IoT Smart Home System

This smart home system aims to solve the issues that exist in the smart home ecosystem. This new system allows users by setting up their smart devices in an easy and fun way. The process to set up a device will be less than one minute allowing users to set light bulbs, thermostats, smart vacuum, TVs and other devices in a very short amount of time with minimum effort thus allowing for a fun process. Its recommended that the devices are set up by users who are 18 and older. Users will be able to easily set up personalized schedules by user on the app or website.

Business Requirements

User Experience

* Easy to setup
* Friendly user interface on smart phone, tablet, wearable device
* Bluetooth and WIFI device detection at device set up
* Smart speakers with several microphones for easy user hearing

User Security and Privacy

* User data logs will not be shared with 3rd party
* User data will be encrypted
* Require multifactor user authentication

Cost

* Smart speaker cost should be $25
* App will be free without adds
* Member subscription will be based on annual and monthly basis
  + $120 per annual basis
  + $16 per monthly basis

Customer Support

* 24/7 customer support available via phone, email or chat
* Tutorials and other customer support material available on the app or the website

Non-Functional Requirements

Performance

* The response time between the cloud and the smart device should be less than 3 seconds
* Servers handling the app will use the latest sever processor with 10GB ethernet connectivity for faster response when receiving prompts

Scalability

* The servers will use load balancers by having 2 servers of equal technical specs, once one of the servers reaches a certain limit
* The server will use vertical scaling to easily upgrade the server hardware

Security

* The communication will use TLS withing the cloud and WAP3 for Wi-Fi communication on the user’s home
* All data will be encrypted per regulations
* Patches and security updates will be done in a weekly basis

Back Up and Recovery

* Backups will be done on a daily basis to protect historical user data

User Authentication

* Users will be prompted to use two-factor authentication by proving either their email address or phone number
* User will be authenticated by prompting users for a physical address
* When users add new user to their home system, they will be prompted to send the new user a link so it can be added either as admin or normal user with less privileges

Code Modularity

* Encapsulation will be used through the software development to enable flexibility when new features are added and old features are removed
* API usage will be necessary though the software development, specially when connecting the external API for payment and other search availabilities
* Reusable code will be part of the software implementation to increase performance when the app features are being used

Documentation

* Code will be documented on GitHub when developers add or remove from code
* The usage of API’s will be document to keep control of which APIs will be used and what they do for the app, also in case the Api receives a security update

Testing

* Software testing will be done by in-house tester using unit testing for better performance
* Automated testing will be used on software that has been implemented before and only takes few inputs

UML Use Case

Use Case 1: Smart Thermostat

* Use Case ID: UC-ST-001
* Name: Smart Thermostat
* Primary Actor: Homeowner, Smart Thermostat, App, Smart Speaker, guest
* Secondary Actors: Internet Provider
* Preconditions: homeowner calls for the thermostat to change the temperature at homeowners preferences.
* Postconditions: thermostat will set the appropriate temperature desired by the homeowner.
* Description: the thermostat set the desired temperature by the homeowner either by manually adjusting it through the app or by setting up a personalized schedule

Use Case 2: Smart Lights

* Use Case ID: UC-SL-001
* Name: Smart Lights
* Primary Actor: Homeowner, Smart Light Bulbs, App, Smart Speaker
* Secondary Actors: Internet Provider
* Preconditions: Homeowner can control the lights by asking the smart speaker to do so, by going into the app and do it manually or by setting up a schedule
* Postconditions: The lights will turn on or off
* Description: The lights will be controlled by the homeowner via the app, smart speaker or the set up schedules, also the homeowner has the ability to control the lights manually by flipping the wall switch in case internet is not available at that moment

Use Case 3: Smart Cameras

* Use Case ID: UC-SC-001
* Name: Smart Cameras
* Primary Actor: Homeowner, Smart Cameras, App, Smart Speaker
* Secondary Actors: Internet Provider, Law Enforcement
* Preconditions: Homeowner sets up the cameras at a discrete and location preferred by the homeowner, then check via the app the live or historical feed
* Postconditions: The camera will detect any movement in its perimeter and provide live feed to the homeowner
* Description: the cameras can be set up at a location that the homeowner wants it. They will provide live and historical feed to the homeowner, and if the homeowner decides to corroborate a break in or incident, he/she can provide the historical data to law enforcement

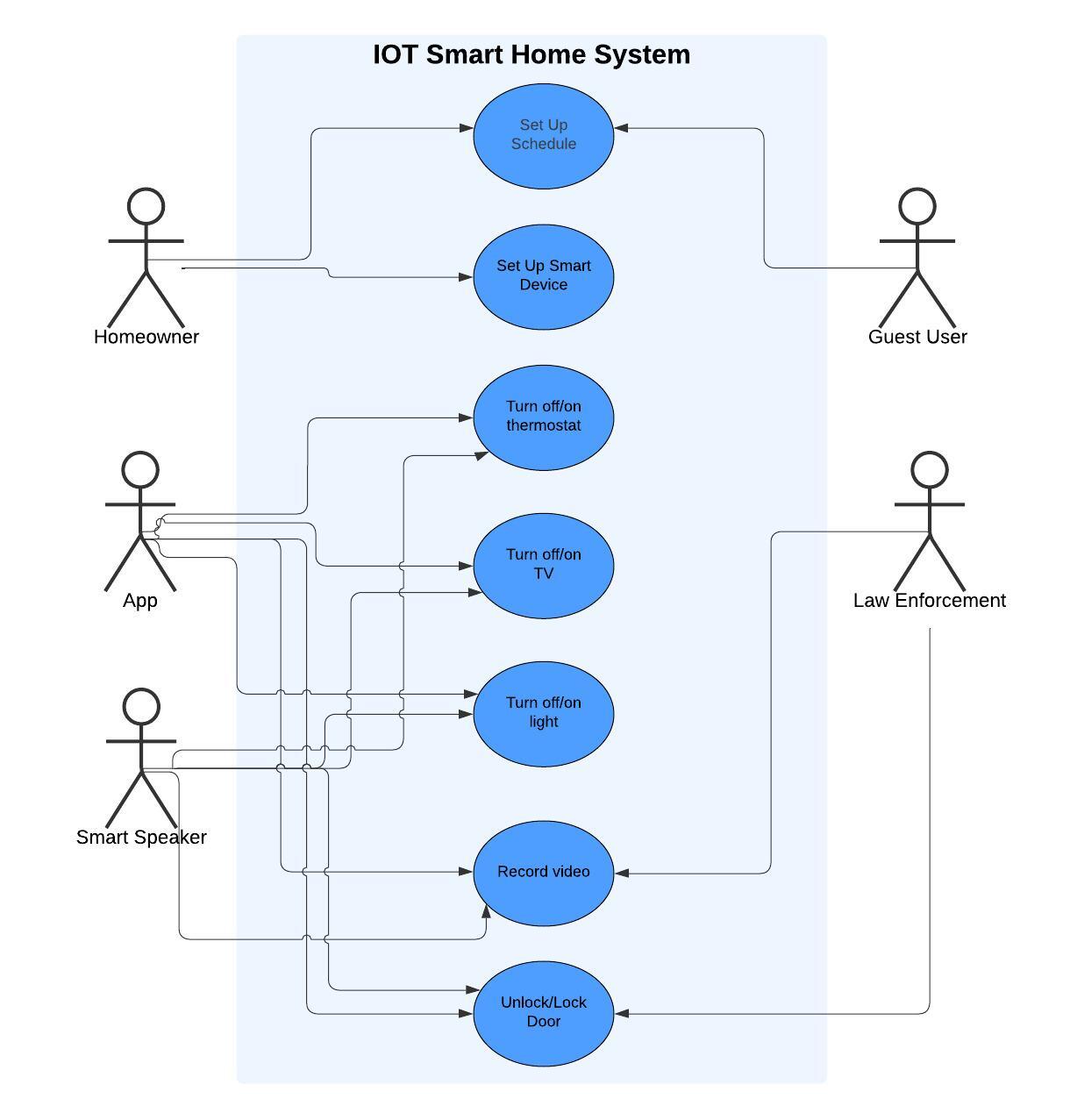
Use Case 4: Smart Keyless Entry

* Use Case ID: UC-SKE-001
* Name: Smart Keyless Entry
* Primary Actor: Homeowner, Smart Lock, App, Smart Speaker
* Secondary Actors: Internet Provider, Law Enforcement
* Preconditions: Homeowner sets up the smart lock on the door of their preference
* Postconditions: The door will be unlocked or locked by the smart speaker or the app, it can also be unlocked manually
* Description: homeowner can set up a smart lock device system to enter the house by opening the app. lock the door by asking the smart speaker or using the app, even manually so. And if a break-in happens, it can inform law enforcement

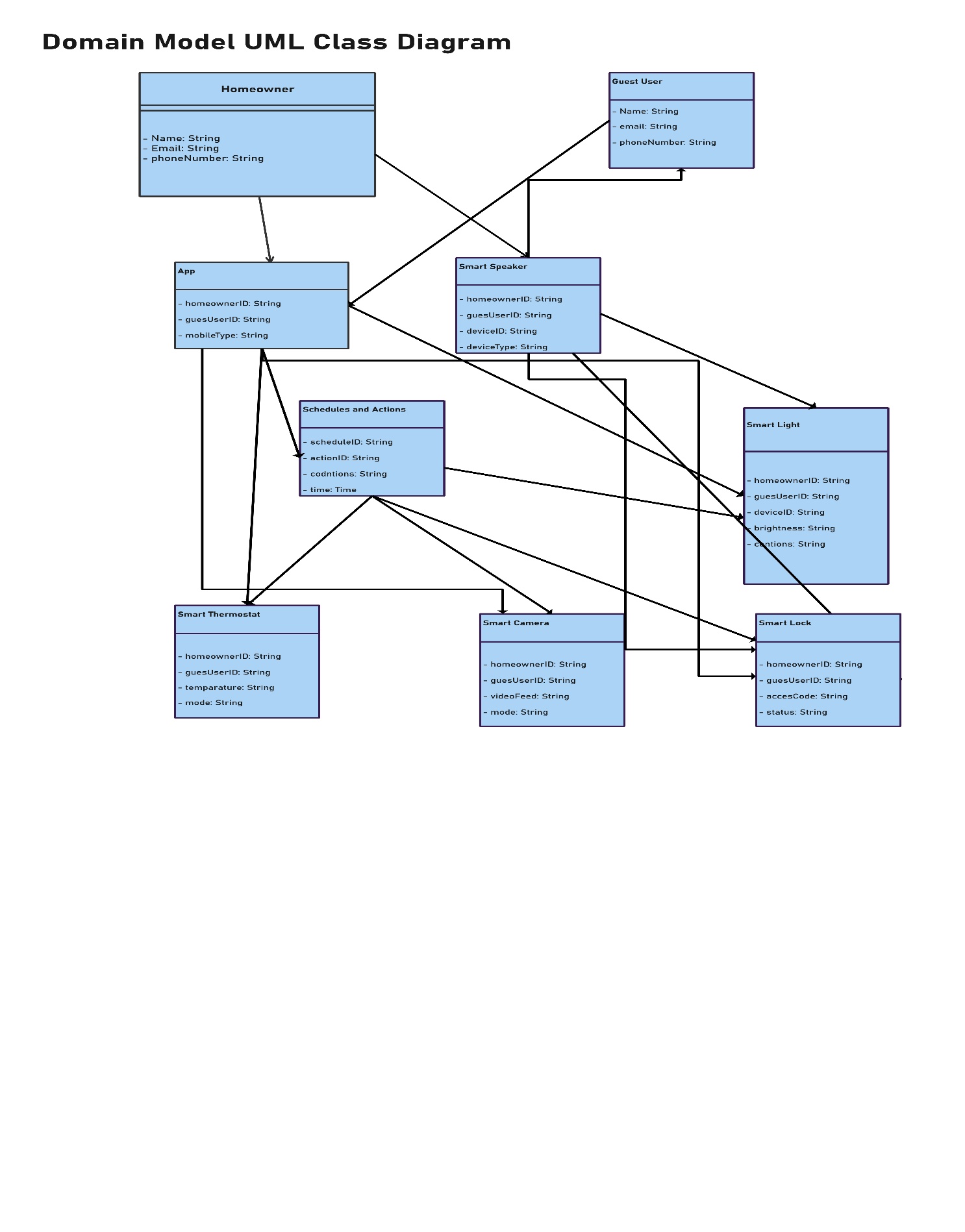
Use Case 5: Smart TV

* Use Case ID: UC-SL-001
* Name: Smart TV
* Primary Actor: Homeowner, Smart TV, App, Smart Speaker
* Secondary Actors: Internet Provider
* Preconditions: Homeowner can control the TV by speaking to the smart speaker or by accessing the virtual remote controls provided by the app
* Postconditions: The TV will be controlled by the app or the smart speaker
* Description: The TV can be controlled by the smart speaker or the app and change channels using the app or the smart speaker,

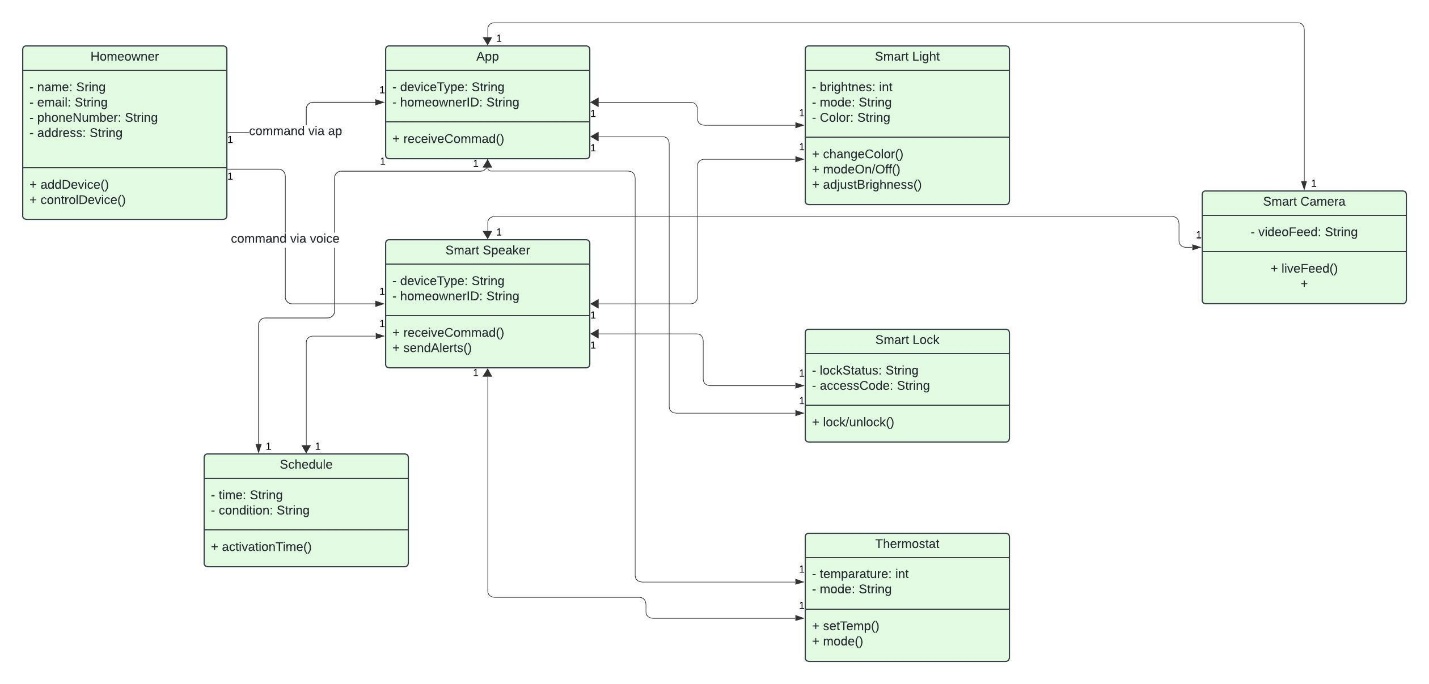
Use Case Diagram



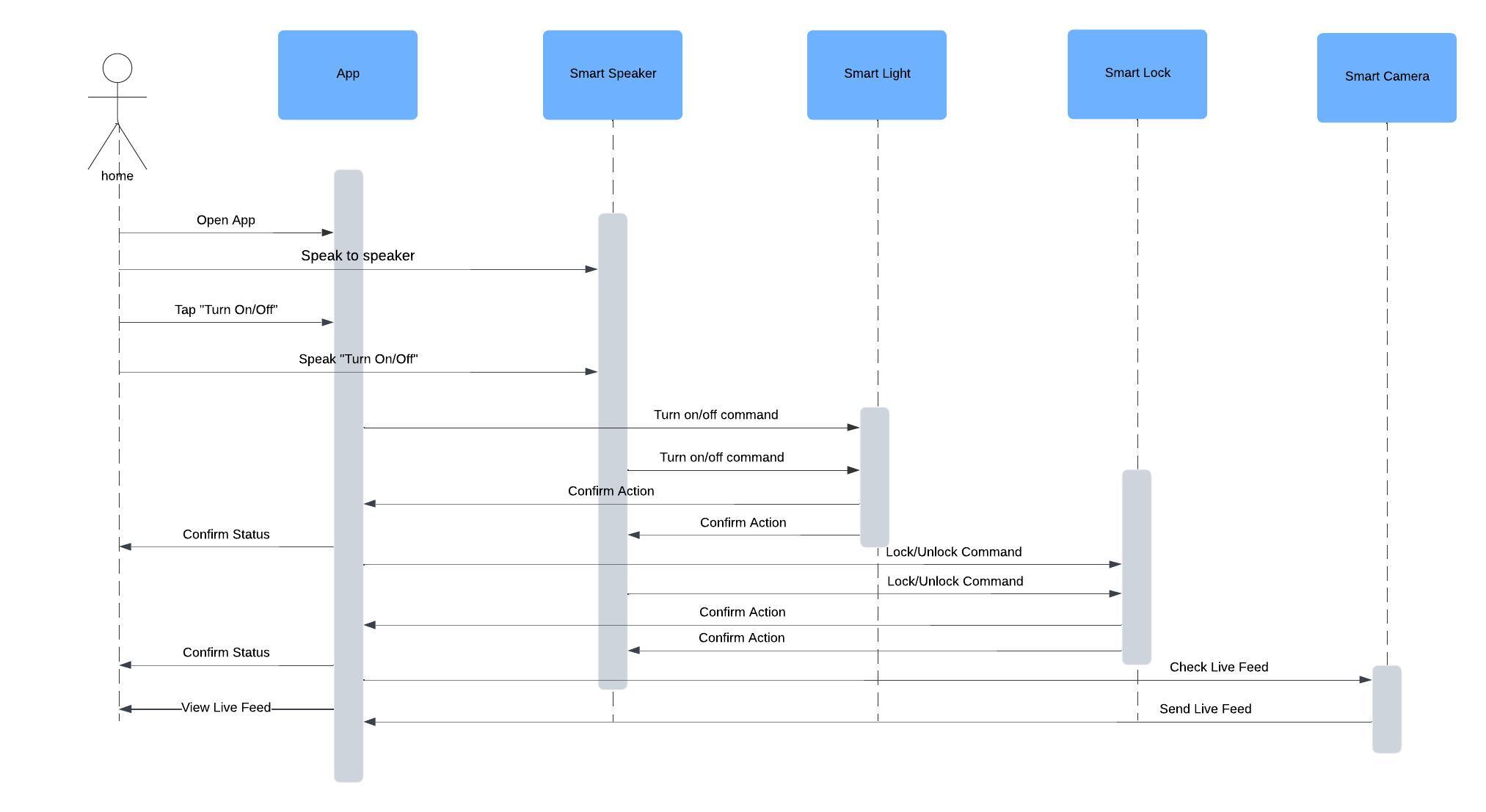
Domain Model



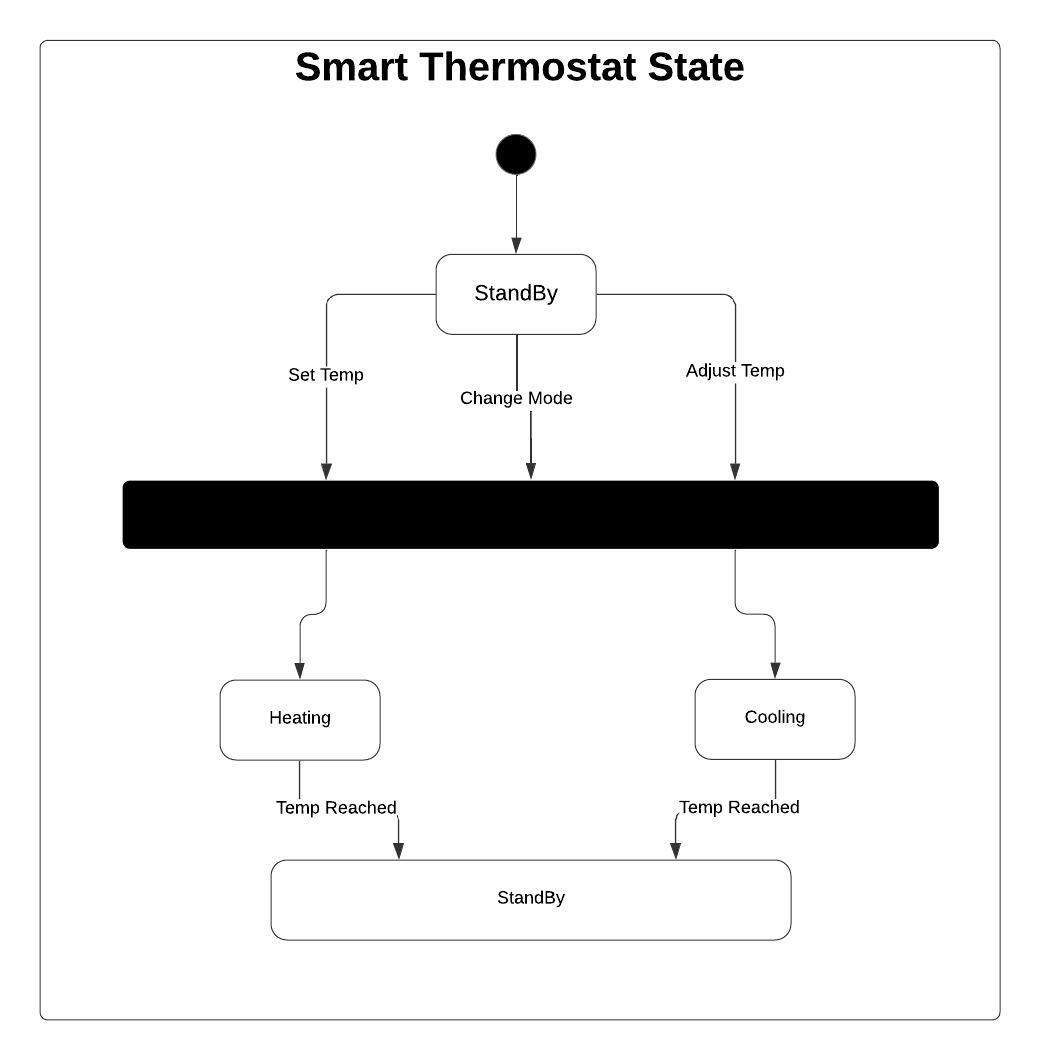
Class Diagram



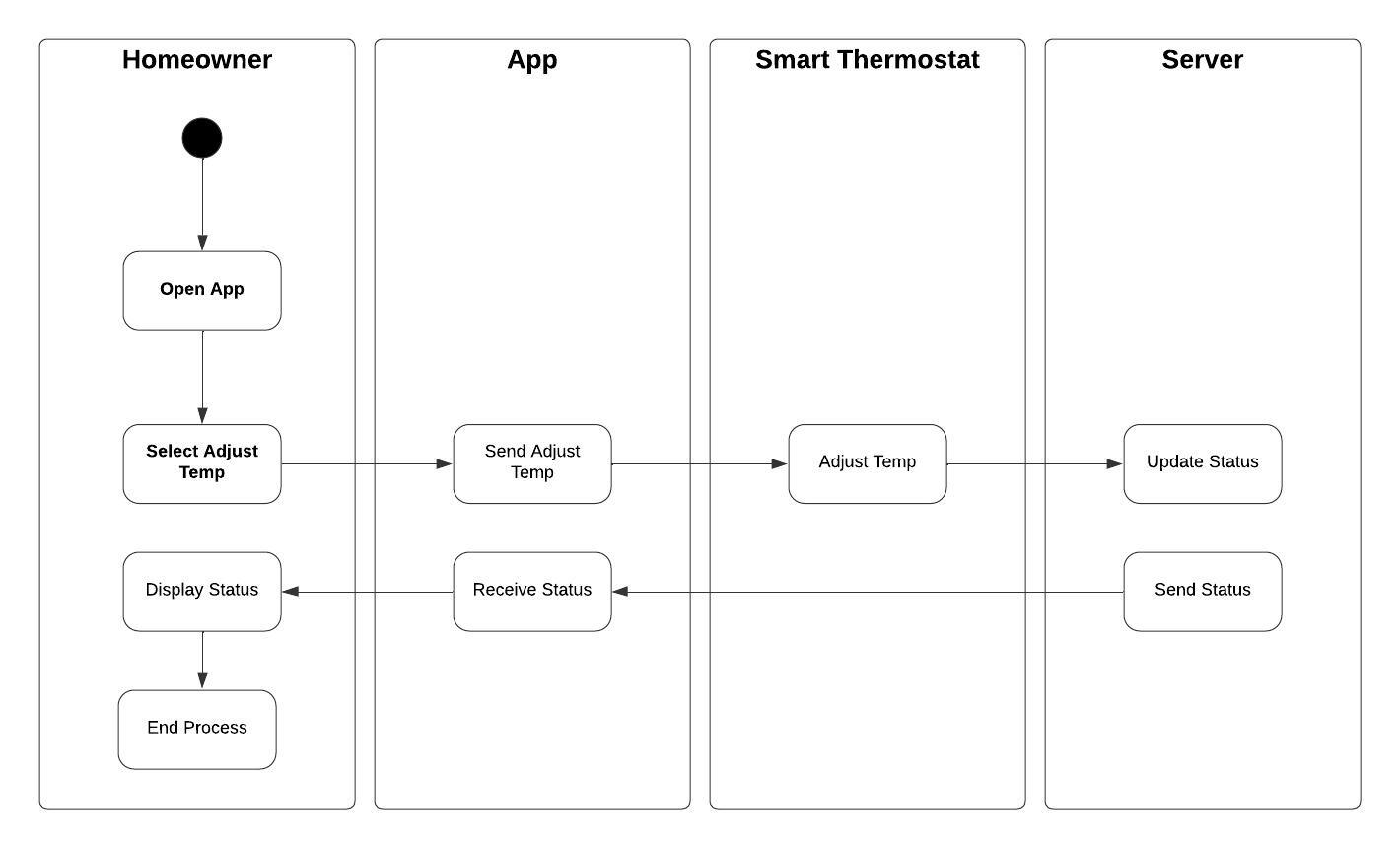
Sequence Diagram



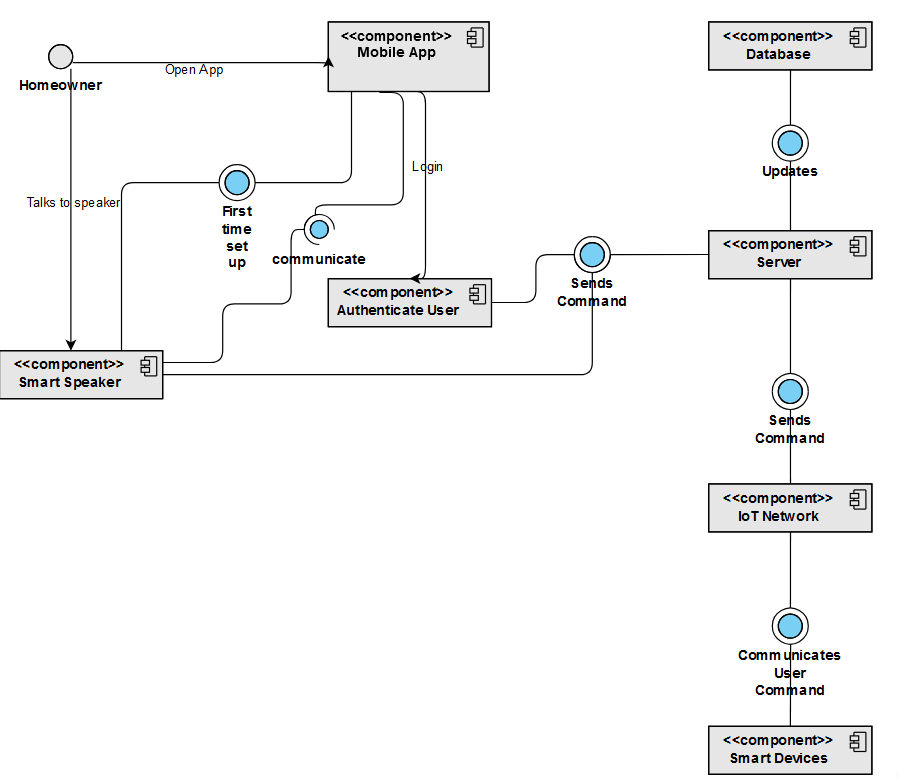
UML State Diagram



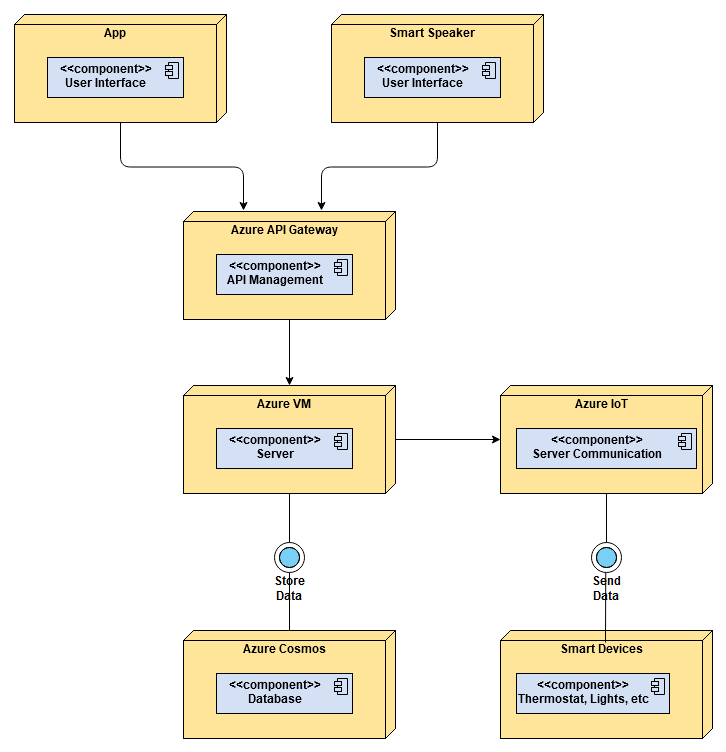
UML Activity Diagram (Swimlane Diagram)



Component Diagram



Deployment Diagram



Skeleton Classes and Tables Definition

**User**

Attributes

* name: String
* email: String
* phoneNumber: String
* address: String

Methods

* addDevice()
* controlDevice()

**App**

Attributes

* deviceType: String
* homeownerID: String

Methods

* receiveCommand()

**Smart Light**

Attributes

* Brightness: int
* Mode: String
* Color: String

Methods

* changeColor()
* modeon/off()
* adjustBrightness()

**Smart Camera**

Attributes

* videoFeed: String

Methods

* liveFeed()

**Smart Speaker**

Attributes

* deviceType: String
* homeowner: String

Methods

* receiveCommand()
* sendAlert()

**Smart Lock**

Attributes

* lockStatus: String
* accessCode: String

Methods

* Lock/unlock()

**Schedule**

Attributes

* time: String
* condition: String

Methods

* activationTime()

**Thermostat**

Attributes

* temperature: String
* mode: String

Methods

* setTemp()
* mode()

Database Structure

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | **Data** | **Data Type** | **Description** | | Name | String | User name | | Email | String | User email address | | phoneNumber | String | User phone number | | address | String | User physical address | | userID | String | Stores the user ID |   **Users Table** |  |  |
| **Thermostat Settings**   |  |  |  | | --- | --- | --- | | **Data** | **Data Type** | **Description** | | temperature | String | User temperature set by the user | | mode | String | This stores the mode by the user | | thermostatID | String | Stores the ID of the thermostat settings | |  |  |

**Smart Lights**

|  |  |  |
| --- | --- | --- |
| **Data** | **Data Type** | **Description** |
| Mode | String | This saves the mode of the user |
| Color | String | Saves the color of the user |
| lightID | String | Stores the primary key of the light |

Design Patterns

I used the microservice design by implementing the API gateway in order for the app and the smart speaker to communicate with the server and by using the api gatway will save time in development and the security updates will be taken care by the cloud provider. Also the use of GRASP by using the controller provide by the azure cloud in order for the user interface to be more responsive and for the system to user the cloud servers