**Software Requirements**

**Specification**

**for**

**Smart Home Control System**

**Version 1.0**

**Prepared by**

|  |  |
| --- | --- |
| Roaa Mamdoh Mohamed | 202220123 |
| Farah Hazem Hamed | 202220081 |
| Tasabeih Talaat Shabaan | 202220236 |

|  |  |
| --- | --- |
| Instructor: | Mohamed Reda |
| Course: | Software Engineering |
| Teaching Assistant: | Mohamed Karam |
| Date: | *<*date of submission*>* |

# Table of Contents

[Table of Contents 2](#_Toc181811587)

[Revision History 2](#_Toc181811588)

[1. Introduction 1](#_Toc181811589)

[1.1 Purpose 1](#_Toc181811590)

[1.2 Document Conventions 1](#_Toc181811591)

[1.3 Intended Audience and Reading Suggestions 1](#_Toc181811592)

[1.4 Product Scope 1](#_Toc181811593)

[1.5 References 1](#_Toc181811594)

[2. Overall Description 1](#_Toc181811595)

[2.1 Product Perspective 1](#_Toc181811596)

[2.2 Product Functions 2](#_Toc181811597)

[2.3 User Classes and Characteristics 2](#_Toc181811598)

[2.4 Operating Environment 2](#_Toc181811599)

[2.5 Design and Implementation Constraints 2](#_Toc181811600)

[2.6 User Documentation 2](#_Toc181811601)

[2.7 Assumptions and Dependencies 2](#_Toc181811602)

[3. External Interface Requirements 3](#_Toc181811603)

[3.1 User Interfaces 3](#_Toc181811604)

[3.2 Hardware Interfaces 3](#_Toc181811605)

[3.3 Software Interfaces 3](#_Toc181811606)

[3.4 Communications Interfaces 3](#_Toc181811607)

[4. System Features 4](#_Toc181811608)

[4.1 System Feature 1 4](#_Toc181811609)

[4.2 System Feature 2 (and so on) 4](#_Toc181811610)

[5. Other Nonfunctional Requirements 4](#_Toc181811611)

[5.1 Performance Requirements 4](#_Toc181811612)

[5.2 Safety Requirements 5](#_Toc181811613)

[5.3 Security Requirements 5](#_Toc181811614)

[5.4 Software Quality Attributes 5](#_Toc181811615)

[5.5 Business Rules 5](#_Toc181811616)

[6. Other Requirements 5](#_Toc181811617)

[Appendix A: Glossary 5](#_Toc181811618)

[Appendix B: Analysis Models 6](#_Toc181811619)

[Appendix C: To Be Determined List 6](#_Toc181811620)

# Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
|  |  |  |  |
|  |  |  |  |

# 1. Introduction

## 1.1 Purpose

This Software Requirements Specification (SRS) document outlines the functional and non-functional requirements for the Smart Home Control System. This system enables users to control and automate various devices and appliances in their home, enhancing comfort, convenience, and security. The system allows remote control of smart devices such as lights, thermostats, locks, cameras, and more, all from a mobile device. The initial release of the system is version 1.0.

# 1.2 Document Conventions

High Priority: Essential features that are critical for the system’s basic functionality.

Medium Priority: Features that enhance user experience but are not essential.

Low Priority: Features that can be implemented in future versions but are not immediately required.

TBD: To be determined.

## 1.3 Intended Audience and Reading Suggestions

Developers: Understand the technical requirements and constraints for app development.

Project Managers: Use the document to track project scope and deadlines.

Testers: Refer to the test cases and validation requirements.

End-users: While not directly reading the SRS, end-users should be considered during the design of user-friendly interfaces and experiences.

The document is organized as follows:

* Section 2: Overall description of the product and system context.
* Section 3: External interface requirements (User, Hardware, Software, Communications).
* Section 4: Functional requirements organized by system features.
* Section 5: Non-functional requirements.
* Section 6: Additional requirements.

## 1.4 Product Scope

The Smart Home Control System enables users to monitor, control, and automate home appliances through a mobile interface. The system integrates with various smart devices, creating a seamless ecosystem for smart homes. Its primary goals are to increase convenience, security, energy efficiency, and cost savings by automating routine tasks and offering remote control.

## 1.5 References

Smart Home Automation: Concepts and Benefits, Imaginovation Blog, 2024.

Relevant third-party device integration guides (e.g., Amazon Alexa, Google Assistant, Apple Siri).

# 2. Overall Description

## 2.1 Product Perspective

This system is a multi-purpose solution for smart home automation, enabling users to control a variety of devices from different manufacturers through one integrated system. It serves as an all-in-one hub for device management, monitoring, and automation.

The system will be connected to a cloud backend for remote access, ensuring data is available on demand. The system's functionality is reliant on device compatibility with various communication protocols such as Wi-Fi, Bluetooth, ZigBee, and others.

## 2.2 Product Functions

The Smart Home Control System provides the following high-level functions:

* Remote Control: Turn devices on/off and adjust settings (e.g., lights, thermostat).
* Voice Integration: Control devices via voice commands using assistants like Alexa, Google Assistant, and Siri.
* Alerts & Notifications: Receive real-time notifications about device status changes, security alerts, and environmental changes (e.g., smoke or water leaks).
* Automation: Set schedules and triggers for devices (e.g., lights off at bedtime, thermostat adjusts when nearing home).
* Customizable Scenes: Create and activate preset routines like "Movie Night" or "Vacation Mode".
* Energy Analytics: Track energy consumption patterns and provide insights into usage for cost-saving measures.

## 2.3 User Classes and Characteristics

* End-User (Primary): Homeowners or renters who wish to control their smart home devices remotely. They may have varying levels of tech-savvy.
* Family Members: Users may have different access levels, such as allowing children to control lights but preventing access to locks or security features.
* Device Manufacturer: For integration purposes, external device manufacturers provide APIs and protocols for the system to communicate with their devices.

## 2.4 Operating Environment

* Mobile Platforms: iOS and Android devices (smartphones and tablets).
* Smart Devices: Compatible with lights, thermostats, security cameras, door locks, etc.
* Cloud Storage: For remote access and backup (cloud storage like AWS, Google Cloud, or Azure).
* Communication Protocols: Wi-Fi, Bluetooth, ZigBee, Z-Wave, MQTT for device communication.

## 2.5 Design and Implementation Constraints

* Device Compatibility: The system must support a broad range of smart home devices and communication protocols.
* Security: All user data must be encrypted, and communication between devices should be secured.
* Performance: The system should handle multiple devices efficiently and deliver a seamless experience without significant latency.

## 2.6 User Documentation

* User Manual: Provides guidance on setting up devices, configuring automation, and troubleshooting common issues.
* Help & FAQs: Integrated into the app for easy access to support.
* Online Tutorials: Step-by-step guides for configuring devices and creating routines.

## 2.7 Assumptions and Dependencies

* **Internet Connection: A stable internet connection is required for remote access and device control.**
* **Device Compatibility: The app assumes that the connected devices support the relevant communication protocols (e.g., Wi-Fi, ZigBee).**
* **Cloud Storage Service: Assumes the availability and reliability of the selected cloud storage provider.**

# 3. External Interface Requirements

## 3.1 User Interfaces

* Mobile App Interface: Clean, intuitive UI with control options for each device, scenes, and schedules.
* Voice Control: Integration with Alexa, Google Assistant, and Siri for hands-free operation.
* Alerts & Notifications: Real-time notifications and alerts with customization options.

## 3.2 Hardware Interfaces

* Smart Devices: Devices like smart thermostats, locks, lights, and cameras must support cloud integration via Wi-Fi, ZigBee, or Bluetooth.
* Sensors: Temperature sensors, motion sensors, and leak detectors must be compatible with the system’s communication protocols.

## 3.3 Software Interfaces

* Third-party APIs: Integration with device manufacturers' APIs for data exchange and remote control.
* Cloud Backend: Integration with cloud services to store user preferences, device settings, and history.

## 3.4 Communications Interfaces

* Wi-Fi / Bluetooth / ZigBee: Communication protocols used for connecting the system to devices and controlling them.
* Cloud Communication: Use of secure protocols like HTTPS for communication between the mobile app and the cloud backend.

# 4. System Features

**4.1** Device Control

* Description: Allows users to remotely control devices in their home (lights, thermostat, locks, etc.) via the app.
* Priority: High
* Functional Requirements:

REQ-1: The app must allow users to turn devices on or off remotely.

REQ-2: The app must allow users to adjust settings of devices (e.g., thermostat temperature, light brightness).

**4.2** Voice-Based Control

* Description: Enables users to control smart devices using voice commands.
* Priority: Medium
* Functional Requirements:

REQ-3: The app must integrate with voice assistants such as Amazon Alexa, Google Assistant, or Apple Siri.

**4.3** Notifications & Alerts

* Description: Sends automatic notifications for important events (e.g., device status changes, security alerts).
* Priority: High
* Functional Requirements:

REQ-4: The app must send alerts for smoke detection, water leaks, or when a door/window is left open.

REQ-5: The user must be able to customize notification preferences.

**4.4** Customizable Scenarios

* Description: Allows users to set up customized automation routines (e.g., "Movie Night").
* Priority: Medium
* Functional Requirements:

REQ-6: The app must allow users to create, save, and trigger scenarios involving multiple devices.

# 5. Other Nonfunctional Requirements

## 5.1 Performance Requirements

* Response Time: Commands should execute within 2 seconds, and critical notifications should be delivered within 5 seconds.
* Device Handling: The app must support up to 50 devices per user without performance issues.
* Data Syncing: Sync settings and preferences between devices and cloud should take < 3 seconds.
* Battery Efficiency: The app should use <10% of battery during continuous background operations.

## 5.2 Safety Requirements

* Device Failures: Prevent conflicting commands and alert users if devices fail to respond.
* Emergency Protocols: Certain functions (like unlocking doors) should be disabled during security breaches or emergencies.
* Data Backup: Automatically back up critical data to prevent loss.

## 5.3 Security Requirements

* Authentication: Implement two-factor authentication (2FA) and biometric options.
* Encryption: Use end-to-end encryption for data in transit and at rest.
* Privacy Control: Allow users to manage data privacy and opt-out of data sharing.
* Incident Response: Lock down functions and alert users in case of security breaches.

## 5.4 Software Quality Attributes

* Usability: The app should be intuitive with easy setup and navigation.
* Reliability: Ensure 99.5% uptime and graceful failure handling.
* Scalability: The app must support growing numbers of devices and users.
* Interoperability: Integrate with multiple smart devices and protocols (e.g., ZigBee, Wi-Fi).
* Maintainability: Well-documented codebase to support updates and bug fixes.
* Robustness: Handle errors gracefully and implement failover mechanisms.

## 5.5 Business Rules

* Role Management: Users (Admin, Family Member, Guest) have different permissions.
* Device Ownership: Only the primary user can add or remove devices.
* Scene Customization: Admin can create automation routines, and all users can access predefined scenes.
* Data Retention: Comply with privacy laws (e.g., GDPR), allowing users to delete accounts and data.

# 6. Other Requirements

* Database Requirements: Use a cloud-based database with real-time synchronization and data backup.
* Internationalization: Support multiple languages and adjust units like temperature based on locale.
* Legal: Comply with privacy laws like GDPR and CCPA, with clear user consent mechanisms.
* Reusability: Ensure the app's code is modular and reusable across platforms.
* Hardware Integration: Support devices from various manufacturers and communication protocols (Wi-Fi, Bluetooth, ZigBee).
* Accessibility: Ensure the app is accessible to users with disabilities, including features like voice control and screen reader compatibility.

# Appendix A: Glossary

Defines terms like:

* Smart Home: A home with remote-controlled devices.
* Home Automation: Controlling home appliances via the internet.
* Hub: Central device managing communication between devices.
* Interoperability: Compatibility with devices from different manufacturers.
* GDPR/CCPA: Data privacy laws.

# Appendix B: Analysis Models

Includes diagrams like:

* Data Flow Diagram: Illustrates how data moves in the system.
* Use Case Diagram: Shows user roles and their interactions.
* Class Diagram: Defines the structure of the app and main classes.
* Entity-Relationship Diagram: Depicts the database schema.
* Activity Diagram : Focuses on the sequence of actions and how components interact with each other.
* Sequence diagram: represents the interaction between objects in a system over time

# Appendix C: To Be Determined List

* Device Compatibility: Final list of supported devices.
* Communication Protocols: Final decision on supported protocols.
* Privacy Policy: Final privacy policy and user consent documents.
* API Integrations: List of third-party integrations (e.g., Alexa, Google Assistant).
* Localization Plan: Languages and regions for the initial release.