IoT Solution for Rental Houses

Nuno Afonso Rebelo Patrício Freire dos Santos
Instituto Superior Técnico
Lisboa, Portugal
nuno.afonso.p.s@tecnico.ulisboa.pt

ABSTRACT
The tourism growth led to an increase of short-term accommodation. However, the management of a property may be time consuming and difficult to achieve. This paper proposes an Internet of Things' system to assist people in performing house management tasks, integrating a smart lock, utility monitoring and appliance control in a smart home system.
The proposed solution uses various smart devices connected to sensors and actuators, to monitor and control a house's environment. Those devices use a Wi-Fi network to communicate to a home server, which is responsible for processing the data. Additionally, it offers a web interface to users, and connects to a central server that allows the management of multiple rental houses from a single point. A prototype was developed that uses ESP8266 boards as smart devices. For the home server a Raspberry Pi was chosen. The system was tested, in which commands were sent through the Raspberry Pi to the ESP nodes, to change an environment's property. Using the prototype, it was possible to conclude that the proposed system provides significant assistance in the management of short-term accommodation businesses.

Author Keywords
Smart House Rental; House Management; IoT; Smart Home; Smart Device; ESP8266.

INTRODUCTION

The world population upsurge and worldwide development lead to a growth in tourism, as the number of people travelling from a country to another increases. According to World Bank Group's data, between 2009 and 2019, Portugal doubled the number of tourist arrivals and the European Union countries, combined, saw the number of tourists raise almost 40% [1]. Additionally, for the same period, the aggregated number of arrivals increased from 1.6 billion to 2.2 billion, for all countries in the world, as depicted on Figure 1.

In December 2019, the virus SARS-CoV-2 spread around the world, which lead to travel restrictions imposed by governments, thus reducing tourism. This work accounts that, despite these limitations, once these restrictions subside, the number of international arrivals will increase, following the aforementioned growth..

Motivation

The management of short-term accommodation requires a significant investment by a person responsible for the administration of a property, which may be the property's owner, a contracted person, or any other entity responsible for the asset. In the context of this document, this person shall be referred to as a house manager, or administrator. Accommodation management can be a complicated task in situations where the administrator has more than one property to manage, and a reduced number of employees, or people, that could help perform this task. Costumers arriving at the same time in different locations can be impeditive, as they might require the house key and the manager may not be available to open the door. Moreover, the administrator might want to access information, such as utility consumption during a client's reservation, and even turn off any appliances that have been left on, once the tenant leaves the property, without traveling to the it.

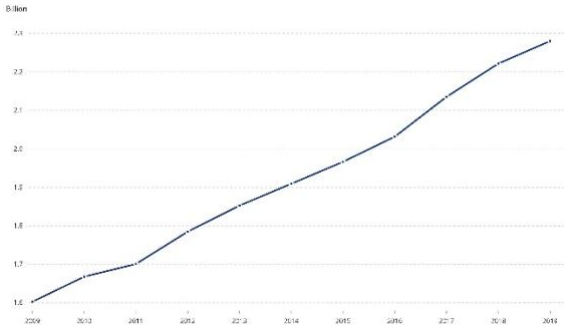

Figure 1. International tourism, aggregated number of arrivals, in billions, between 2009 and 2019 [2] (adapted).

Technology's purpose, among others, is to help individuals and organizations to be more productive. The use of information technology can support the accommodation industry, assisting users to manage their properties. It should be taken into consideration that an embedded computer has been defined by Perry Xiao as being a small size, low-cost computer system, embedded in hardware [3]. Internet of Things (IoT) is a concept introduced in the aggregation of smart devices (objects with an embedded computer), interchanging data between each other through wired or wireless communications, thus creating a network on which they can operate together to reach common goals
[4].

The work presented in this paper aims to provide comfort to the administrator, and to a client renting a house, by creating a smart house system. This is an IoT system integrated in a house environment, that provide contextaware actions, resulting in what appears to be intelligent behaviour (home automation). The term *domotics* can also be used in this context, referring to a smart house.

Objectives

It is required that the system controls digital access to the property, enabling access to a registered user and restricting it to undesired people.

Additionally, this domotic solution needs to allow a person to monitor the house's utility usage, by metering the water, energy, and gas consumption, and assist in the control of the house appliances, allowing, for example, to remotely power them on, prior to the client's arrival, and, after their departure, verify if the client left one of these devices turned on, allowing them to be turned off when desired. By providing these solutions, the system aids the management of the property and helps improve the costumer's stay. Lastly, the system must authorize the administrator, through a user interface, to remotely manage all properties, by communicating with the house's devices, without compromising the system's security.

Document's Structure

Following this introduction, the "IoT Solution for Rental Houses" project presentation is composed of a section dedicated to the state-of-the-art solutions applied to smart home rental, and another to identify the system requirements and present a solution to solve them. Additionally, a section is used to describe a prototype designed and another to describe a set of tests conducted to evaluate the system.

Finally, a last section is used to provide some final remarks.

RELATED WORK

The following section will focus on state-of-the-art approaches that can be applied to this project, including communication protocols and scientific papers regarding developed projects.

Wireless Communication Protocols

Responsible for transmitting data between two entities, the common wireless communication protocols used in IoT are presented in this subsection. They allow for the creation of star networks, where devices are connected to a central unit, and mesh networks, where they are connected between each other, as illustrated in Figure 2.

Bluetooth Low Energy

Bluetooth, a personal area network, uses 2.4GHz radio frequency to provide wireless communication between a central unit, the master, and other devices, the slaves, forming a start network [5]. Bluetooth Low Energy is an improved version of this protocol, where entities form a star or a mesh network, and can communicate consuming less energy, transferring encrypted messages with a transmission rate of up to 1Mbps.

ZigBee

Zigbee uses 2.4GHz frequency to create a personal area network, enabling data share between devices. Built to improve IoT communication, it is a low cost and low power consumption protocol, that uses a mesh network to deliver encrypted messages at lower transmission speeds, providing higher propagation distances, by hopping messages through other ZigBee compatible devices [6]. However, the device cost is quite elevated, as it requires a fee for each product, therefore becoming a disadvantage.

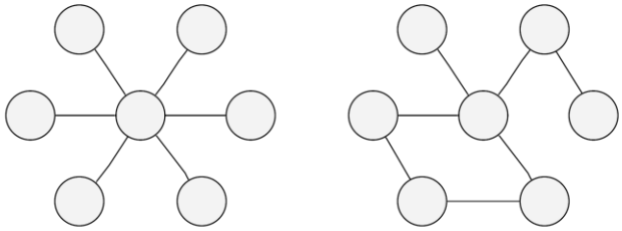

Figure 2. Star (left) and mesh (right) network topology.

Z-Wave

Z-Wave is a protocol designed to be integrated into smart home devices through low energy radio waves, setting a wireless mesh network in which other Z-Wave devices broadcast an encrypted message, granting secure communication at a highly reduced power consumption with reduced interference, as it uses lower frequencies (around 0.9GHz) [7]. Z-Wave biggest downside being the device cost, supported by the fact this protocol is a proprietary technology, requiring a license for the protocol to be integrated in a device.

Wireless Fidelity

One of the most frequently used communication protocols is a wireless Local Area Network (LAN) named Wireless Fidelity (Wi-Fi), in which devices can connect to a central unit, a wireless router, that creates a 2.4 or 5GHz radio frequency network in which devices can communicate between each other, through it, using a half-duplex connection [8]. Wi-Fi, being used in most habitations, can cover large distances, at increased data propagation speeds, with a relatively low power consumption, thus having been chosen to integrate the proposed system.

Application Layer Protocols

Application layer protocols are used by an entity running an application, enabling communications by transferring data between hosts.

Some of the most used application protocols in distributed systems and IoT are presented in this subsection. It should be noted that TCP based protocols can integrate Transport Layer Security, allowing a secure communication channel to be

established between two peers, by encrypting each entities' messages.

Constrained Application Protocol

Constrained Application Protocol (CoAP) was introduced in 2014 as a request and response protocol, similar to Hypertext Transfer Protocol (HTTP, or HTTPS when encrypted), specially designed for IoT systems, as entities may access resources requesting *GET* (retrieves an object), *POST* (alters a resource on the server), and *DELETE* (removes an asset from an application) methods to a Uniform Resource Identifier (URI), representing the endpoint to which the message is sent and an asset. Additionally, replies may have a payload, containing additional data, and a *status code*, where a *200* symbolizes a request successfully processed, and client error would be specified by a *400* [9].

This protocol has been created for devices which required low power consumption and low overhead connections. with the objective of decreasing their power consumption, achieved with the use of UDP.

Being based on UDP, CoAP requires a packet identification strategy, in which messages are re-transmitted, if an *acknowledgement* is not received within a defined limit of time, thus providing a reliable connection between devices.

Message Queuing Telemetry Transport

Message Queuing Telemetry Transport (MQTT or MQTTS when encrypted) is a publish-subscribe protocol designed for the IoT, as it consumes less computational resources, then HTTP, thus being an advantage to operate with lowpower devices.

A publish-subscribe architecture requires three types of entities: the publisher, the subscriber, and the broker. The publisher is responsible for generating and publishing data into the broker which can then send the message to a subscriber, acting like the middleman, being responsible for coordinating the system's information [10].

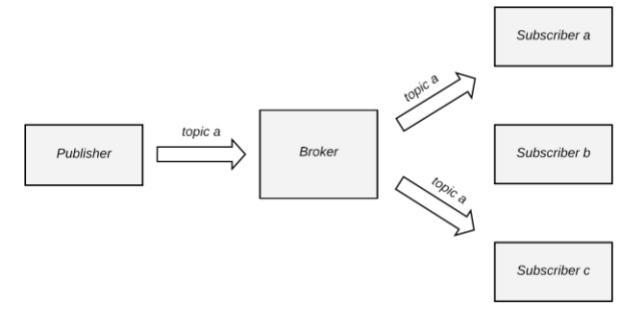

Figure 3. MQTT's Publish-Subscribe pattern. Figure 3 provides an example of a simple MQTT system, containing a broker that broadcasts a message, published to *topic a*, to *Subscriber a* and *Subscriber c*, but not to *Subscriber b*, as the latest not subscribed to that topic.

WebSocket Protocol

The WebSocket protocol, or WSS when encrypted, is based on TCP that enables a client to send information to a server, but also allows the last to freely push information to the client, without requiring a request to be made, thus providing a full-duplex communication [11]. This communication requires an initial *handshake*, performed by an HTTP request, after which communication is kept open until one of the two entities disconnects, allowing an entity to detect the *status* on another. This *handshake* is the main disadvantage of the protocol, as it leads to a large overhead whenever the communication channel is initiated. For this reason, it is not desired to use the protocol for situations where connections are opened for a single request. The rate packets are sent (throughput), for these three main IoT protocols, can be compared using Figure 4, which led the authors concluded that despite WebSocket's overhead, this protocol performs as good as CoAP better an MQTT, when sending reliable messages over a long-term connection.

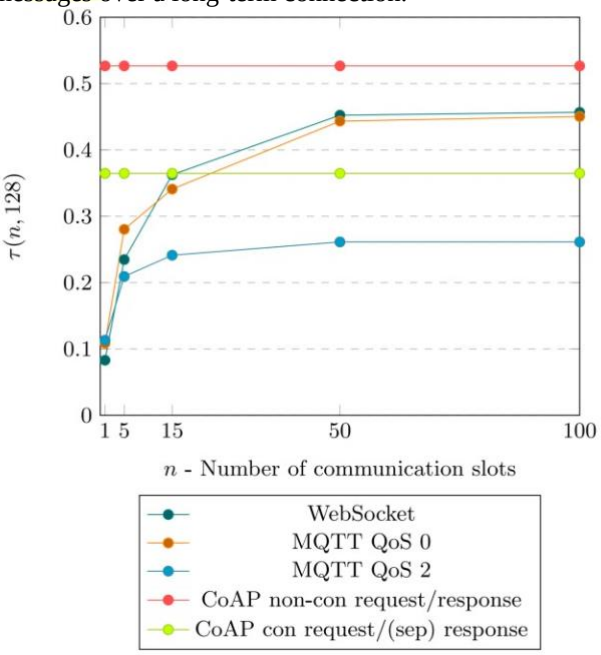

Figure 4. IoT protocols communication throughput with a simulated packed loss of 20%[12].

Academic Projects

This subsection focuses on summarizing home automation projects described in scholarly literature, in specifying door lock and utility energy systems relative to property rental.

Door Authentication

Door authentication allows a person with a specific key access a room, or a house.

Y. T. Park, P. Sthapit, and J.-Y. Pyun created a digital lock system, that allows a person to unlock a property's door by introducing an access code in a keypad, or by using Radio Frequency Identification to detect the presence of possession of a specific tag [13]. Moreover, it uses a set of sensors, such as a temperature and gas detectors, allowing the system to

recognize fire hazards, using Short Messaging Service (SMS) to notify a person when these situations are detected. The communication between these sensor nodes and a central controller is provided by Zigbee, reducing the system's power consumption.

M. Paveli´c *et al.* developed a project that uses an ESP8266 based microcontroller to control an electric lock, using secured WebSockets to communicate between the microcontrollers and a web server, which manages the system data, by storing it in a database, which can then be accessed by a user [14].

Utility Monitoring

Utility monitoring systems ensure users can supervise a house's environment, by measuring power, water, and gas consumption.

K. Luechaphonthara and A. Vijayalakshmi proposed an IoT project using sensors for a microcontroller to detect the alternating or the direct current being consumed by an appliance, using a Wi-Fi connection to propagate the sensed data to a Cloud hosted server [15]. This last entity aggregates the data sensed by the device and stores it in a database, allowing for a user to access it, thus monitoring the house's environment over time.

Che Soh *et al*. developed a system to monitor the water consumption of a house using a water flow sensor to meter [16]. This data is perceived by a microcontroller, which uses MQTT to send the data to a Cloud server, so that a user can access it, through a dashboard. Additionally, this server uses emails and SMS to send notifications to the user, in situations where the consumption surpasses a defined threshold.

ARCHITECTURE

This section presents the home automation system architecture designed to solve the house rental and property management problem. For this reason, it includes a section in which the system requirements are presented, followed by another in which the most relevant communication protocols, selected for a real-time home automation system, are identified. Finally, the architecture of the system is described.

Requirements

A home automation system which focuses on assisting a person managing a property in short-term accommodation scenarios are required to contain a set of specifications, with the goal of achieving the purpose they were designed for.

Reservation Management

Reservation management features represent the defined system traits that provide house managers the possibility to define and control the house tenant's reservations.

First, the solution should provide a web application, through which inexperienced users may access with ease, to manage and control the system and its entities, thus storing the information, in particular client reservations. Additionally, it is required that the system concedes privileged users (the property managers) the ability to control and monitor the house prior, during and after a client's rent period.

This system is designed for homeowners. These may only wish to rent part of a property, such as a room, an entire house, or both. For this reason, the system should provide owners with the possibility to integrate several properties in a single solution, managing them from a single application.

However, the application may be accessed by different types of users. for this reason, it is required that these have restricted access to the application, depending on their role.

Home Automation

The system must also provide a set of features that allow managers to observe changes in a house's environment, such as verifying if an appliance has been left powered on. Moreover, owners may provide tenants with additional comfort before and during their accommodation period, by controlling the environment properties. A solution for this can be achieved with a domotic system, using smart devices to sense and actuate on the house's surroundings, which should be provided by an application.

An example of a smart device whose integration in a home automation system for rental house management is a smart lock, allowing for tenants to have scheduled access to the property.

It is desired that the system is able to store data whenever there a user defined event has occurred, ensuring house managers may observe changes in the house environment, throughout time. The system must also provide a set of alert features, with the goal of notifying user after an event occurred, ensuring its detection and a rapid response, from any user, in case an event occurs.

Communication

Communication between system entities is an important factor to take in consideration when designin a distributed solution.

First, having identified the necessity to manage several houses in a single solution, it is desired that this task can also be achieved remotely, thus ensuring control over the system over the Internet, without requiring a manager to enter the property in order to configure, monitor and control the system in real-time, or close to real-time. In case a message fails to be sent or processed, the system should asseverate the message delivery. Security is also an important aspect to consider, when creating a home automation project, as it controls a person's assets. For this reason, access to the system should be provided only to authorized users and machines, minimizing the vulnerabilities of the solution. Additinally, messages exchanged in the system should use secure protocols to ensure transmitted data cannot be access by malicious people.

Lastly, as there is a variety of home automation devices available in the market, from different companies, offering diverse functionalities, the solution should provide owners

with the possibility to incorporate devices regardless of their communication technology and messaging format.

System Entities

To surpass the adversities presented, a home automation system has been designed, connecting numerous devices and utilities via the house's network. By taking advantage of Wi-Fi connections inside a house, this system connects home automation instances using this protocol, allowing for messages to be rapidly sent across devices, without requiring additionally hardware to be obtained by the owner, as most properties already rely on this technology, with HTTP being used to exchange information between a server and a web-browser. Additionally, the WebSocket protocol allows entities to communicate in real-time, being used in machine-to-machine communications. These protocols must take advantage of TLS, ensuring a message's integrity is maintained.

Taking into account the protocols proposed, this section describes the system entities that provide an hybrid to solve the proposed problem.

Smart Device

A house's object or appliance can be turned into a smart device, provided that its tasks can be performed by an electronic component. For this reason, a smart device should contain, at least, one property (sensor or actuator) capable of interacting with the house's environment.

Smart Controller

A microcontroller entity responsible for the control of, at least, one smart device is called a Smart Controller. This entity should have Wi-Fi integration, in order to establish a connection to another entity, via a software interface that allows two applications to communicate. This intermediary is called an Application Programming Interface (API).

Proxy

It is proposed that the controllers use the WebSocket protocol to communicate with an API. However, it should be noted that, as previously mentioned, smart devices are not required to utilize this application protocol to communicate. For this reason, the system should contemplate the use of a proxy to translate messages between different communication protocols, thus receiving messages from a Smart Controller and forwarding them to the house's central unit, using the WebSocket protocol, and *vice-versa*.

Gateway

Wi-Fi is the communication protocol proposed to integrate this system. However, in resemblance to a proxy, it is suggested that the project includes IoT gateways, responsible for connecting a network, whose Smart

Controllers utilize a different protocol, to the house's Wi-Fi network.

Router

As the system relies on a Wi-Fi connection, for entities to exchange data, it needs a Wireless Local Area Network router, responsible for generating a secure wireless network and manage the Wi-Fi packets flow within it. network, forwarding a packet sent from entities to another. Additionally, it works as a home gateway, redirecting packets sent from the Internet to the house server, via WiFi.

Home Server and Database

Lying within a property's limits, the Home, House, or Local Server is an independent unit that provides users with an application, in which they can oversee the system, allowing the management of reservations and system entities. Additionally, the application allows the control and monitoring of the house's appliances.

As data is required to be persisted on the server, this unit utilizes a connection to a dedicated database. The main objective of this second entity is for the server to store information relative to the property.

This server works as an API, called by both Smart Controllers, proxies and gateways with the objective of connecting the entire network to the users.

Cloud Server and Database

Responsible for integrating several homes into a single solution, the Cloud provides users the capability of remotely controlling and configuring the components of a house, therefore fulfilling one of the most important project requirements: allowing a manager to access the house without being connected to its LAN.

Using the Cloud's server, it should be possible for privileged users to access a house's data, ensuring it is management from afar. By dedicating a database to this server, users may access a property's data without requesting information from the House Server, reducing the number of messages, exchanged between the instances. In similarity to a Home Server, this entity acts as an API, managing connections between the House Servers and a user.

Web Browser

A person may access the system using the web application, provided by one of the system servers, through a web browser.

System's Architecture

After defining the system entities and the protocols used for them to communicate, it is possible to provide the architecture of the system. Figure 5 provides an overview of the designed system architecture, where an abstract connection may be wired, wireless or a loop-back connection. On the left, drawn in light blue, the components of a smart home (local entities) can be found. A server (Sl), connected to a database (Bl), and Wi-Fi using controllers (Cw), composed by any type of smart devices (D, depicted with a lock, a meter, a thermometer and light bulb icons).

All of these, connected by the router (R).

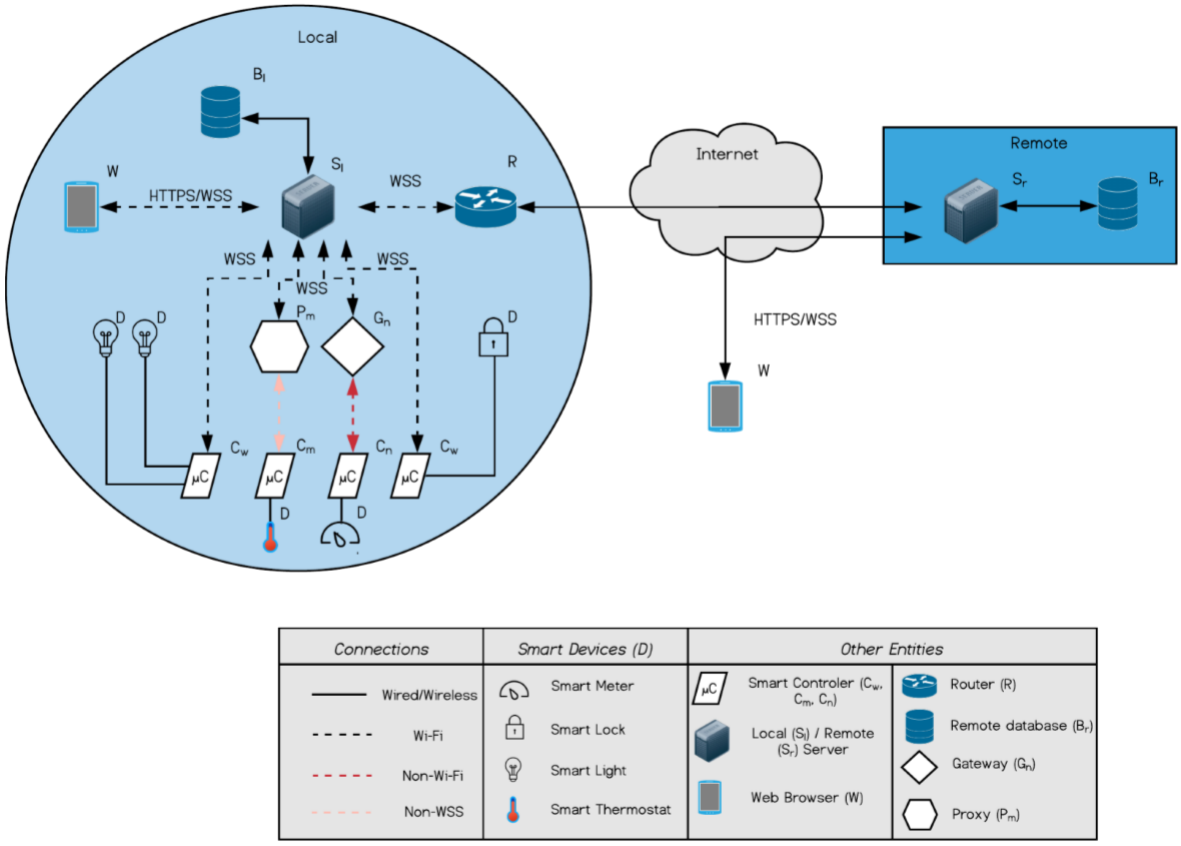

Figure 5. System Architecture

On the left, drawn in light blue, the components of a smart home (local entities) can be found. A server (Sl), connected to a database (Bl), and Wi-Fi using controllers (Cw), composed by any type of smart devices (D, depicted with a lock, a meter, a thermometer and light bulb icons). All of these, connected by the router (R).

The system also allows the owner to integrate Wi-Fi supported controllers, that do not use WebSockets, thus requiring a proxy to connect to the Home Server. The controller, Cm, transfers data to the server, using an application protocol m! = WebSockets, thus requiring a proxy, Pm, to connect to it.

Similarly, the owner may introduce controllers that utilize other communication protocols. The controller, Cn, identifies a controller that transfers data using a communication protocol n, such as BLE, ZigBee, etc (excluding Wi-Fi), which requires a gateway, Gn, for that same protocol.

In darker blue, on the right side of the image, the system's remote entities are designed. These are composed by a server (Sr) and a database (Br), ensuring properties can be managed and monitored from outside of the house's network.

The architecture also contemplates the presence of a user, via a client web browser (W), which can be connected to the house's server using the property's LAN, or to the Cloud Server, by accessing the Internet. In both situations, data is exchanged using HTTPS or WSS. This layered architecture allows managers to extend the application by adding different smart devices to it and access them from a single application, the Cloud Server.

SYSTEM IMPLEMENTATION

With the goal of testing the designed architecture, a prototype has been developed in which smart devices, servers and databases have been included. This section focuses going one step further and describing the implementation decisions taken, to establish the architecture described, detailing some of the system characteristics.

Figure 6. NodeMCU (left)[17] and Raspberry Pi 3B+ (right)[18].

Smart Door

A smart door device, or smart lock, was created, using a NodeMCU, shown in Figure 6, for a person to access the inside of a house, using a code to grant him/her access to the property.

This system allows the storage of up to three codes, allowing managers to define more than one code to be associated to different people, for example: one master code to be used by the manager and temporary for tenants which can be revoked after their reservation.

Components

This device been developed using two sensors and two actuators: a keypad, a reed switch, a relay, and a red-greenblue light emitting diode.

The first sensor, the keypad, is composed of 16 push buttons which can be used to insert the code. The first actuator, a relay module, is used for simulating the locking and unlocking of a door.

The light emitting diode is used by the controller to signal the user, exhibiting a green color whenever the door is unlocked, and a red color in situations where the inserted code was invalid. The last module of the device, the reed switch, uses a magnetic field to close a circuit. allowing the controller to perceive the state of the door by detecting the presence of current. Figure 7 illustrates the wiring used to for the controller to group the five modules into a single device.

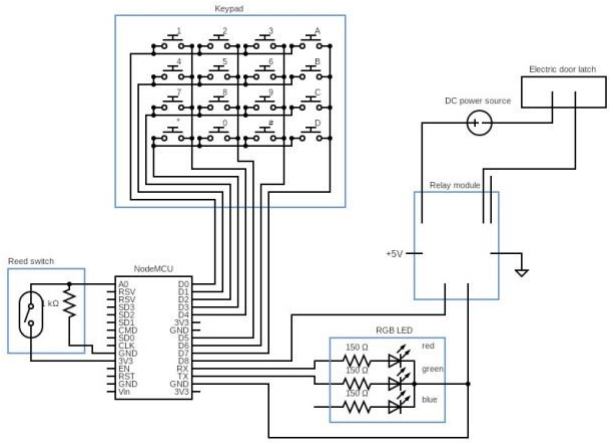

Figure 7. Smart door controller's components and wiring diagram.

Operation and Features

A smart door task loop starts by sensing the buttons pressed on the keypad. To prevent situations where the key is pressed for a large amount of time, leading to invalid readings, the button is only considered pressed after being released. A person has 10s to insert a character after a new one is introduced, otherwise the code is reset, thus preventing codes to be left half written for future situations. The number of failed attempts to introduce a code is stored in memory, thus preventing users from simply reconnecting the device whenever the disabling period is too long. Only a correct code unlocks the door, meaning only a person with it may access the property. Upon detecting the insertion of a correct code, the controller uses an electric signal to trigger the relay for a period of 3s. In case the introduced key fails to match all stored codes, after three attempts, the keypad is disabled for 10s, preventing the situation in which random codes are inserted to unlock the door.

Some of the device's remote properties cannot be accessed when the door operates in offline mode. When connected to a remote unit, the controller requests for the current time, to transfer messages with this information, so that the API is aware of the moment in which an event occurs. This is useful as the device senses the environment perceiving situations where the door was opened without the insertion of one of the three codes. It is also possible to detect situations where a door was left open for more than one minute. Both events are reset whenever the door is closed. In the interest of safety, the notification related to these events are always sent to the server (provided that the notification is activated) even after the door is closed. The moment where these events occurred is stored in memory as a timestamp, ensuring that the messages related to them are sent, even if the device loses power or the connection to the Wi-Fi is lost.

This property may present a security flaw, as it allows the door to be unlocked at will. For security reasons, it is required that a person presses the bell ring button (the keypad's "B" button) to use this functionality. Ten seconds after this button is pressed, it is no longer possible to remotely open the door, thus ensuring the door is only unlocked when someone is at the property's entrance. Figure 8 depicts all possible messages received by the smart door controller which influence the system, as well as those sent after an event has been triggered. Additionally, it provides a description of each of these messages.

Direction	Method	Device	Property	Value	Description
Outgoing	GET	Server	Time	---	Obtain current time from API
	POST	Door	State	Open	Door opened using the primary code
				Open2	Door opened using the secondary code
				Open3	Door opened using tertiary code
				OpenWeb	Door opened by remote server
				Closed	Door closed
			Warning	ForcedOpen	Door Opened without a code
				LongOpen	Door left open for too long
				MaxAttempts	Keypad inserted code failed 5 consecutive times
			Bell	Rang	The door bell button was pressed
Incoming	POST	Door	Bolt	Unlock	Unlocks the door
			PrimaryCode	{value}	Changes the primary code to {value}
			Secondary Code	{value}	Changes the secondary code to {value}
			TertiaryCode	{value}	Changes the tertiary code to {value}

Figure 8. SmartDoor messages and corresponding description.

Home and Cloud Servers

The Homer Server is connected to every entity on the system, managing requests from both users and machines (controllers and cloud server). For this reason, a Raspberry Pi 3B+, shown in Figure 6, has been chosen to host this server, being responsible for manipulating and redirecting messages from

the system units, and allow user to create, update, view and delete system objects.

As it is required that the cloud server allows users to perform the same tasks to each house system, the cloud server must perform the same operations, where each model object is unique for the house they belong to. This application was hosted in a personal computer.

Reservation Management Tasks

The system supports four types of users. *Admin*, a person with permission to navigate through the application without minimum restrictions, thus it has been designed to be attributed to a property owner. *Managers*, the second most privileged account type which may also be used to manage the system, ideal to be assigned to a person contracted to perform house managing tasks. *Client* created to represent the person renting the house and *Staff* designed for people whose job is to assist in maintaining the property, such as cooks, cleaners, gardeners, or electricians, both very limited account types.

The system represents reservations and shift for both *client* and *staff* users, respectively. Both additionally require a house and start date and an end time, storing a date-time value as suggested by their names. Thus, allowing the reservations to be managed by a privileged user.

Home Automation Tasks

The system allows the creation of smart controller and smart device entities, which provides users to insert in the system several smart devices. These devices may contain discrete, continuous and code properties. The first represents categorical properties, such as the State of the door which can be “Open” or “Closed”. The second defines numeric data within an interval, comprised between two values. Measurements, such as the energy consumption sensed, can be introduced in the system using this property. The third, is merely used for home access codes to be managed, so that the system can propagate a notification to all devices containing this property, whenever a reservation starts, or terminates. To perform this task, the system uses a thread which compares the current date and time with reservation’s start or end times.

Additionally, home automation systems are desired to have a functionality that allows an action to be performed upon detecting an event occurrence. This project allows users to create events which detect a change on a smart device’s property, by defining a condition and a threshold. Whenever the server receives a message from a microcontroller, it tests whether the condition is verified. In affirmative situations, the server generates a notification object, which stores the event occurrence in both databases. Additionally, it sends a email to a set of defined users, allowing a manager to monitor the house. House Systems handle a reduced number of users but may contain many smart controllers. Some of these controllers contain smart devices which sense the environment, sending messages to the local server which could then store this value on the system’s database. However, databases write speed is large when compared to their read speed. To prevent bottleneck situations, where devices send data faster than the system can write to its storage, a structure is used to store received values without saving them in the database, as device’s process memory is much faster. Upon receiving a valid value, the server stores it in the cache and uses a task to save these values in the database once every 30s. Data may be lost between database accesses, but the system database is accessed in periodic intervals, reducing its workload. Additionally, if a user is connected to the server, this thread propagates the values sent by a smart controller to the user’s web-browser every 250ms, giving the user the impression values are shared in real time.

EVALUATION

System testing is a very important task that can be used to evaluate the quality of a project, by analysing its features and its limitations on a real-world scenario.

This section refers the tests conducted to evaluate the prototype and the system proposed.

Lock Code Change

It is desired that the system smart lock authorizes a client to enter the property during a limited time period, the client’s reservation. To assess this functionality, a smart door has been added to the system, with a code, accepting 4 digits. A Reservation object has been introduced in the local server, using the access code ”1234”. This reservation was restricted to a short-time period, beginning at 11 hours and 30 minutes of 11/6/2021 and finishing after 5 minutes. This does not happen in real world scenarios, but it prevents long waits.

Additionally, it should be noted that the ESP’s memory did not contain any stored value, meaning It was detected that, at 11h25m of reservation’s day, that the access code ”1234” could not unlock the door. At 11h31, the same code has been introduced in the device’s keypad, activating the relay and displaying a green color on the light emitting diode. Once the reservation time has elapsed, at 11h36, the code was inserted one last time, to wish the microcontroller responded by displaying a red color light. With this test, it is possible to conclude that the main task of this project has been achieved.

Dear user,

Property Door:
Warning State value set to Door Forced Open on 25-Oct-2021, 18:45:13

Best regards.

Figure 9. Emailed alert content.

System Monitoring

The requirements used to design the proposed system take into consideration that the generation of notifications, to alert users when an event occurs in the house environment, has been identified as being important.

To assess verify if the property owner was alerted in these situations, the smart door created has been used to detect the event of a door being forced open. The operation of opening a door has been simulated, using the reed switch, without in introducing the access code. The device detected this change on the environment, thus creating a message to the Home Server. The server reacted to this request, by sending an email to the Administrator account used to test the system, as illustrated by Figure 9.

This test led to the conclusion that the system satisfies the requirement of alerting the user when an event has been detected in the property.

Cache Processing Time

The cache used to store device's sensed data is an important feature of this system, as most databases, have reduced read and write speeds.

To test this, a set of requests have been sent to the server for a total of 500 messages, alternating the sensed value. For this reason, two scenarios have been performed to test the server's message processing time, in seconds, the first containing a single device and the second with 100 of these instances. This time was tested using the cache to identify the device and property to which the sent message was related, saving the value in the same structure ("Cache Read & Write"). These values were compared to another two situations. In the first, the identification was performed querying the database and the cache was used to store the sensed data ("Database Read & Cache Write"), while in the second the database was used for both of these tasks ("Database Read & Write"). Figure 10 contains the summation of the processing time of these 500 messages for the six tests. By analysing these times, it is possible to see that, independently of the number of devices sending messages, the server would always process it in similar times. However, it is clear that the use of a hashing data structure to prevent database accesses significantly reduced the message processing time, due to its O(1) amortized time complexity, caused by the context switch and the storage unit read and write speeds.

N Devices	Cache Read & Write	Database Read & Cache Write	Database Read & Write
1	1,32	4,32	9,11
100	1,52	4,29	9,18

Figure 10. Message processing time comparison, in seconds.

CONCLUSIONS

The administration of short-term accommodation necessitates a substantial time and effort commitment on the part of a person wishing to rent a property. To surpass this difficulty, a home automation system has been proposed to assist a person on man-aging a client's reservation and other house's administration tasks, by reducing some of their work independently of that person being the owner, or a person hired to execute the task. The decision of creating a domotic system, allows smart devices to sense the environment and actuate on it. These devices use the house's local area network to establish a Wi-Fi connection to a server instance, running inside the same property's network, isolating the system from the Internet. The communication channel is provided using WebSockets, allowing a smart device to transfer the sensed information in real-time.

The system data is stored on a database so that data can be persisted in case of power loss.

Additionally, the system provides administrators the possibility to add a cloud hosted server, which communicates, through WebSockets, with the property's central processing unit, providing the administrator immediate remote access to the house's hosted server functionalities. Moreover, many house servers may connect to the cloud hosted instance, allowing a user to manage several house system's, from afar.

A prototype has been developed, in which two smart devices have been created using a NodeMCU, one controlling a person's access through the house's entrance door and another for controlling a light bulb's state. These devices communicated with a server hosted on a Raspberry Pi, collecting data from them and sensing it to a user using the remote server's web application. Furthermore, this last server was developed so that users could send commands through the Raspberry Pi to the ESP nodes, to change an environment's property.

By using the NodeMCU to test whether the code associated with a reservation was accepted, during that reservation's time, revoking afterwards, it was possible to access that this system's main functionality has been correctly implemented.

In sum, it is possible to conclude that the system provides a significant assistance in the management of short-term accommodation businesses.

REFERENCES

1. World Bank, "International tourism, number of arrivals - European Union, Portugal", available online at: https://data.worldbank.org/indicator/ST.INT.ARVL?end=2019&locations=PT-EU&start=2009, last access in September 2021.
2. World Bank, "International tourism, number of arrivals," available online at: https://data.worldbank.org/indicator/ST.INT.ARVL?end=2019&start=2009, last access in September 2021.
3. P. Xiao, Introduction to Arm R©MbedTM. John Wiley & Sons, Ltd, 2018, ch. 1, pp. 1–22, available online at: https://onlinelibrary.wiley.com/doi/abs/10.1002/9781119364009.ch1, last access in September 2021 4. L. Atzori, A. Iera, and G. Morabito, "The Internet of Things: A Survey," Comput. Netw., vol. 54, no. 15, p. 2787–2805, Oct. 2010, available online at: https://doi.org/10.1016/j.comnet.2010.05.010, last access in September 2021.

5. C. Gomez, J. Oller Bosch, and J. Paradells, “Overview and Evaluation of Bluetooth Low Energy: An Emerging Low-Power Wireless Technology,” Sensors (Basel, Switzerland), vol. 12, no. 1, pp. 11 734–53, dec 2012.
6. “Zigbee,” available online at: https://zigbeealliance.org/solution/zigbee, last access in September 2021.
7. “Z-Wave,” available online at: https://www.zwave.com/learn, last access in September 2021.
8. “Discover Wi-Fi,” available online at: https://www.wifi.org/discover-wi-fi, last access in September 2021.
9. Z. Shelby, K. Hartke, and C. Bormann, “The Constrained Application Protocol (CoAP),” RFC 7252, Jun. 2014, available online at: https://rfceditor.org/rfc/rfc7252.txt, last access in September 2021
10. U. Hunkeler, H. L. Truong, and A. Stanford-Clark, “MQTT-S — A publish/subscribe protocol for Wireless Sensor Networks,” in 2008 3rd International Conference on Communication Systems Software and Middleware and Workshops (COMSWARE ’08), 2008, pp. 791–798
11. A. Melnikov and I. Fette, “The WebSocket Protocol,” RFC 6455, Dec. 2011, available online at: https://rfceditor.org/rfc/rfc6455.txt, last access in September 2021.
12. V. Sarafov and J. Seeger, “Comparison of IoT Data Protocol Overhead,” in Proceedings of the Seminars of Future Internet (FI) and Innovative Internet Technologies and Mobile Communication (IITM), 2018, pp. 7–14.
13. Y. T. Park, P. Sthapit, and J.-Y. Pyun, “Smart digital door lock for the home automation,” in TENCON 2009 - 2009 IEEE Region 10 Conference, 2009, pp. 1–6
14. M. Paveli´c, Z. Lonˇcari´c, M. Vukovi´c, and M. Ku ˇsek, “Internet of Things Cyber Security: Smart Door Lock System,” in 2018 International Conference on Smart Systems and Technologies (SST), 2018, pp. 227–232.
15. K. Luechaphonthara and A. Vijayalakshmi, “Iot based application for monitoring electricity power consumption in home appliances,” International Journal of Electrical and Computer Engineering (IJECE), vol. 9, p. 4988, 12 2019.
16. Z. H. Che Soh, M. S. Shafie, M. A. Shafie, S. Noraini Sulaiman, M. N. Ibrahim, and S. Afzal Che Abdullah, “IoT Water Consumption Monitoring Alert System,” in 2018 International Conference on Electrical Engineering and Informatics (ICELTICs), 2018, pp. 168–172.
17. “Nodemcu,” available online at: https://commons.wikimedia.org/wiki/File:Nodemcu amicabot 02.png, last access in October 2021.
18. “Raspberry Pi Model 3B+ (Figure),” available online at: https://www.raspberrypi.com/products/raspberry-pi-3-model-b-plus/, last access in October 2021