Smart Home Automation System

Project Documentation

1. Introduction

A *Smart Home Automation System* is designed to enhance comfort, energy efficiency, and security by allowing users to monitor and control home devices (lights, fans, air conditioners, security systems, etc.) remotely or automatically. This project implements a user-friendly, role-based smart home system with automation, scheduling, and device customization features that are accessible via a graphical interface.

2. Objectives

- Remote and local control: Enable users to control and monitor home devices from anywhere via a desktop application.
- Role-based access: Implement authentication and authorization for Admin and Regular Users, restricting sensitive operations to admins.
- Automation: Support device automation (e.g., motion-activated lights, auto temperature adjustment in ACs).
- Scheduling: Allow users to schedule device operations (turn on/off, set temperature, etc.).
- Customization: Support device-specific features (e.g., light color selection, AC modes).
- Security: Integrate a security system with an alarm and motion detection.
- Extensibility: Design the system to easily add new devices and features.

3. System Architecture

3.1 Components

- User Interface (SmartHomeGUI): Java Swing-based GUI for user interaction.
- Core Logic (SmartHomeSystem): Handles authentication, device management, scheduling, and automation.
- Devices: Abstract base class with concrete implementations for Light, Fan, AirConditioner, and SecuritySystem.
- Users: Admin and Regular User classes with role-based permissions.
- Exceptions: For handling authentication and device errors.

- Interfaces: For device capabilities (Switchable, Dimmable).
- Scheduling: Allows users to automate device actions at specific times/days.

3.2 Device Features

Device	Features
Light	On/Off, Brightness control, Motion activation, Color selection (White, Warm, Blue, Red)
Fan	On/Off, Speed control (no oscillation/auto-temp adjust)
Air Conditioner	On/Off, Temperature (16-30°C), Modes, Energy Saving, Auto Temperature Adjust
SecuritySystem	On/Off, Alarm, Mode (Home/Away/Disarmed), Motion detection, Event logs

4. User Roles and Permissions

- Admin
    - Add/remove devices and users
    - View system logs
    - Full device control and scheduling
- Regular User
    - Control devices and schedule tasks
    - Cannot add/remove devices or users
    - Cannot view logs

Authentication is required for all users. Permissions are enforced throughout the application.

5. Automation & Scheduling

- Motion-activated lights: Lights in a room turn on automatically when motion is detected.
- AC Auto Temperature Adjust: The AC can automatically set the temperature based on the time of day for comfort and efficiency.
- Scheduling: Users can schedule device actions (e.g., turn on the light at 7 pm every day).

6. User Interface

- Login Screen: Secure login for all users.
- Main Dashboard: Device list, system controls, user info.
- Device Control Panel: Device-specific controls (brightness, color, speed, temperature, etc.).
- Device Scheduling: Add/view/remove scheduled tasks.
- Admin Features: Add/remove users/devices, view logs.

All controls are permission-aware and visually intuitive.

7. Key Classes and Structure

```
src/
    └─ smarthome/
       ├─ SmartHomeApp.java
       ├─ SmartHomeGUI.java
       ├─ system/
       │    └─ SmartHomeSystem.java
       ├─ models/
       │    ├─ Device.java
       │    ├─ Light.java
       │    ├─ Fan.java
       │    ├─ AirConditioner.java
       │    ├─ SecuritySystem.java
       │    ├─ ScheduledTask.java
```

```
│   ├─ User.java
│   └─ AdminUser.java
├─ interfaces/
│   ├─ Switchable.java
│   └─ Dimmable.java
└─ exceptions/
    ├─ AuthenticationException.java
    └─ DeviceNotFoundException.java
```

8. Example Use Cases

- Login: User enters credentials; system grants access based on role.
- Device Control: User selects a device, adjusts settings (e.g., light brightness, AC temperature).
- Automation: Light turns on automatically when user enters a room.
- Scheduling: User sets AC to turn on at 6pm on weekdays.
- Admin Management: Admin adds a new user or device.

9. Extensibility

The system is designed for easy expansion:

- New device types can be added by extending the Device class.
- New automation rules can be implemented in SmartHomeSystem.
- GUI updates automatically reflect new device features.

10. Steps to run: Right click on the SmartPhoneApp class

- Select the option RunAs>1 Java Application
- A window will pop, enter the username and password
- For admin- enter username as admin and password as admin123

11. Conclusion

This Smart Home Automation System demonstrates a robust, extensible, and user-friendly approach to home automation. Combining role-based security, automation, scheduling, and device customization provides a comprehensive solution for modern

smart homes. The modular design ensures future enhancements can be integrated with minimal effort.