SOFTWARE REQUIREMENTS SPECIFICATIONS

FOR

SMART-DOOR-UNLOCK

Prepared By	
Name	**SID**
Daksh Gupta	20103026
Yash Kapur	20103117
David Makhija	20103095
Pranav Aggatwal	20103066

Punjab Engineering College, Chandigarh

Table of Content

Revision History

Name	**Date**	**Reason For Change**	**Version**
Smart-door-unlock	01/09/2022	Initial Version	1.0

1. Introduction

1.1 Purpose

The purpose of this document is to outline the system requirements for smart-door-unlock which will aim to improve the security and accessibility of doors. This also provides reduction of the dependence on physical methods keeping in mind the ever-growing idea of digitization of home security and smart connection systems.

The software should allow access (unlocking and locking a door, automatically by a click of a button from the web application) of a particular door by creating a ‘environment’ for the multiple trusted members of the house/hostel.

It will keep a track of the history of when, where and who accesses a particular door. These records will be privately visible to every member of the ‘environment’.

1.2 Document Convention

Font Face	Arial
Heading font size	20px
Sub Heading font size	14px
Table / Sub sub-heading font size	13px
Content font size	12px

Apart from this, special highlighting of important keywords has been done to make it differentiable. Every requirement stated in this document has its own unique priority and every functionality is equally important.

1.3 Intended Audience and Reading Suggestions

While the software requirement specification (SRS) document is written for a more general audience, this document is intended for individuals directly involved in the development of smart-door-unlock.

This includes software developers, project consultants, and team managers. This document need not be read sequentially; users are encouraged to jump to any section they find relevant. Below is a brief overview of each part of the document.

The developers will coordinate every activity taking place in the Software Engineering process and will be guided by Prof. Rajesh Bhatia.

1.4 Definition and Acronym

Term	**Full Description**
Admin	Person responsible for the upkeep, configuration, and reliable operation of the system
Users	Anyone using the system
DFD	Data Flow Diagram
API	Application Programming Interface
Arduino	A micro-controller board that stores the set-of-instructions and actually provides motion in smart-door-unlock.
React	JavaScript framework for frontend UI development.
Python	A programming language which will be used for any machine learning aspect of application.
Flask	Python based backend code development library used to communicate with the frontend and the database.
SQL	Structured Query Language that will permanently store the records in global database.

1.5 Overview

The remaining part of document comprises of **4** sections as follows:

- **Section- 2:** It describes the product perspective, its functional requirements and stakeholders involved in the system, along with the non-functional requirements like scalability, performance etc.

- **Section- 3:** It describes all the interfaces comprising the user interface specifications, hardware interfaces and software interfaces.

- **Section- 4:** Index.

2. Overall Description

2.1 Product Perspective

The perspective of this system is to allow users the ability to access their homes or other locations in a safe and secure manner without the threat of insecurity. We hope to enable all aspects of security to be digitally manageable in a safe, secure and transparent manner and this particular system is a step in that direction.

2.2 Product Functions

The functions of our software are as follows: -

- Configure a username and password for the server and allow devices to gain access to the server.
- The connected devices can then communicate with server to access doors from anywhere
- The server can perform actions on the door:
    i. Lock the door
    ii. Unlock the door using servo motor.
- Create a database that maintains a record of actions performed on the shared door for complete trust and transparency

2.3 User Requirements

∉ **Minimum user age: -** The user should be of **minimum 15 years old** as this product is related to security and some maturity in its operation is required

∉ **Hardware: -**

1. Arduino Uno Smart-door handle
2. A good internet connection.
3. Smart-door server.

2.4 Process Constraints

Thinking of the edge case when 2 or more environment members run into a conflict of accessing the door at the same moment of time while the first process is being run. This situation may end up in unfulfilled requirements since the latter person’s request is to be kept on hold until the former request is completed.

3. Specific Requirements

3.1 Functional Requirements

Functional Feature	**Remarks**
Authentication	Authenticate users using their phone number.
Create environment	Setting up a new target door and creating the group with access rights to members.
Join environment	Getting access to a target door.
Lock the target door	Locks a given door and saves the record in database and notifies every other member of environment.
Unlock the door	Unlocks the door and saves in database.

3.2 Non-Functional Requirements

3.2.1 Performance Requirements

SMART-DOOR-UNLOCK can handle up to 32,767 door environments. This limitation is caused by vertical scaling of SQL that only enable 32,767 documents per each database.
The response time of opening the door bottles to 10th of seconds throttling upon the motor power and upon the data communication between the frontend and the backend setup for each user in proximity to their target door.

3.2.2 Security Requirements

Since the door access is to be created by an owner, the owner should carefully give access to other members. Since once a member owns the environment, they are allowed to lock and unlock the door at anytime from anywhere.

3.3 Constraints

Hardware Requirement

- Arduino Uno
- Smart-door-unlock personal server

Software Requirement

- **Flask** (A python runtime environment)
- **SQL** (Database system)
- **Arduino IDE** (Seet of instructions plugged into Arduino Uno)
- **React.js** (JavaScript library for frontend development).

INDEX