## **Project Documentation: Smart Home Controller**

## **Table Of Content:**

[Project Documentation: Smart Home Controller 1](#_Toc1675431206)

[Table Of Content: 1](#_Toc614621246)

[1. Introduction 1](#_Toc1048080127)

[1.1 Project Overview 1](#_Toc140636459)

[1.2 Purpose 1](#_Toc1819640403)

[1.3 Objectives 1](#_Toc1440737437)

[2. System Features 1](#_Toc610703342)

[2.1 Key Features 1](#_Toc390466174)

[3. System Design 2](#_Toc274376326)

[3.1 Architectural Overview 2](#_Toc1035753641)

[3.2 Key Components 2](#_Toc1270657858)

[3.2.1 SmartDevice (Abstract Class) 2](#_Toc1085244971)

[3.2.2 Device Classes 2](#_Toc1612715348)

[3.2.3 Room 3](#_Toc1289936548)

[3.2.4 SmartHomeController (Main Class) 3](#_Toc1905267068)

[4. Requirements Specification 3](#_Toc996862983)

[4.1 Functional Requirements 3](#_Toc1919674014)

[4.2 Non-Functional Requirements 3](#_Toc322873463)

[5. User Guide 4](#_Toc1329792046)

[5.1 Launching the Application 4](#_Toc185409125)

[5.2 Managing Rooms 4](#_Toc54124289)

[5.3 Managing Devices 4](#_Toc1653372734)

[5.4 Device Control 4](#_Toc1598981804)

[5.5 Saving and Loading State 4](#_Toc167765000)

[6. Future Enhancements 4](#_Toc1041208590)

[6.1 Additional Device Support 4](#_Toc111437934)

[6.2 Remote Access 4](#_Toc6729938)

[6.3 Dashboard Integration 5](#_Toc2035700174)

[8. Class Diagram 5](#_Toc981075322)

## **1. Introduction**

### **1.1 Project Overview**

The **Smart Home Controller** is a Java-based application designed to manage and control smart devices within a household. The application provides a user-friendly graphical interface for adding rooms, managing devices, and controlling device states such as lights, thermostats, and cameras. State persistence is ensured through serialization, allowing users to save and resume their session seamlessly.

### **1.2 Purpose**

The application aims to simplify the management of smart devices in a home by offering an integrated, scalable, and intuitive solution. It empowers users with control over their smart devices while ensuring flexibility and reliability.

### **1.3 Objectives**

* Provide a graphical interface for managing rooms and devices.
* Enable device-specific control panels for intuitive interaction.
* Support the saving and loading of application state for continuity.
* Ensure scalability to accommodate new devices and features.

## **2. System Features**

### **2.1 Key Features**

1. **Room Management**:
   1. Add and remove rooms dynamically.
   2. Display a tabbed interface for room organization.
2. **Device Management**:
   1. Add and remove devices per room.
   2. Support for smart lights, thermostats, and cameras.
3. **Device Control**:
   1. Toggle device states (e.g., on/off, recording).
   2. Adjust device settings such as brightness and temperature.
4. **State Persistence**:
   1. Save the current application state to a file.
   2. Load a previously saved state for session continuity.
5. **Error Handling**:
   1. Comprehensive validations and error messages for robust operation.

## **3. System Design**

### **3.1 Architectural Overview**

The **Smart Home Controller** is implemented using an object-oriented design in Java, leveraging the Swing framework for GUI development. The system employs serialization for state persistence and ensures scalability through extensible class structures.

### **3.2 Key Components**

#### **3.2.1 SmartDevice (Abstract Class)**

* **Responsibilities**:
  + Acts as the base class for all devices.
  + Provides common functionalities such as turning on/off.
* **Methods**:
  + turnOn(), turnOff(): Control device state.
  + getControlPanel(): Abstract method for generating device-specific UI.

#### **3.2.2 Device Classes**

* **SmartLight**:
  + Attributes: Brightness, intensity control.
  + UI: Power toggle, brightness slider.
* **SmartThermostat**:
  + Attributes: Temperature, intensity control.
  + UI: Power toggle, temperature slider.
* **SmartCamera**:
  + Attributes: Recording state.
  + UI: Power toggle, recording toggle.

#### **3.2.3 Room**

* **Attributes**:
  + name: Name of the room.
  + devices: List of devices in the room.
* **Methods**:
  + addDevice(), removeDevice(): Manage devices in the room.

#### **3.2.4 SmartHomeController (Main Class)**

* **Responsibilities**:
  + Manages rooms and devices.
  + Provides GUI components such as menus, buttons, and tabs.
  + Handles state persistence.

## **4. Requirements Specification**

### **4.1 Functional Requirements**

* The system must allow users to add, remove, and manage rooms.
* The system must provide device-specific control panels.
* Users must be able to save and load the application state.

### **4.2 Non-Functional Requirements**

* The application should be intuitive and user-friendly.
* The system must handle errors gracefully.
* The solution should be scalable for future enhancements.

## **5. User Guide**

### **5.1 Launching the Application**

* Compile and run the SmartHomeController class to launch the application.

### **5.2 Managing Rooms**

* **Add Room**: Use the "Add Room" button and provide a unique room name.
* **Remove Room**: Use the "Remove Room" button and select a room from the list.

### **5.3 Managing Devices**

* **Add Device**: Navigate to a room tab, select a device type, and click "Add Device."
* **Remove Device**: Navigate to a room tab, select a device type, and click "Remove Device."

### **5.4 Device Control**

* **SmartLight**: Toggle power and adjust brightness.
* **SmartThermostat**: Toggle power and set temperature.
* **SmartCamera**: Toggle power and start/stop recording.

### **5.5 Saving and Loading State**

* **Save State**: Use the "Save State" button or File > Save State menu option.
* **Load State**: Use the "Load State" button or File > Load State menu option.

## **6. Future Enhancements**

### **6.1 Additional Device Support**

* Add support for new smart devices such as speakers or blinds.

### **6.2 Remote Access**

* Introduce a mobile application or web interface for remote control.

### **6.3 Dashboard Integration**

* Provide a centralized dashboard for an overview of all devices.

## **8. Class Diagram**

