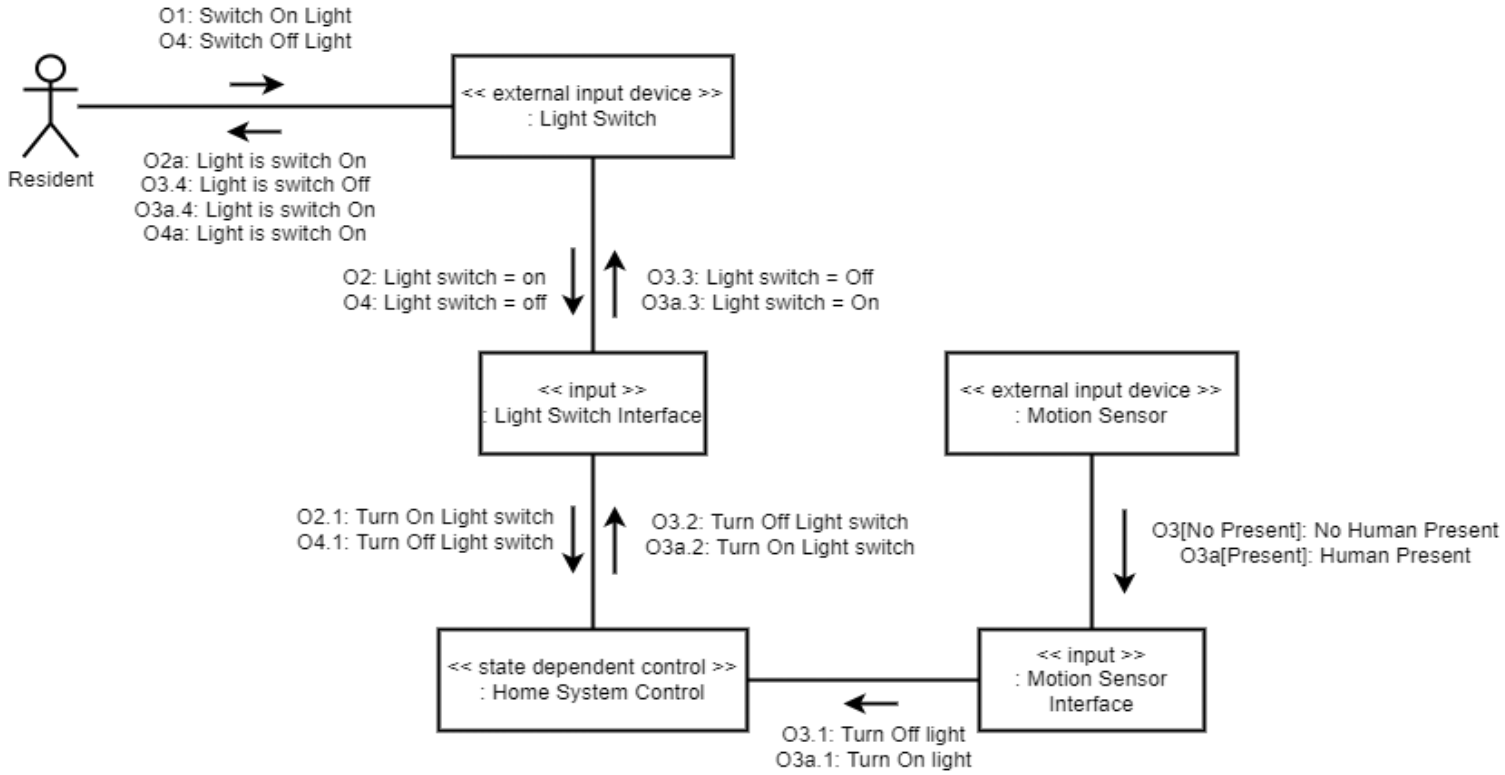
- We are using Temperature Data for storing temperature related information which is used in other user stories.
- Time log is used to stored data stamp for each activity is done by state dependent controller.

Use Case Name Record Automobile gas use:



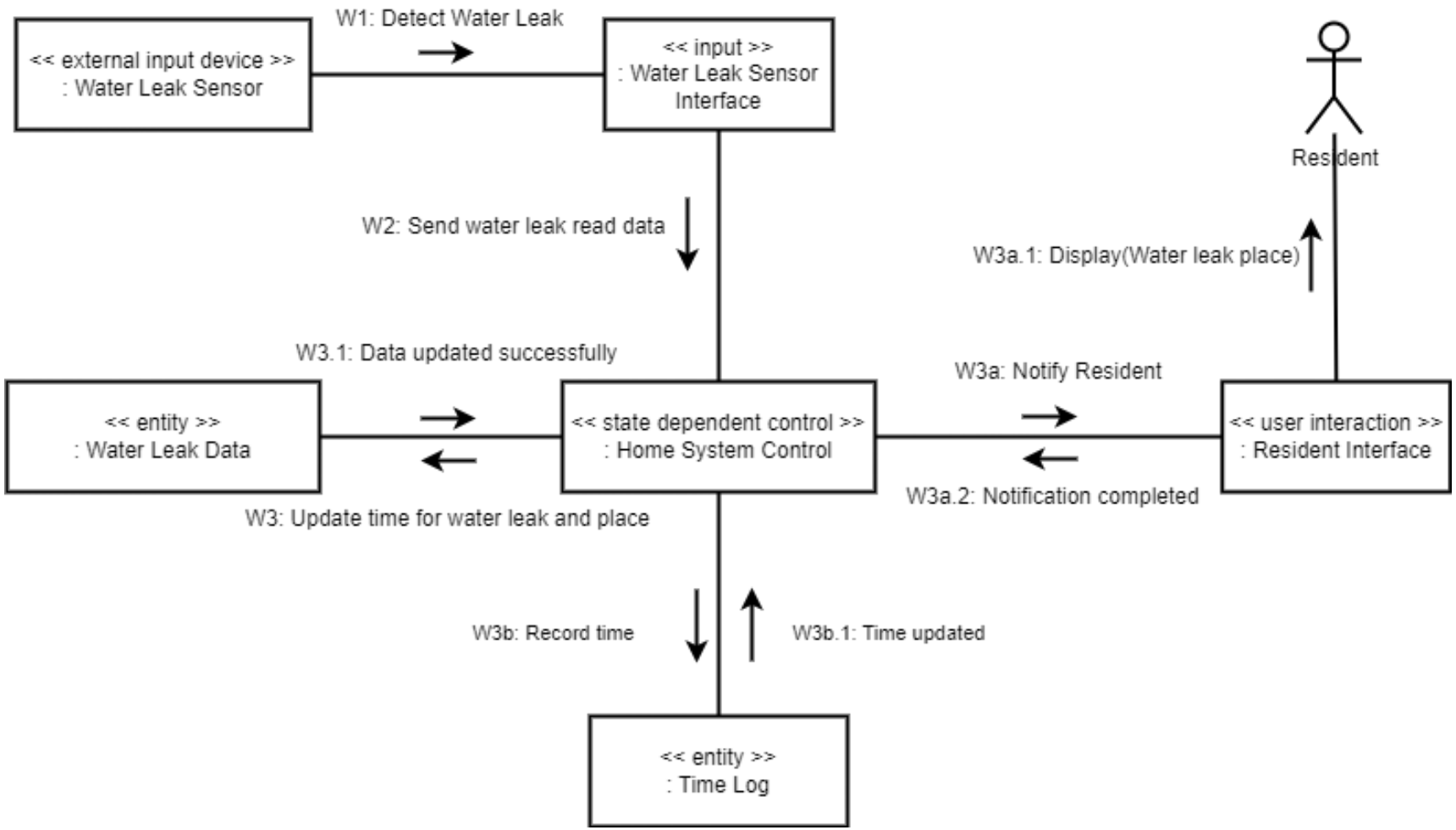
Assumptions:
In our system, Auto mileage data records both mileage and gas usage.

Use Case Name Light ON/OFF:

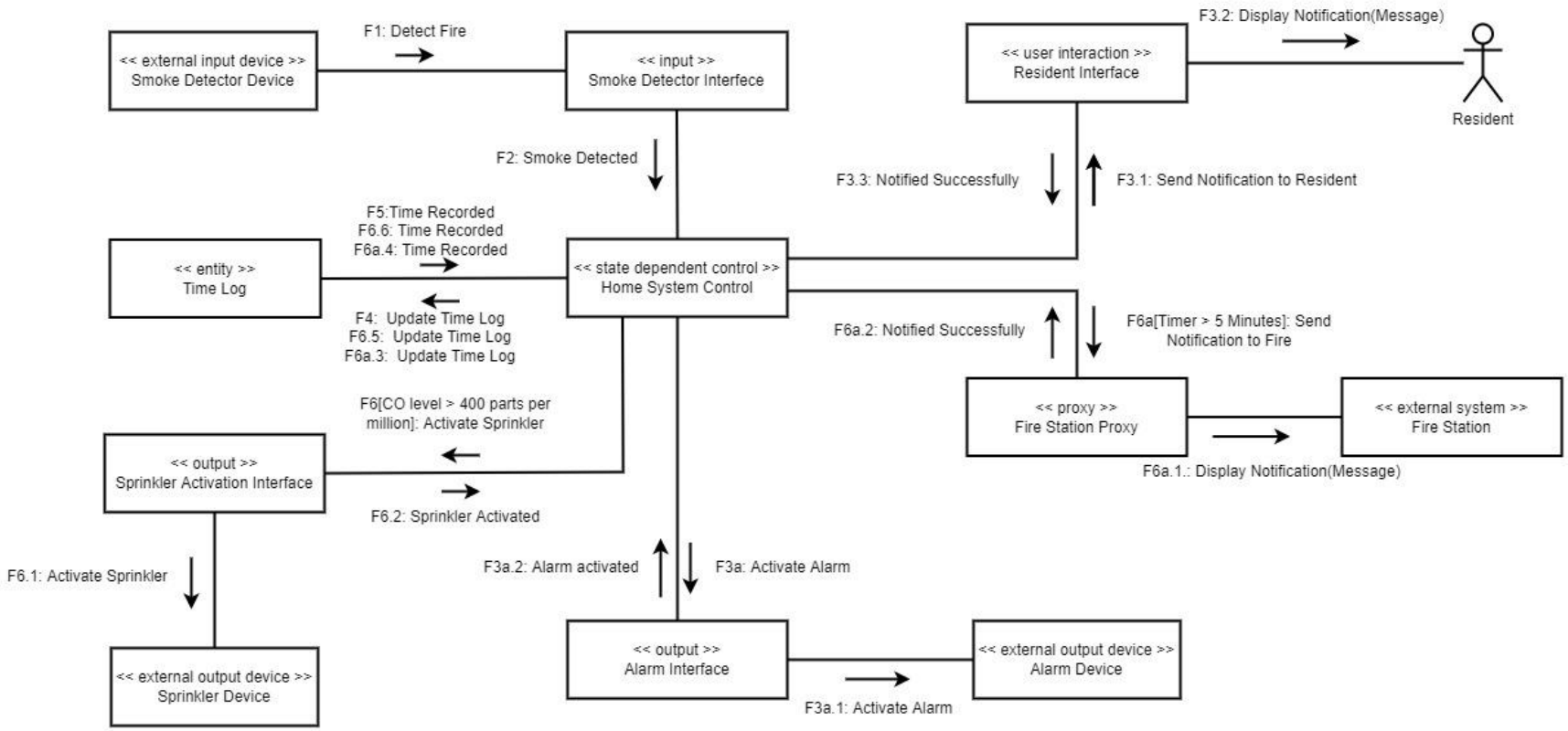


Assumptions:
Every single light fixture in the room follows the motion sensor signal which means all of them turn on and off at the same time.

Use Case Name Monitor Water Leak:



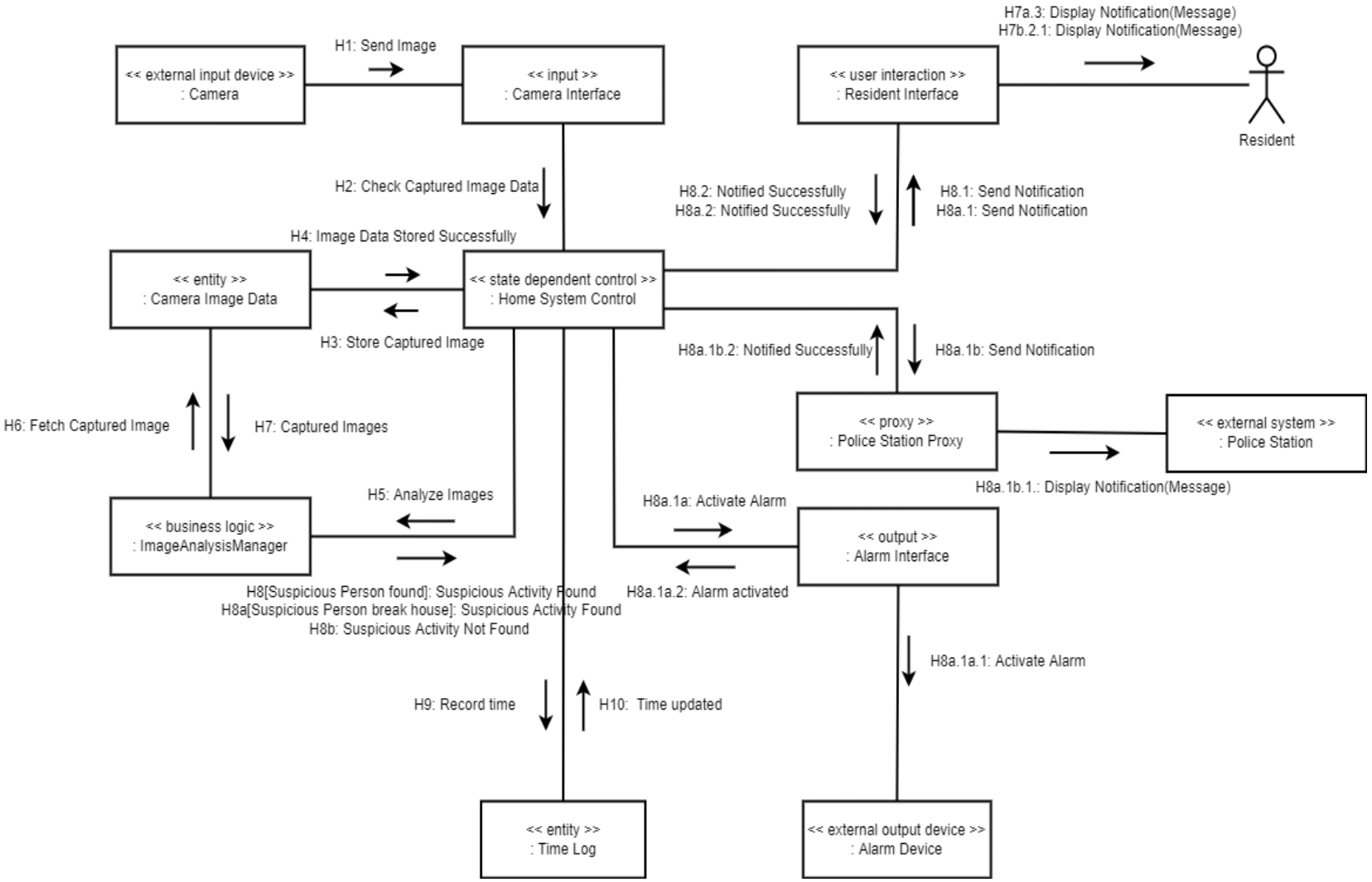
Use Case Name Detect Fire:



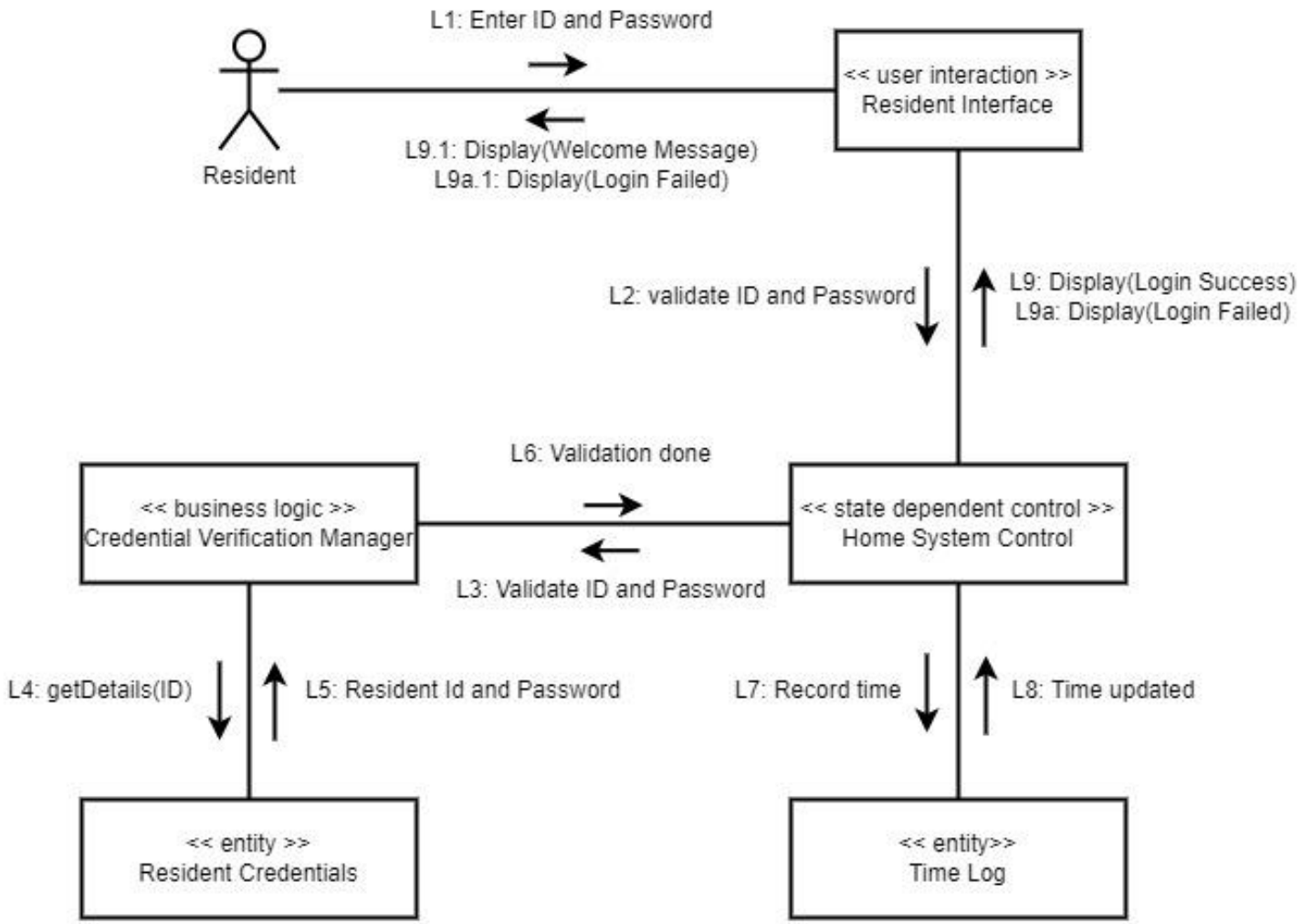
Assumptions:

Our system maintains several smoke detectors but each one is given a unique ID so that the notification will have information about which part of the home is in danger.

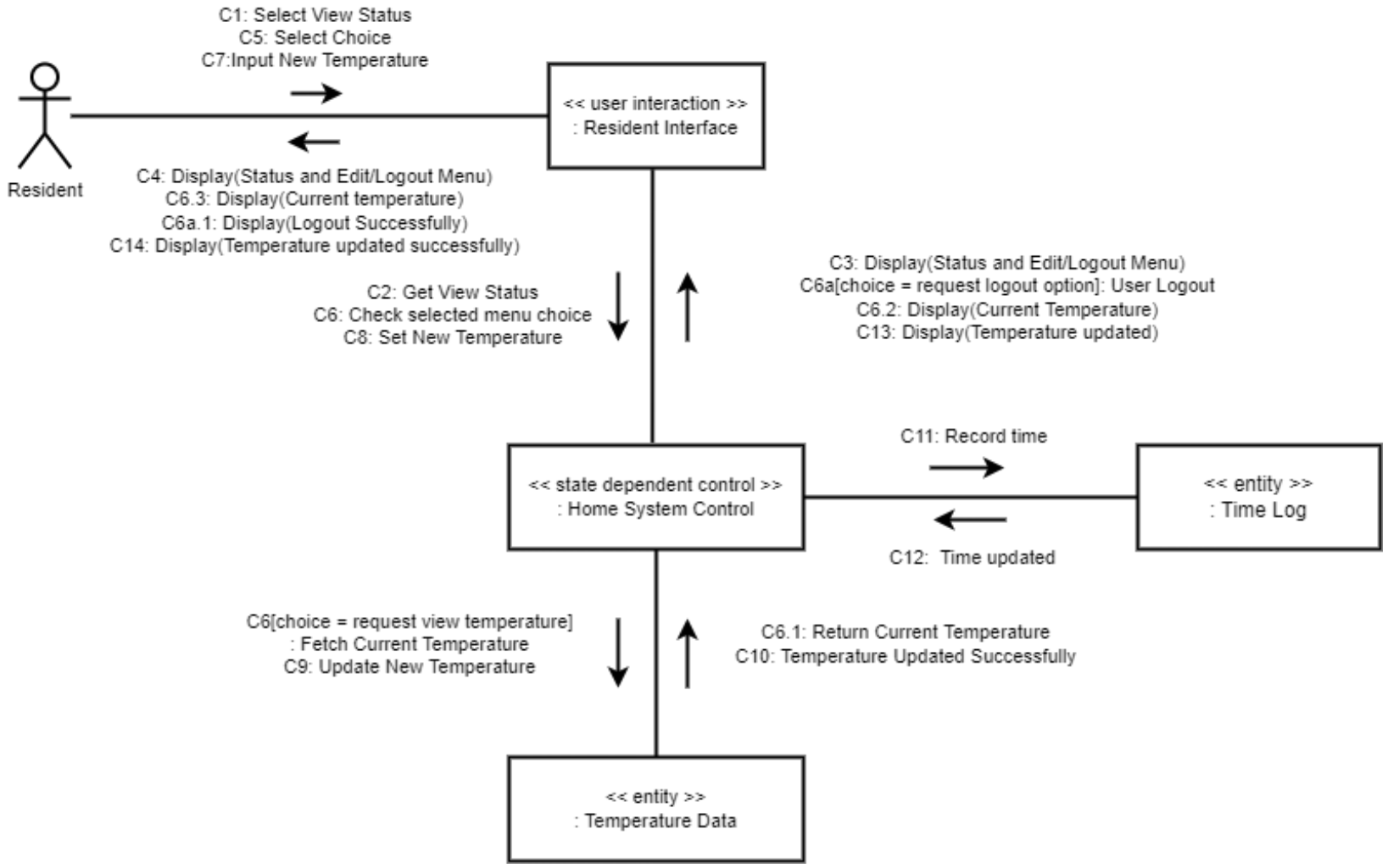
Use Case Name Home Security:



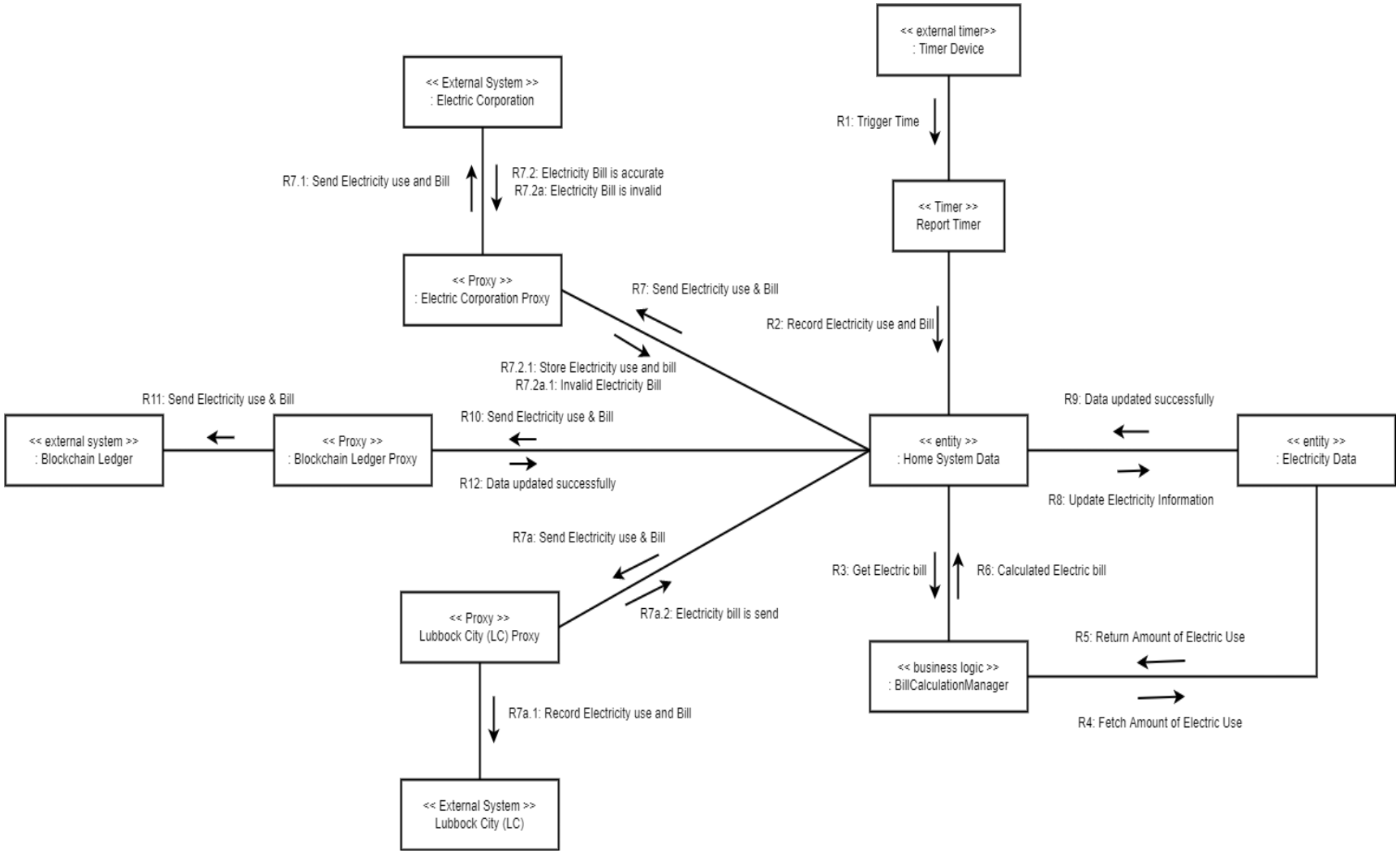
Use Case Name Login:



Use Case Name View/Change Values:



Use Case Name Record Electricity use and bill:

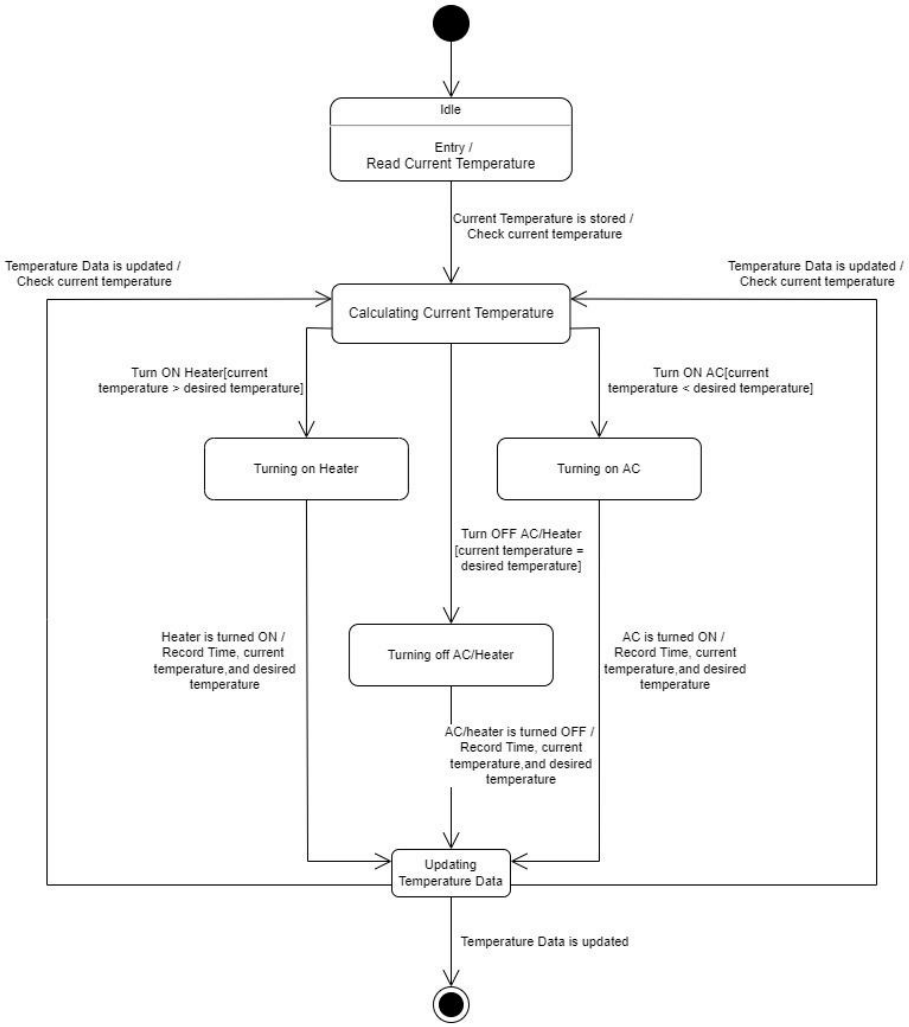


Assumptions:

Our system records electricity usage and bill every month and keeps up with any changes on the discount rates.
If the electricity usage and bill are invalid, it triggers an automatic recalculation of the data.

d) Develop the statechart for the control object. The statechart needs to show the alternative paths on the sequence diagrams (or communication diagrams). Ensure that the statechart is consistent with the sequence (or communication) diagram for use cases. (8 pts)

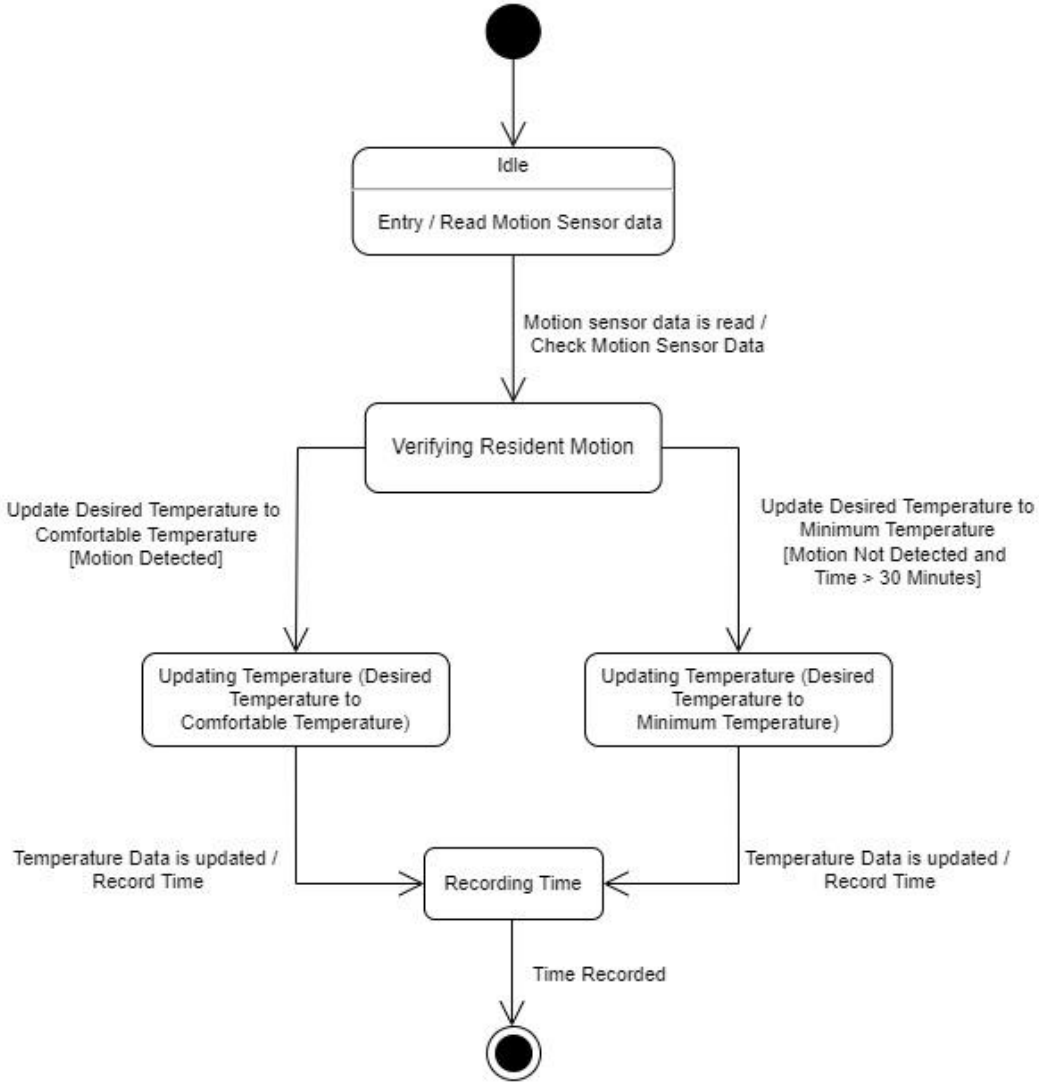
Use Case Name Control temperature:



Assumptions:

- In control temperature, controller will calculate temperature and take decision based on current temperature value that to turn on/off AC and heater.
- And at last, we are updating temperature data in Temperature log repository.

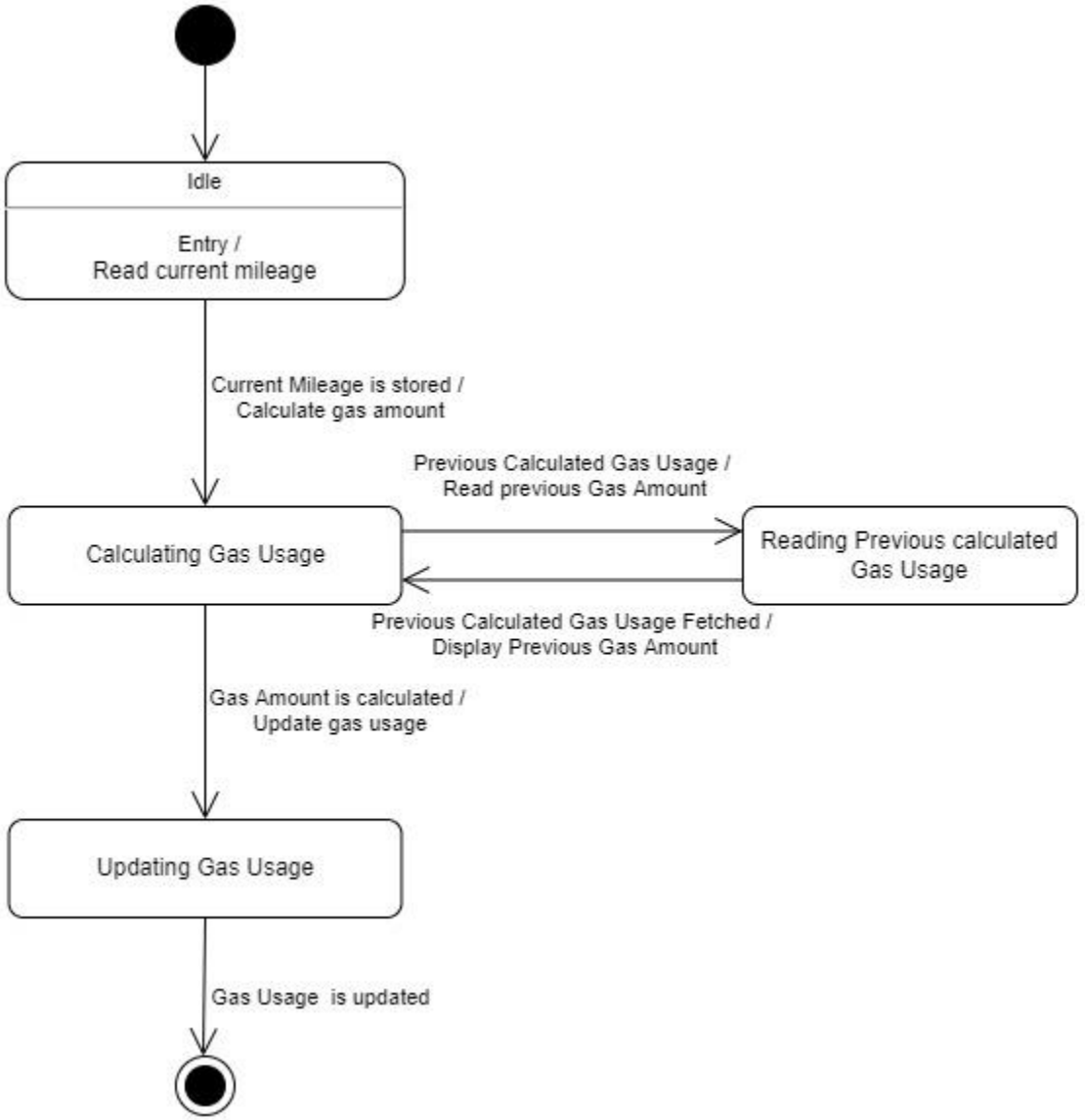
Use case name: Set Desired Temperature



Assumption –

- Motion sensor will detect the motion and if person is not available in the room, change temperature to minimum temperature. If person is in the room, update temperature to desired temperature and record the time after temperature update.

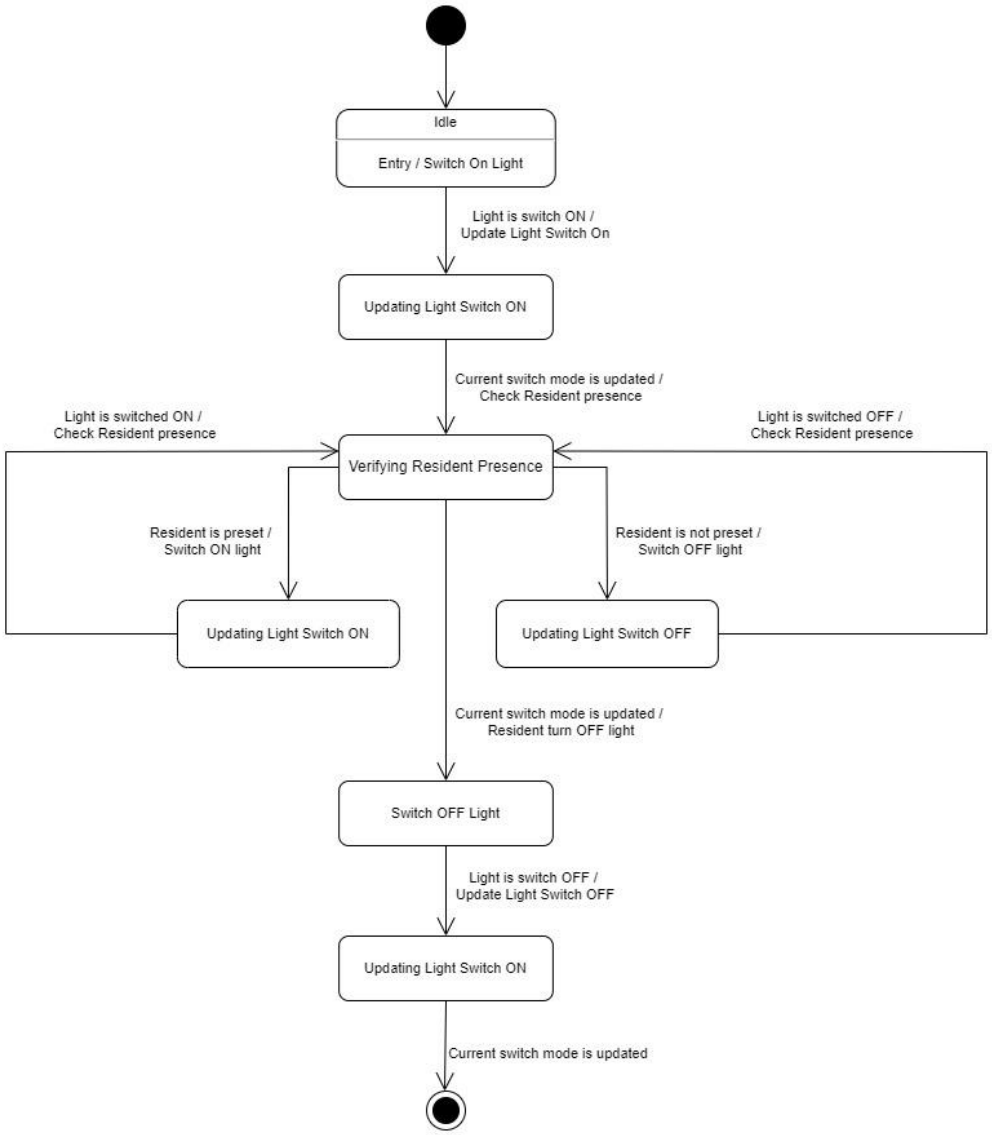
Use case name: Record Automobile Gas Use



Assumptions –

- Auto Mileage sensor device gives current mileage to controller and controller will calculate mileage using mileage database.

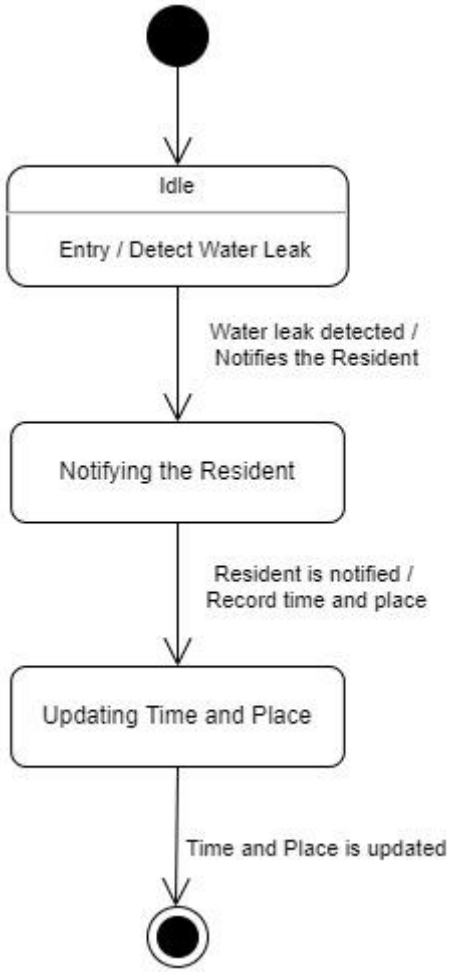
Use case name: Light On/Off



Assumptions –

- Resident turn on light switch. After that Light switch interface gives update to state dependent controller about Light switch ON. Our Motion sensor will continuously monitors the home and detects motion. If person is not present in room, then turn off light, else turn on light.

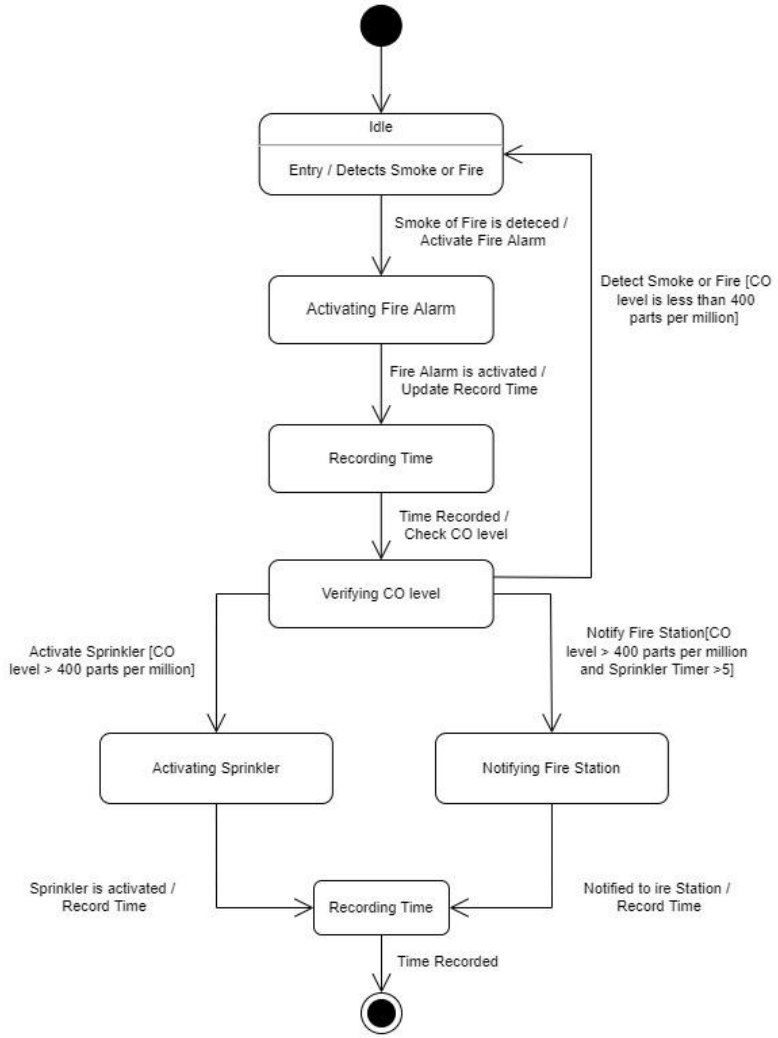
Use case name: Monitor Water Leak



Assumptions –

- Water Leak sensor detects water leak and share place and time to leak.
- And, Notify to resident about water leak. And update time and place it data repository.

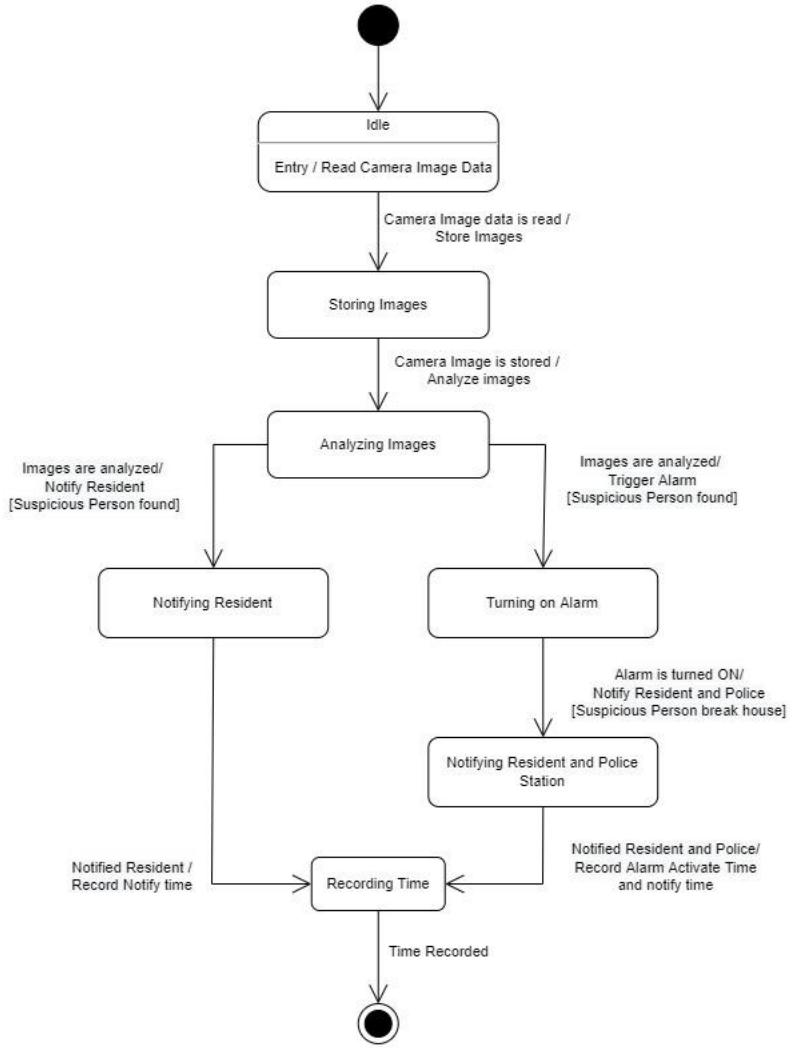
Use case name: Detect Fire



Assumptions –

- We are considering smoke detector sensor will detect smoke in idle state and continues further process such as Activate Alarm, record time. If CO level is more than 400, then gives alert to Fire station and turn on sprinkler. And, record time after turning on sprinkler.

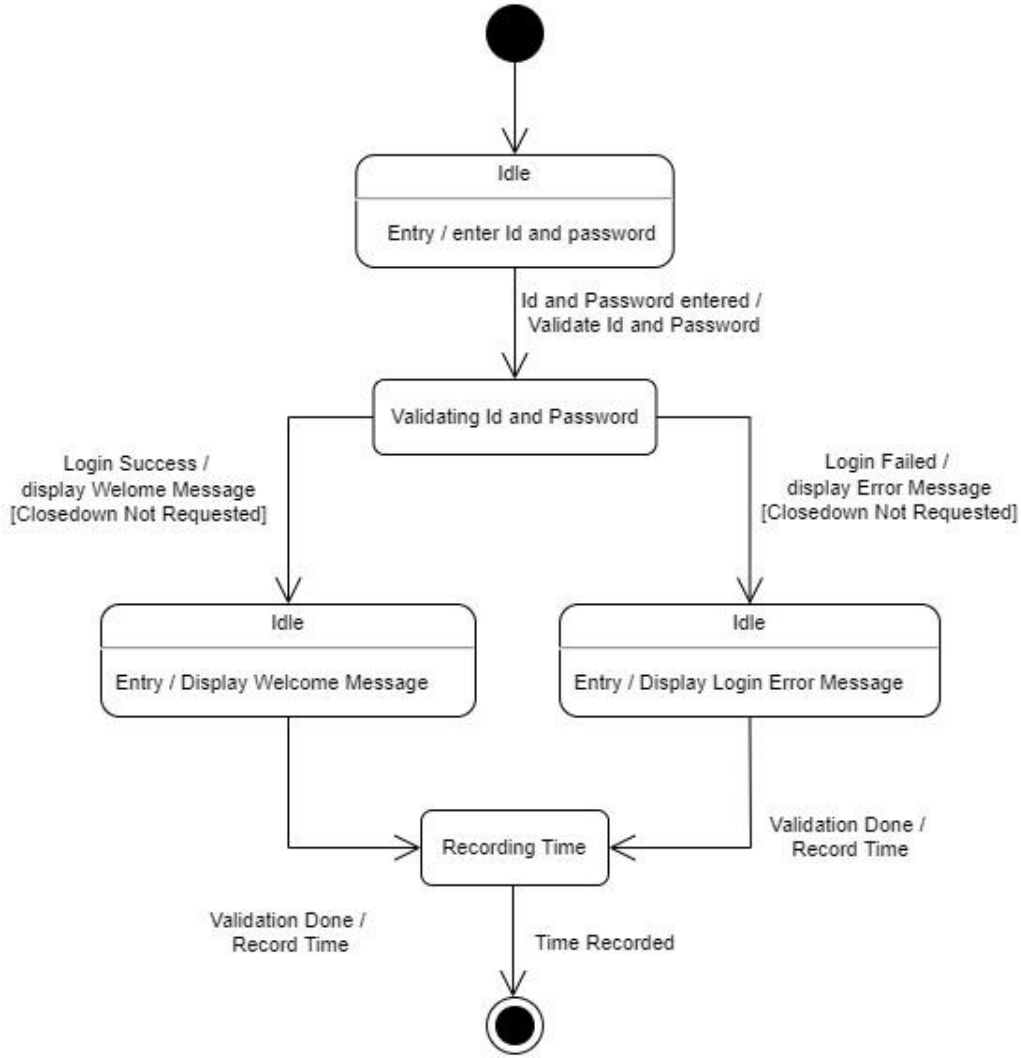
Use case name: Home Security



Assumptions -

- Camera will record home images and send to our Home System where it will store image and analyse image and takes an action such as if suspicious person is hanging around, it will notify to resident device and if suspicious person breaks the house, system will trigger alarm and notify to police station. And, record the time after triggering alarm and notifying.

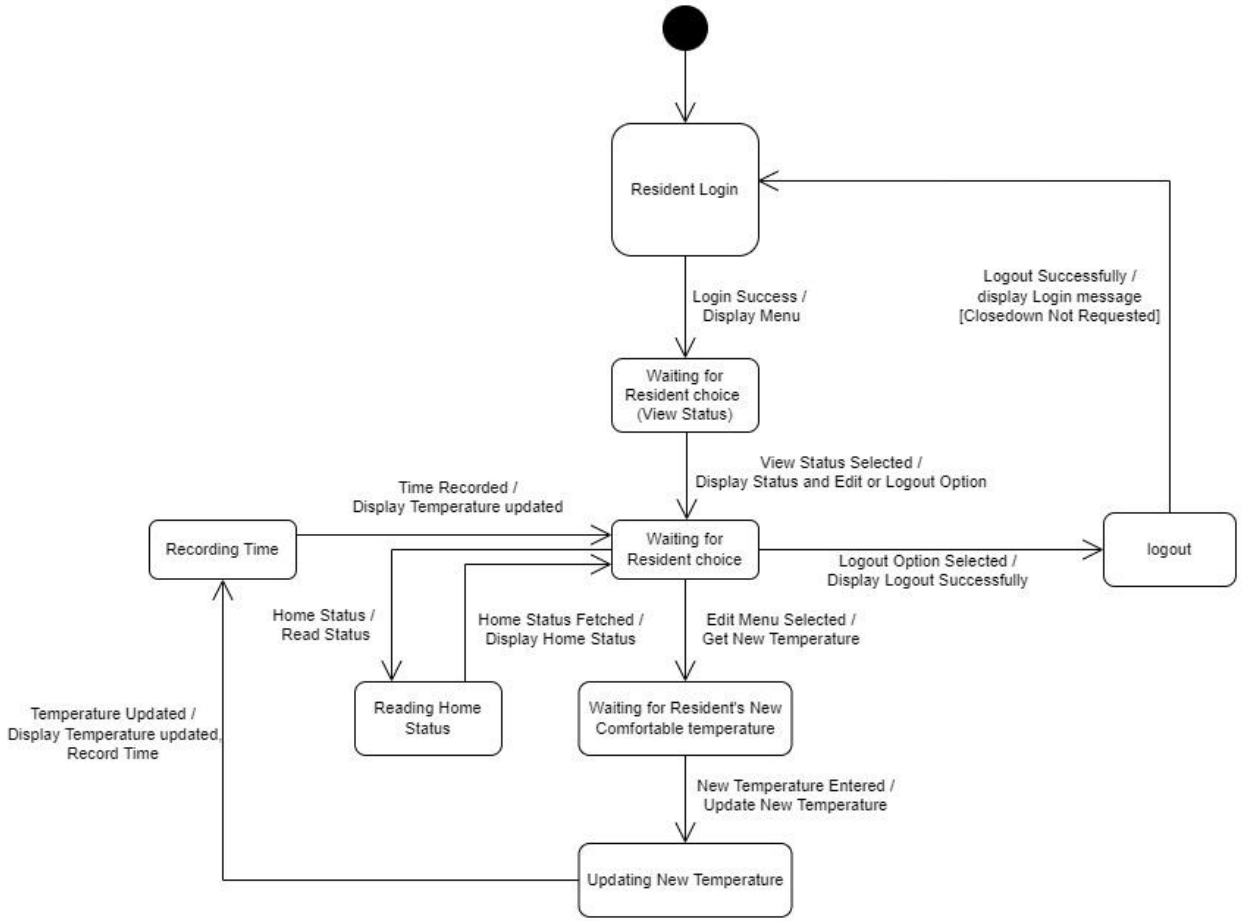
Use case name: Login



Assumptions –

- Resident enters username and password. Our Authentication business logic will verify authentication details. And display message accordingly. If authentication is ok, display welcome message, else, it would be error message. And, we are recording time after authentication.

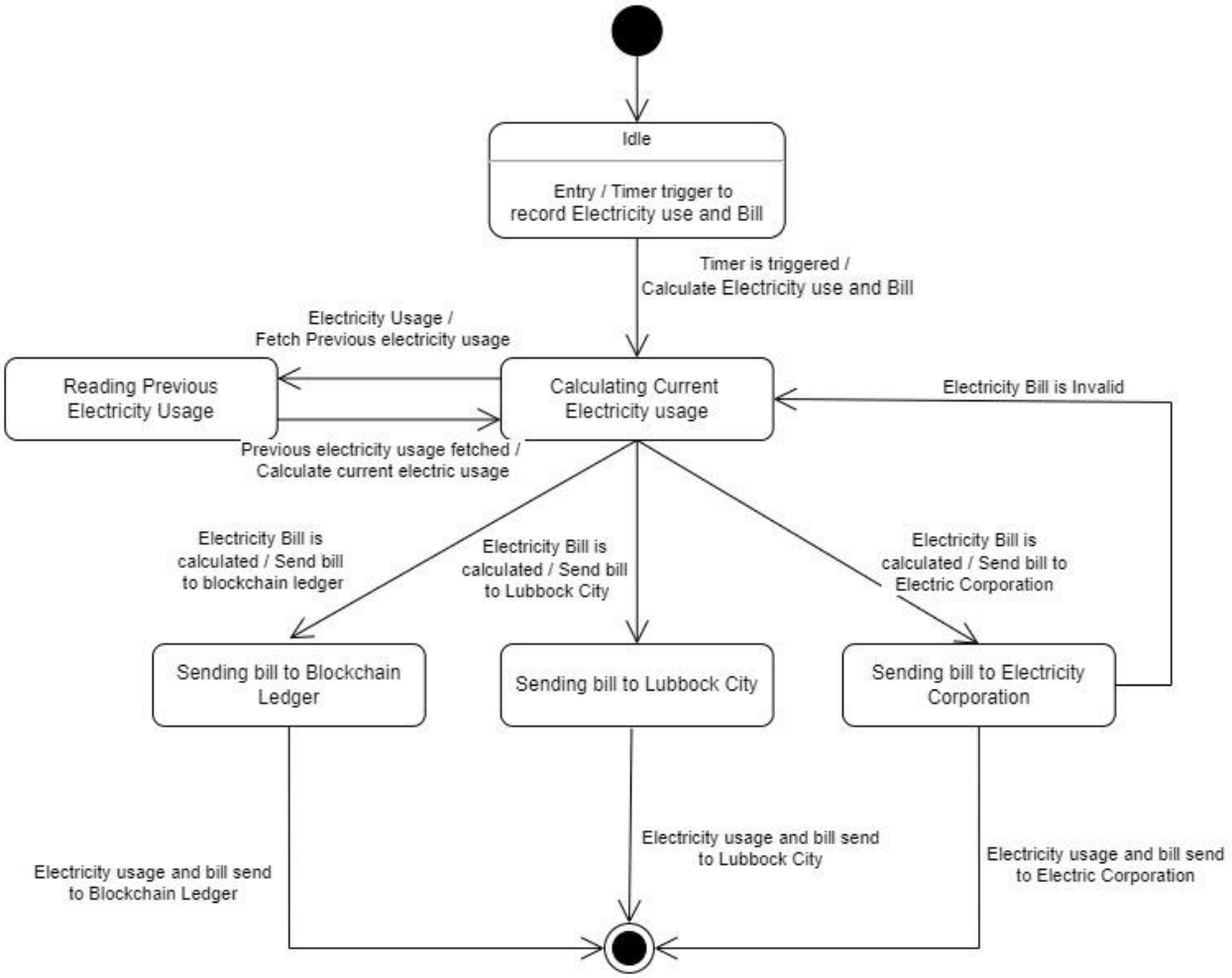
Use case name: View/Change Values



Assumptons –

- **We have merged state chart diagram for change values.** We merge Login state chart diagram because it is precondition for this change values use case. Once, we will get success login message from Login state chart, our change values state chart will begin. And, we have combined Logout state chart as it is post condition for this use case.
- We have considered states of each component that we had used in our communication diagram. We will take resident choice, if he/she selects Login, then we will route our path to Logout state diagram. Else, we will route to change temperature state.
- After this operation, we are calculating Time log for each action.

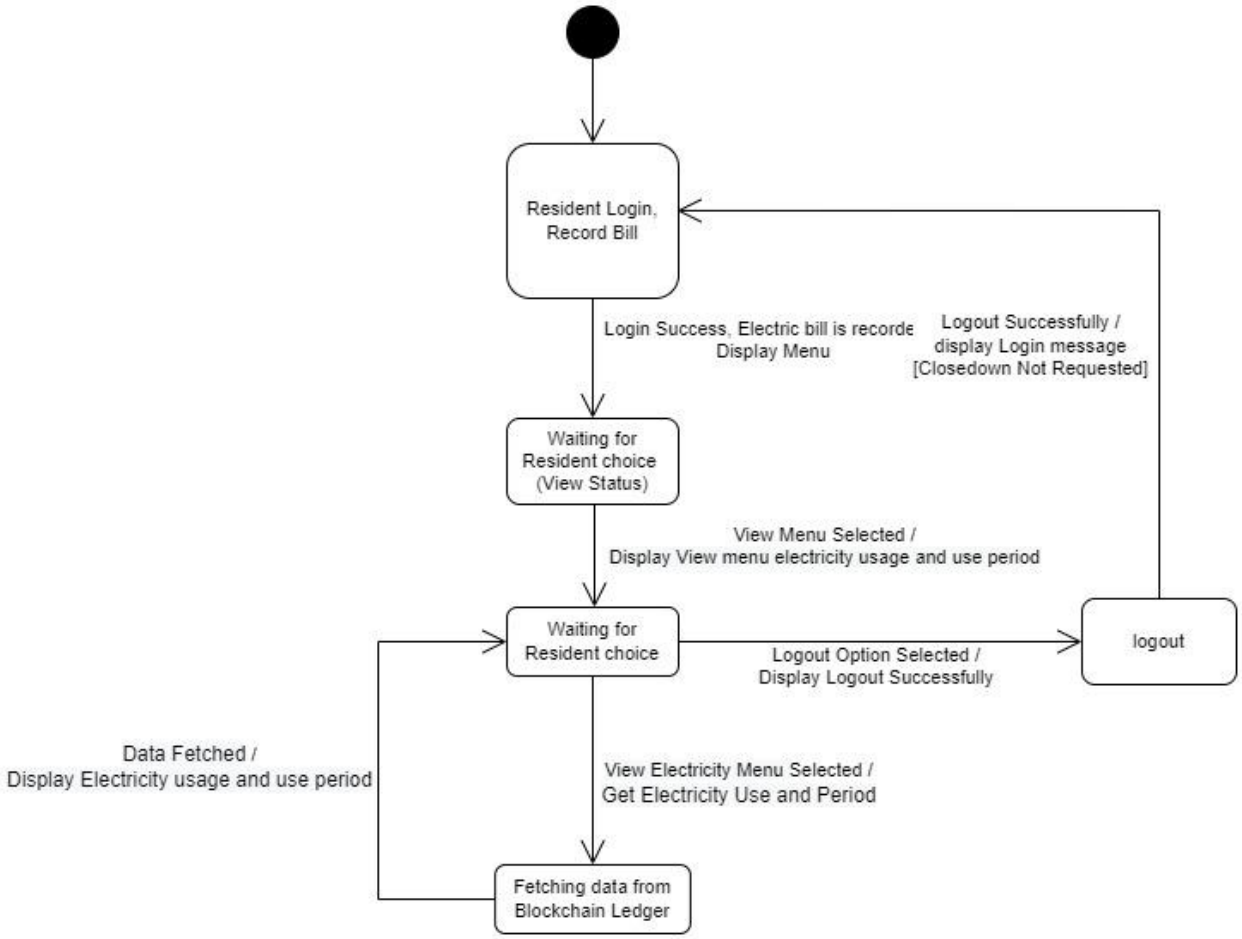
Use case name: Record Electricity Use and Bill



Assumptions –

- Timer will gives trigger about recording electricity use and bill. Our control will calculate bill and send electricity information to our proxy such as Electricity corporation, Lubbock city and Blockchain ledger.

Use case name: View Electricity Use and Bill



Assumptions –

- We have merged state chart for this View bill, **we combined Login, record bill and logout state** chart for this diagram. We have added module name for this user story.
- After success login and record bill execution, view bill use case will begin for execution.

Term Project: Phase II

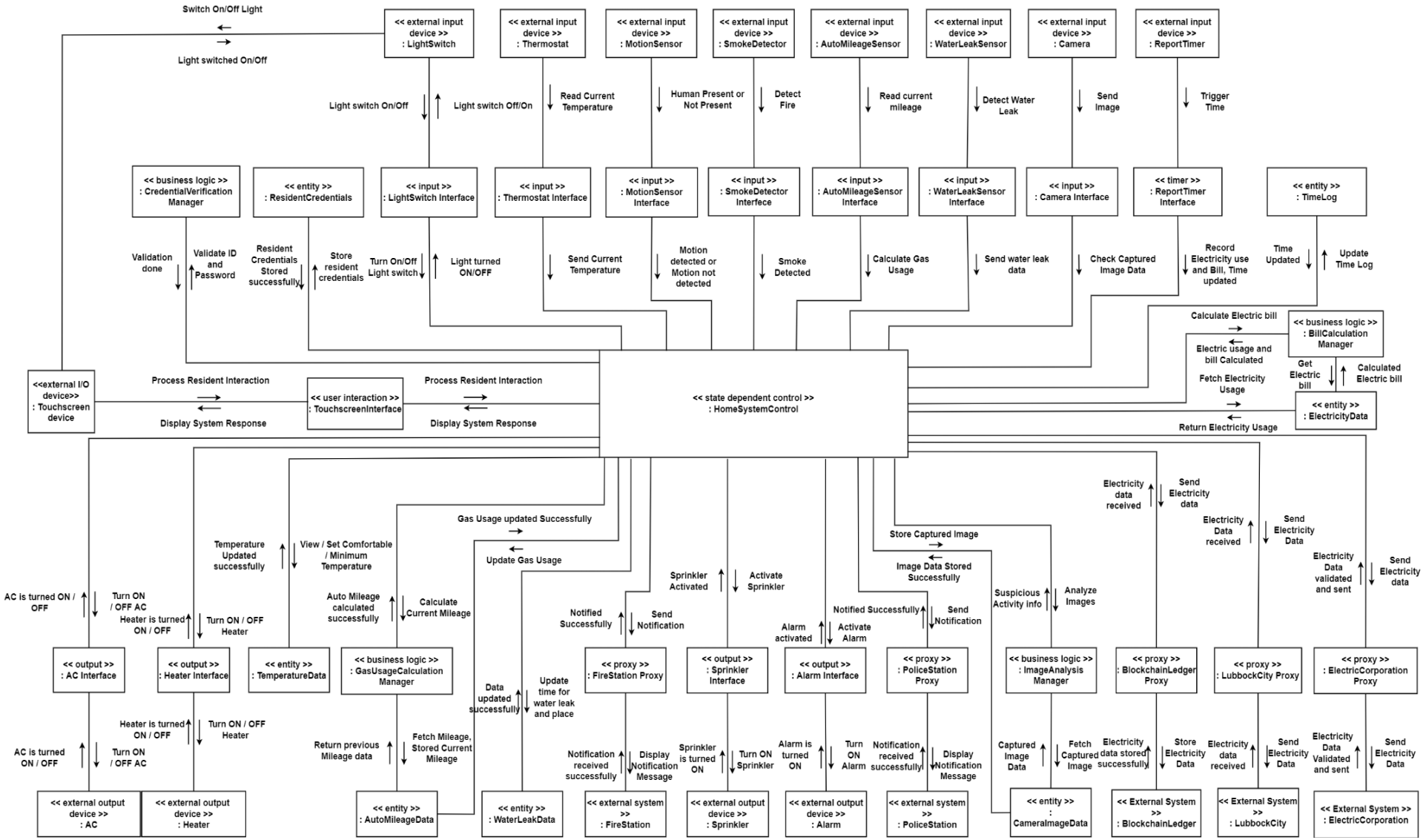
For Phase II of the project, you need to develop a Design Model for the SHS. In particular:

a) **Develop an integrated (consolidated) communication diagram(s) showing all the objects and messages (or abstract messages) in the system. (4 pts)**

Assumptions:

Aggregate message composed of simple messages	
Aggregate message	Consists of simple messages
Process Resident Interaction	Log in/off request, view/change values request, switch light on/off request
Suspicious activity information	Suspicious activity detected or not detected
Time updated	Time stamps for all necessary events along with device ID they are associated with

1. Resident interacts with the Smart Home System (SHS) though a touchscreen device which is interfaced directly to the SHS.
2. When every timer event is reported, the time stamps are stored in the timer log along with the device ID with which the timer event is associated. This helps the system determine specifically where the timer event is coming from.


Switch On/Off Light
Light switched On/Off
<< external input device >> : LightSwitch
<< external input device >> : Thermostat
<< external input device >> : MotionSensor
<< external input device >> : SmokeDetector
<< external input device >> : AutoMileageSensor
<< external input device >> : WaterLeakSensor
<< external input device >> : Camera
<< external input device >> : ReportTimer
Light switch On/Off
Light switch Off/On
Read Current Temperature
Human Present or Not Present
Detect Fire
Read current mileage
Detect Water Leak
Send Image
Trigger Time
<< business logic >> : CredentialVerification Manager
<< entity >> : ResidentCredentials
<< input >> : LightSwitch Interface
<< input >> : Thermostat Interface
<< input >> : MotionSensor Interface
<< input >> : SmokeDetector Interfece
<< input >> : AutoMileageSensor Interface
<< input >> : WaterLeakSensor Interface
<< input >> : Camera Interface
<< timer >> : ReportTimer Interface
<< entity >> : TimeLog
Validation done
Validate ID and Password
Resident Credentials Stored successfully
Store resident credentials
Turn On/Off Light switch
Light turned ON/OFF
Send Current Temperature
Motion detected or Motion not detected
Smoke Detected
Calculate Gas Usage
Send water leak data
Check Captured Image Data
Record Electricity use and Bill, Time updated
Time Updated
Update Time Log
Calculate Electric bill
Electric usage and bill Calculated
<< business logic >> : BillCalculation Manager
Get Electric bill
Calculated Electric bill
Fetch Electricity Usage
Return Electricity Usage
<< entity >> : ElectricityData
<<external I/O device>> : Touchscreen device
Process Resident Interaction
Display System Response
<< user interaction >> : TouchscreenInterface
Process Resident Interaction
Display System Response
<< state dependent control >> : HomeSystemControl
Electricity data received
Send Electricity data
Electricity Data received
Send Electricity Data
Electricity Data validated and sent
Send Electricity data
Store Captured Image
Image Data Stored Successfully
Suspicious Activity info
Analyze Images
Gas Usage updated Successfully
Update Gas Usage
Temperature Updated successfully
View / Set Comfortable / Minimum Temperature
Sprinkler Activated
Activate Sprinkler
Notified Successfully
Send Notification
Alarm activated
Activate Alarm
Notified Successfully
Send Notification
AC is turned ON / OFF
Turn ON / OFF AC
Heater is turned ON / OFF
Turn ON / OFF Heater
Auto Mileage calculated successfully
Calculate Current Mileage
<< output >> : AC Interface
<< output >> : Heater Interface
<< entity >> : TemperatureData
<< business logic >> : GasUsageCalculation Manager
<< proxy >> : FireStation Proxy
<< output >> : Sprinkler Interface
<< output >> : Alarm Interface
<< proxy >> : PoliceStation Proxy
<< business logic >> : ImageAnalysis Manager
<< proxy >> : BlockchainLedger Proxy
<< proxy >> : LubbockCity Proxy
<< proxy >> : ElectricCorporation Proxy
Heater is turned ON / OFF
Turn ON / OFF Heater
AC is turned ON / OFF
Turn ON / OFF AC
Return previous Mileage data
Fetch Mileage, Stored Current Mileage
Data updated successfully
Update time for water leak and place
Notification received successfully
Display Notification Message
Sprinkler is turned ON
Turn ON Sprinkler
Alarm is turned ON
Turn ON Alarm
Notification received successfully
Display Notification Message
Captured Image Data
Fetch Captured Image
Electricity data stored successfully
Store Electricity Data
Electricity data received
Send Electricity Data
Electricity Data Validated and sent
Send Electricity Data
<< external output device >> : AC
<< external output device >> : Heater
<< entity >> : AutoMileageData
<< entity >> : WaterLeakData
<< external system >> : FireStation
<< external output device >> : Sprinkler
<< external output device >> : Alarm
<< external system >> : PoliceStation
<< entity >> : CameraImageData
<< External System >> : BlockchainLedger
<< External System >> : LubbockCity
<< External System >> : ElectricCorporation

b) Define the subsystem architecture (depicted on a concurrent communication diagram), showing the subsystems and their interactions. Describe the criteria used for subsystem structuring. Define the message communication interfaces between the subsystems. (6 pts)

Assumptions:

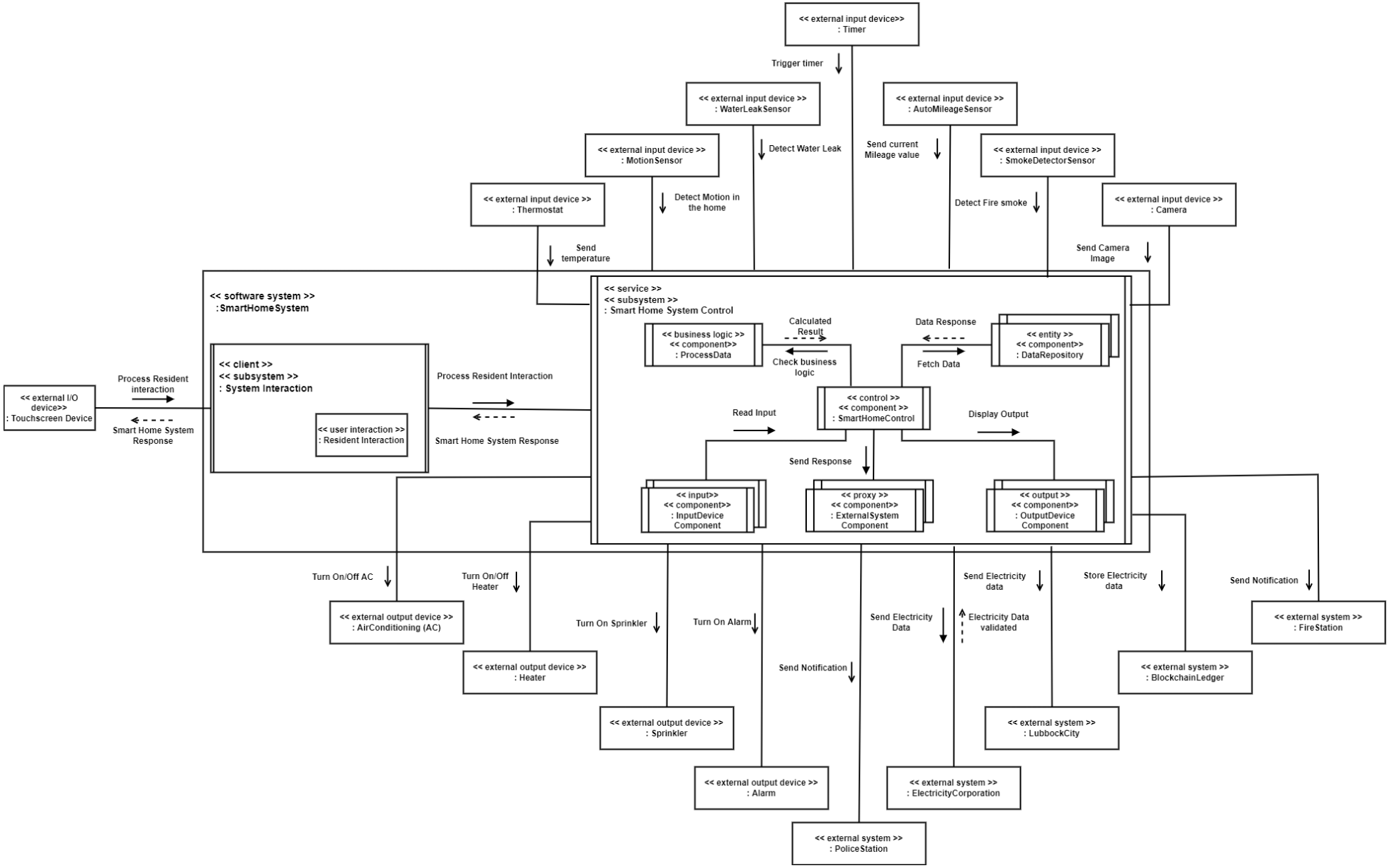
A system called the Smart Home System (SHS) is structured into two subsystems. The first is the client subsystem which encompasses the resident interaction component. This is done because resident interaction is a requester of services. Anything from setting desired temperatures to accessing the electricity usage and bill information is achieved by requesting a service from SHS.

The second is a service subsystem which encompasses the data repository, I/O device interfaces, proxies, business logic component and a SHS control component. This control component is responsible for controlling all the components. This is how it provides service to the client subsystem. Anything the client wants to get done through SHS is accomplished through this service subsystem.

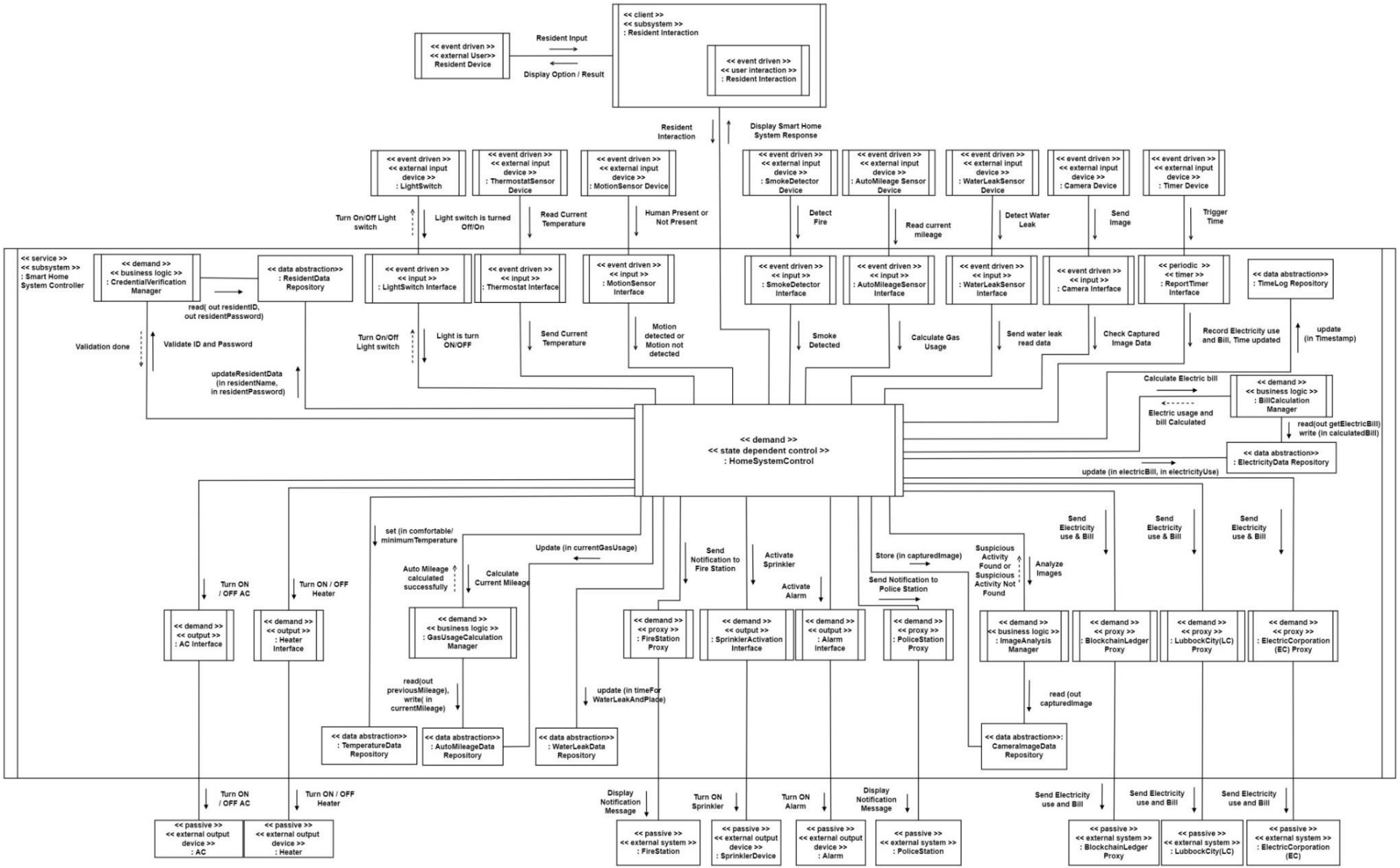
All the input/output devices are structured outside the SHS which keeps the system autonomous while also preserving the possibility of future expansion of SHS.

Message communication in subsystem structuring follows both asynchronous and synchronous conventions. The **Process Resident Interaction** message is a synchronous message with reply as it needs to wait for the message to be accepted and then to be replied before proceeding. The **Check Business Logic**, **Fetch Data** and **Send Electricity Data** to Electricity Corporation are all synchronous messages with reply as without the response the producer cannot proceed.

Every other message communication coming from I/O devices is asynchronous. They just pass the information and go about their business.


<< external input device>>
: Timer
Trigger timer
<< external input device >>
: WaterLeakSensor
Detect Water Leak
<< external input device >>
: AutoMileageSensor
Send current
Mileage value
<< external input device >>
: MotionSensor
Detect Motion in
the home
<< external input device >>
: SmokeDetectorSensor
Detect Fire smoke
<< external input device >>
: Thermostat
Send
temperature
<< external input device >>
: Camera
Send Camera
Image
<< software system >>
:SmartHomeSystem
<< service >>
<< subsystem >>
: Smart Home System Control
<< business logic >>
<< component>>
: ProcessData
Calculated
Result
Check business
logic
Data Response
Fetch Data
<< entity >>
<< component>>
: DataRepository
<< client >>
<< subsystem >>
: System Interaction
<< user interaction >>
: Resident Interaction
Process Resident
interaction
Smart Home System
Response
Process Resident Interaction
Smart Home System Response
<< external I/O
device>>
: Touchscreen Device
<< control >>
<< component >>
: SmartHomeControl
Read Input
Display Output
Send Response
<< input>>
<< component>>
: InputDevice
Component
<< proxy >>
<< component>>
: ExternalSystem
Component
<< output >>
<< component>>
: OutputDevice
Component
Turn On/Off AC
Turn On/Off
Heater
Send Electricity
data
Store Electricity
data
Send Notification
<< external output device >>
: AirConditioning (AC)
Turn On Sprinkler
Turn On Alarm
Send Electricity
Data
Electricity Data
validated
<< external system >>
: FireStation
<< external output device >>
: Heater
Send Notification
<< external system >>
: BlockchainLedger
<< external output device >>
: Sprinkler
<< external system >>
: LubbockCity
<< external output device >>
: Alarm
<< external system >>
: ElectricityCorporation
<< external system >>
: PoliceStation

c) Define the task architecture (depicted on concurrent communication diagrams), showing each subsystem's concurrent tasks and interfaces. Describe the criteria used for task structuring. Define the message communication interfaces between tasks (6 pts)



Task Interaction Specification -

Subsystem: Resident Interaction

Name: Resident Interaction
Information hidden: Details of User Interaction with Home Control System.
Structuring criteria: role criterion: Resident Interaction; concurrency criterion: event driven.
Assumptions: Resident gives request (read / write) and expects output results from system.
Anticipated Changes: Additional information should be display for more user choice.
Task interface:

Task inputs:
Event input: Resident display external user gives input to view/change details.
External input: HomeSystemControl
Synchronous message communication with reply:

- Resident Input
- Display Option/Result

Task outputs: Resident user is able to view and change values.
External output: ResidentDevice
Asynchronous message communication: None
Passive objects accessed: None
Errors detected: Invalid Login Details, Invalid Choice, and Invalid value.

Assumption –

- We consider external device as Resident device as generic name. It represents Resident Mobile, Tab, Laptop or any other hardware device for information access.
- We have synchronous message for external device with name Resident Input as Request message and Display Option as Response message which includes –
    - Resident Input - Enter ID and Password, Select View Status, Select Choice, New Temperature Input, Select View Status, Select View Electricity Usage
    - Display Output - Display(Welcome Message), Display(Login Failed), Display Notification(Message), Display Notification(Message), Display Notification(Message), Display(Status and Edit/Logout Menu), Display(Temperature updated), Display(Logout Successfully), Display(View menu electricity usage and use period), Display(Electricity usage and use period)

SubSystem: Smart Home SystemControl

Name: LightSwitch interface
Information hidden: Turn On/Off Light Switch.

Structuring criteria: role criterion: Input/Output; concurrency criterion: event driven
Assumptions: Turn On/Off Light.
Anticipated Changes:
Task interface:

Task inputs:
Event input: Turn On/Off Light.
External input: turnOn/OffLight
Synchronous message communication with reply:

- TurnOn/OffLight
- LightisON/OFF

Task outputs: Gives data to Controller for further operation
External output: turnOn/OffLight
Asynchronous message communication: None
Passive objects accessed: None
Errors detected: Light is not turned On after ON action and Light is not turned Off after OFF action.

Name: Thermostat interface
Information hidden: Read Current Room Temperature.
Structuring criteria: role criterion: Input; concurrency criterion: event driven
Assumptions: Detect the current room temperature.
Anticipated Changes:
Task interface:

Task inputs:
Event input: Read current temperature.
External input: readCurrentTemperature
Synchronous message communication without reply: None
Task outputs: Gives data to Controller for further operation
External output: sendCurrentTemperature
Asynchronous message communication:

- readCurrentTemperature

Passive objects accessed: None
Errors detected: Wrong temperature detected.

Name: Motion Sensor interface
Information hidden: Detect Motion/Person is the room.
Structuring criteria: role criterion: Input; concurrency criterion: event driven

Assumptions: Detect the person in the room.
Anticipated Changes:
Task interface:

Task inputs:
Event input: Detect Motion.
External input: detectMotion
Synchronous message communication without reply: None
Task outputs: Gives data to Controller for further operation
External output: motionDetectedOrNotDetected
Asynchronous message communication:

- detectMotion

Passive objects accessed: None
Errors detected: Motion not detected even if person is present in room.

Name: Smoke Detector interface
Information hidden: Detect Smoke in the room.
Structuring criteria: role criterion: Input; concurrency criterion: event driven
Assumptions: Detect smoke.
Anticipated Changes:
Task interface:

Task inputs:
Event input: Detect Smoke.
External input: detectFireSmoke
Synchronous message communication without reply: None
Task outputs: Gives data to Controller for further operation
External output: smokeDetect
Asynchronous message communication:

- detectFireSmoke

Passive objects accessed: None
Errors detected: Fire Smoke is not detected even if smoke is present in room.

Name: Auto Mileage Sensor interface
Information hidden: Read Current Mileage Information
Structuring criteria: role criterion: Input; concurrency criterion: event driven
Assumptions: Read Current Mileage.
Anticipated Changes:

Task interface:

Task inputs:

Event input: Read Mileage.

External input: readCurrentMileage

Synchronous message communication without reply: None

Task outputs: Gives data to Controller for further operation

External output: getGasUsage

Asynchronous message communication:

- readCurrentMileage

Passive objects accessed: None

Errors detected: Not able to read current mileage

Name: Water leak sensor interface

Information hidden: Takes action based on the water leak sensor.

Structuring criteria: role criterion: Input; concurrency criterion: event driven

Assumptions: Gives water leak place in home and time of water leak.

Anticipated Changes: Need to give information about last installation/repair date.

Task interface:

Task inputs:

Event input: Read water leak place and time.

External input: detectWaterLeak

Synchronous message communication without reply: None

Task outputs: Gives data to Controller for further operation

External output: sendWaterLeakReadData

Asynchronous message communication:

- detectWaterLeak

Passive objects accessed: None

Errors detected: Invalid detect.

Name: Camera Interface

Information hidden: Provide Camera Images

Structuring criteria: role criterion: Input; concurrency criterion: event driven

Assumptions: Read Live Images.

Anticipated Changes:

Task interface:

Task inputs:

Event input: Read Camera Images.
External input: sendImages
Synchronous message communication without reply: None
Task outputs: Gives data to Controller for further operation
External output: checkCapturedImageData
Asynchronous message communication:

- sendImages

Passive objects accessed: None
Errors detected: Images is not in readable format

Name: Report Timer Interface
Information hidden: Trigger Timer after every predefined time interval for calculate electricity bill information
Structuring criteria: role criterion: Input; concurrency criterion: event driven
Assumptions: Trigger timer.
Anticipated Changes:
Task interface:

Task inputs:
Event input: Trigger time after each interval.
External input: triggerTime
Synchronous message communication without reply: None
Task outputs: Gives data to Controller for further operation
External output: recordElectricityBill
Asynchronous message communication:

- triggerTime

Passive objects accessed: None
Errors detected: Timer is not triggered on specified time.

Name: AC interface
Information hidden: AC turn On/Off details.
Structuring criteria: role criterion: output; concurrency criterion: demand
Assumptions: AC will turn On/Off.
Anticipated Changes:
Task interface:

Task inputs:
Event input: Turn on AC if current temperature is greater than desired temperature
External input: turnOn/OffAC

Synchronous message communication without reply:

- turnOn/OffAC

Task outputs:
External output: ACturnedOn/OFF
Asynchronous message communication: None
Passive objects accessed: None
Errors detected: Failed to turn on/off the AC.

Name: Heater interface
Information hidden: Heater turn On/Off details.
Structuring criteria: role criterion: output; concurrency criterion: demand
Assumptions: Heaterwill turn On/Off.
Anticipated Changes:
Task interface:

Task inputs:
Event input: Turn on Heater if current temperature is greater than desired temperature
External input: turnOn/OffHeater
Synchronous message communication without reply:

- turnOn/OffHeater

Task outputs:
External output: HeaterturnedOn/OFF
Asynchronous message communication: None
Passive objects accessed: None
Errors detected: Failed to turn on/off the Heater.

Name: Alarm interface
Information hidden: Alarm details for trigger.
Structuring criteria: role criterion: output; concurrency criterion: demand
Assumptions: The Alarm will activate.
Anticipated Changes:
Task interface:

Task inputs:
Event input: Activate alarm on Fire detection and on Theft detection from Camera Detection.
External input: activateAlarm
Synchronous message communication without reply:

- activateAlarm

Task outputs:
External output: alarmActivated
Asynchronous message communication: None
Passive objects accessed: None
Errors detected: Failed to activate an alarm.

Name: Sprinkler Activation interface
Information hidden: Sprinkler activation details.
Structuring criteria: role criterion: output; concurrency criterion: demand
Assumptions: Sprinkle will trigger.
Anticipated Changes:
Task interface:
Task inputs:
Event input: Sprinkle activation details.
External input: activateSprinkler
Synchronous message communication without reply:
- activateSprinkler

Task outputs:
External output: ActivateSprinkler
Asynchronous message communication: None
Passive objects accessed: None
Errors detected: Failed to turn on the sprinkler.

Name: Fire Station Proxy
Information hidden: Notification details for Fire Station.
Structuring criteria: role criterion: proxy; concurrency criterion: event driven
Assumptions: System will notify to Fire Station.
Anticipated Changes:
Task interface:
Task inputs:
Event input: Notification sent to fire stations and alert received message from the fire station.
External input: sentNotificationToFireStation
Synchronous message communication without reply:
- sentNotificationToFireStation

Task outputs: sent notification to fire station after smoke detected
External output: notificationSent
Asynchronous message communication: None

Passive objects accessed: None

Errors detected: Wrong notification sent to fire station.

Name: Police Station Proxy
Information hidden: Notification details for Police Station.
Structuring criteria: role criterion: proxy; concurrency criterion: event driven
Assumptions: System will notify to Police Station.
Anticipated Changes:
Task interface:

Task inputs:

Event input: Notification sent to police stations and alert received message from the fire station.

External input: sentNotificationToPoliceStation

Synchronous message communication without reply:

- sentNotificationToPoliceStation

Task outputs: sent notification to fire station after theft detected

External output: notificationSent

Asynchronous message communication: None

Passive objects accessed: None

Errors detected: Wrong notification sent to police station.

Name: Lubbock City Proxy
Information hidden: Send electricity bill information
Structuring criteria: role criterion: proxy; concurrency criterion: event driven
Assumptions: Sent calculated electric bill to Lubbock City
Anticipated Changes: None.
Task interface:

Task inputs:

Event input: Send calculated Electricity bill detail.

External input: None

Synchronous message communication without reply:

- sendElectricityBill

Task outputs:

External output: electricityBillInformation

Asynchronous message communication: None

Passive objects accessed: None

Errors detected: Invalid Electricity information to Lubbock City.

Name: Electric Corporation Proxy
Information hidden: Send electricity bill information
Structuring criteria: role criterion: proxy; concurrency criterion: event driven
Assumptions: Sent calculated electric bill to Electric Corporation
Anticipated Changes: None.
Task interface:

Task inputs:
Event input: Send calculated Electricity bill detail.
External input: None
Synchronous message communication without reply:

- sendElectricityBill

Task outputs:
External output: electricityBillInformation
Asynchronous message communication: None
Passive objects accessed: None
Errors detected: Invalid Electricity information to Electric Corporation.

Name: Blockchain Ledger Proxy
Information hidden: Send electricity bill information
Structuring criteria: role criterion: proxy; concurrency criterion: event driven
Assumptions: Sent calculated electric bill to Blockchain Ledger
Anticipated Changes: None.
Task interface:

Task inputs:
Event input: Send calculated Electricity bill detail.
External input: None
Synchronous message communication without reply:

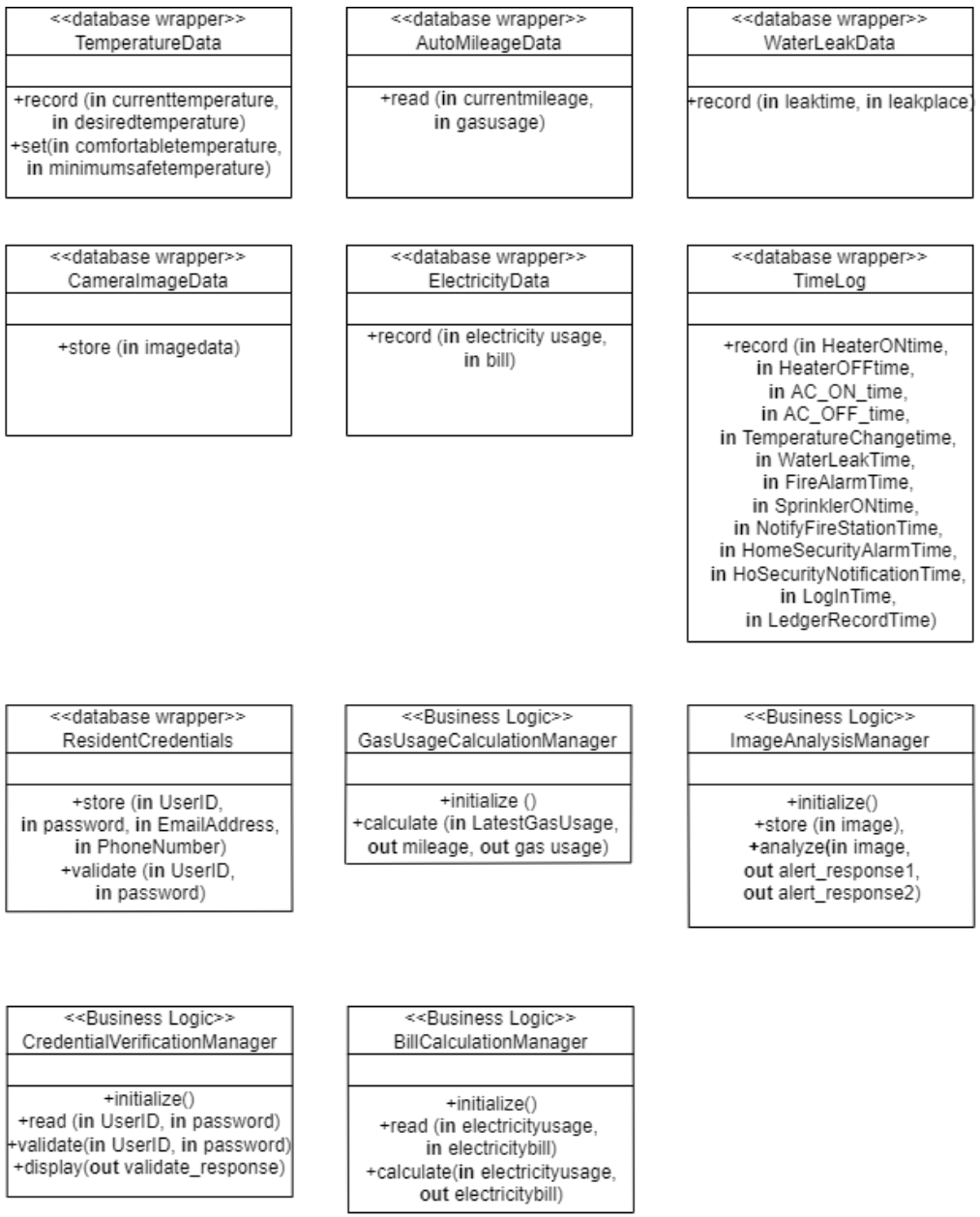
- sendElectricityBill

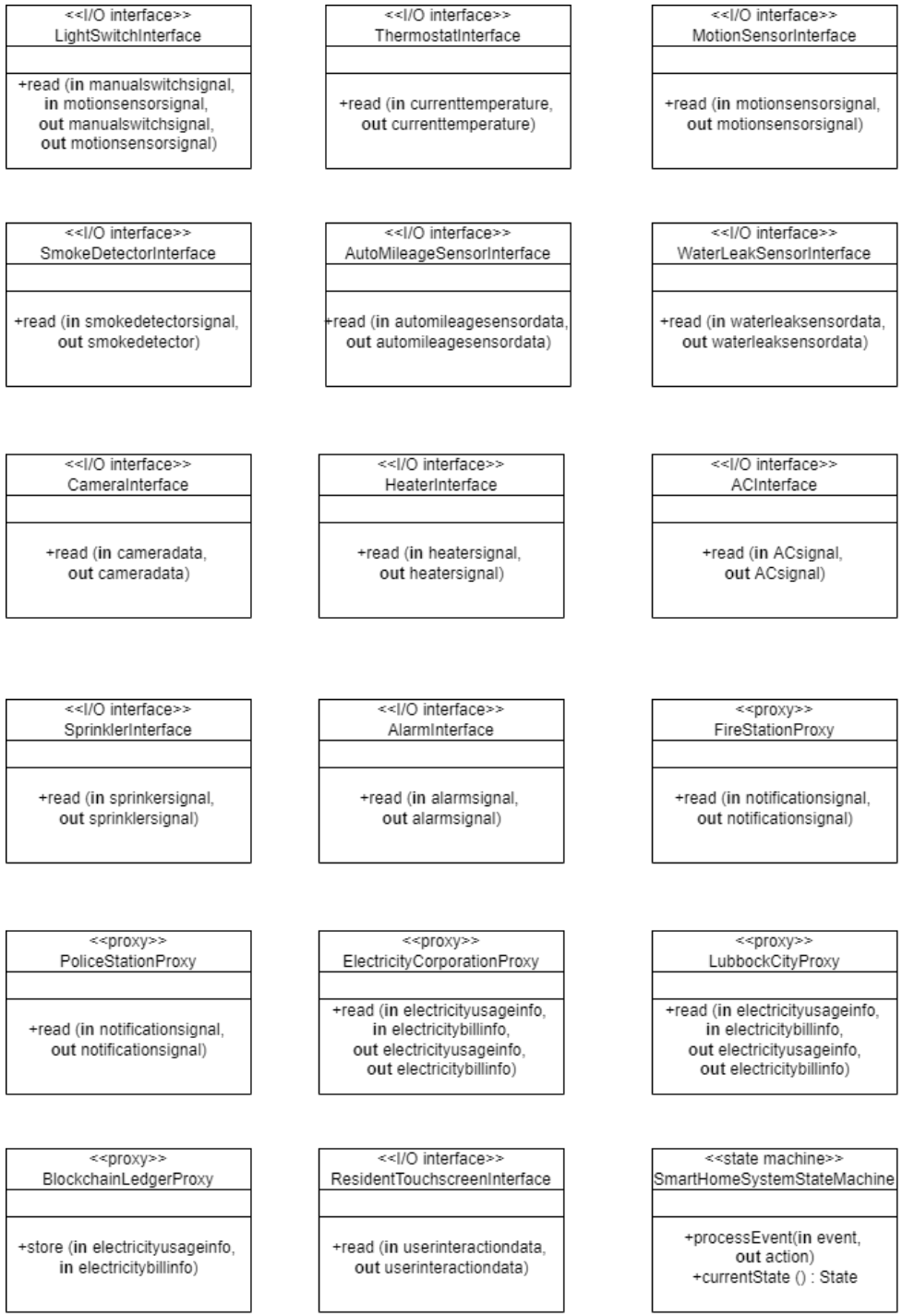
Task outputs:
External output: electricityBillInformation
Asynchronous message communication: None
Passive objects accessed: None
Errors detected: Invalid Electricity information send to Blockchain Ledger.

d) Define the information hiding classes in the system. (4 pts)

Assumptions:

1. The Smart Home System (SHS) stores a lot of data pertaining to a many specific instances of time which will require a database for data encapsulation. So, database wrapper class is used for information hiding.
2. alert_response1 implies notifying resident in case a suspicious person hangs around the house. alert_response2 implies notifying resident and police station if a suspicious person breaks into the house.
3. validate_response will display “Welcome” if credentials are valid and will display “Login error” if credentials are invalid.


<<database wrapper>>
TemperatureData
+record (in currenttemperature,
in desiredtemperature)
+set(in comfortabletemperature,
in minimumsafetemperature)
<<database wrapper>>
AutoMileageData
+read (in currentmileage,
in gasusage)
<<database wrapper>>
WaterLeakData
+record (in leaktime, in leakplace)
<<database wrapper>>
CameraImageData
+store (in imagedata)
<<database wrapper>>
ElectricityData
+record (in electricity usage,
in bill)
<<database wrapper>>
TimeLog
+record (in HeaterONtime,
in HeaterOFFtime,
in AC_ON_time,
in AC_OFF_time,
in TemperatureChangetime,
in WaterLeakTime,
in FireAlarmTime,
in SprinklerONtime,
in NotifyFireStationTime,
in HomeSecurityAlarmTime,
in HoSecurityNotificationTime,
in LogInTime,
in LedgerRecordTime)
<<database wrapper>>
ResidentCredentials
+store (in UserID,
in password, in EmailAddress,
in PhoneNumber)
+validate (in UserID,
in password)
<<Business Logic>>
GasUsageCalculationManager
+initialize ()
+calculate (in LatestGasUsage,
out mileage, out gas usage)
<<Business Logic>>
ImageAnalysisManager
+initialize()
+store (in image),
+analyze(in image,
out alert_response1,
out alert_response2)
<<Business Logic>>
CredentialVerificationManager
+initialize()
+read (in UserID, in password)
+validate(in UserID, in password)
+display(out validate_response)
<<Business Logic>>
BillCalculationManager
+initialize()
+read (in electricityusage,
in electricitybill)
+calculate(in electricityusage,
out electricitybill)


<<I/O interface>>
LightSwitchInterface
+read (in manualswitchsignal,
in motionsensorsignal,
out manualswitchsignal,
out motionsensorsignal)
<<I/O interface>>
ThermostatInterface
+read (in currenttemperature,
out currenttemperature)
<<I/O interface>>
MotionSensorInterface
+read (in motionsensorsignal,
out motionsensorsignal)
<<I/O interface>>
SmokeDetectorInterface
+read (in smokedetectorsignal,
out smokedetector)
<<I/O interface>>
AutoMileageSensorInterface
+read (in automileagesensordata,
out automileagesensordata)
<<I/O interface>>
WaterLeakSensorInterface
+read (in waterleaksensordata,
out waterleaksensordata)
<<I/O interface>>
CameraInterface
+read (in cameradata,
out cameradata)
<<I/O interface>>
HeaterInterface
+read (in heatersignal,
out heatersignal)
<<I/O interface>>
ACInterface
+read (in ACsignal,
out ACsignal)
<<I/O interface>>
SprinklerInterface
+read (in sprinkersignal,
out sprinklersignal)
<<I/O interface>>
AlarmInterface
+read (in alarmsignal,
out alarmsignal)
<<proxy>>
FireStationProxy
+read (in notificationsignal,
out notificationsignal)
<<proxy>>
PoliceStationProxy
+read (in notificationsignal,
out notificationsignal)
<<proxy>>
ElectricityCorporationProxy
+read (in electricityusageinfo,
in electricitybillinfo,
out electricityusageinfo,
out electricitybillinfo)
<<proxy>>
LubbockCityProxy
+read (in electricityusageinfo,
in electricitybillinfo,
out electricityusageinfo,
out electricitybillinfo)
<<proxy>>
BlockchainLedgerProxy
+store (in electricityusageinfo,
in electricitybillinfo)
<<I/O interface>>
ResidentTouchscreenInterface
+read (in userinteractiondata,
out userinteractiondata)
<<state machine>>
SmartHomeSystemStateMachine
+processEvent(in event,
out action)
+currentState () : State