Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Dejan Kostadinovski

Prilagajanje ambienta v bivalnih prostorih

magistrsko delo

magistrski program druge stopnje
računalništvo in informatika

Mentor: prof. dr. Viljan Mahnič

Ljubljana, 2018

University of Ljubljana
Faculty of Computer and Information Science

Dejan Kostadinovski

Ambient adjustment in living spaces

MASTER'S THESIS

THE 2nd CYCLE MASTER'S STUDY PROGRAMME
COMPUTER AND INFORMATION SCIENCE

Supervisor: prof. dr. Viljan Mahnič

Ljubljana, 2018

Acknowledgments

I would like to express my gratitude to my thesis supervisor prof. dr. Viljan Mahnič for his patience, advice and guidance through the process of writing the master's thesis.

I would also like to thank my parents, my sister Ivana and my beloved Elena for their invaluable support in my life.

Dejan Kostadinovski, 2018

Contents

List of used acronyms

DBMS	Database Management System
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
IOT	Internet of Things
JSON	JavaScript Object Notation
LED	Light-emitting diode
MQTT	Message Queuing Telemetry Transport
MVC	Model–view–controller
NA	Negative Affect
ORDBMS	Object-Relational Database Management System
ORM	Object-Relational Mapping
PA	Positive Affect
PANAS	Positive and Negative Affect Schedule
REST	Representational State Transfer
SoC	System-on-Chip
SQL	Structured Query Language
UI	User Interface

Povzetek

Naslov: Prilagajanje ambienta v bivalnih prostorih

Različne študije potrjujejo, da primerna osvetlitev, glasba in vonjave pozitivno vplivajo na razpoloženje, koncentracijo in kakovost življenja posameznika. V tej magistrski nalogi smo zasnovali in razvili sistem, ki uporabniku omogoča prilagajanje ambienta v bivalnih prostorih. Ambient se lahko prilagaja preko krmilnega dela sistema, sestavljenega iz čipa, ki ima integrirano povezljivost z brezžičnim omrežjem, LED lučk, zvočnikov in majhnih ventilatorjev, ki širijo prijetne vonjave v bivalnih prostorih. Uporabnik lahko nadzoruje sistem preko spletne aplikacije v realnem času. Izvedli smo tudi anketo, v kateri je vsak udeleženec, ki je testiral sistem, odgovoril na vprašalnik o tem, kako se počuti pred in po uporabi našega sistema. Rezultati so pokazali, da se po uporabi našega sistema pozitivno počutje poveča, negativno pa zmanjša.

Ključne besede

IoT, MQTT, prilagajanje ambienta

Abstract

Title: Ambient adjustment in living spaces

Different studies confirm that a specific adjustment of the lighting, music and scents positively influences the people's mood, concentration and quality of life. In this thesis we designed and developed a system, which allows the user to adjust the ambient in living spaces. Ambient is adjustable through the controllable part of a system consisting of a Wi-Fi enabled chip, LED strip lights, speakers and small fans spreading pleasant scents. The user is able to control this part of the system through a web application in real time. We also conducted a survey in which each participant testing the system answered a questionnaire about how he/she feels before and after using the system. Results showed that after the use of our system positive affect increases, whereas negative affect decreases.

Keywords

IoT, MQTT, ambient adjustment

Razširjeni povzetek

I Uvod in opis problema

Veliko študij raziskuje učinke osvetlitve, glasbe in vonjav na razpoloženje posameznika. Obstoječe študije preučujejo te učinke, medtem ko ljudje kosijo v restavracijah, nakupujejo v nakupovalnih središčih in delajo v svojih delovnih prostorih. Enake koncepte lahko uporabimo za preučevanje vplivov ambienta na razpoloženje ljudi, medtem ko bivajo v svojih domovih in bivalnih prostorih.

Namen magistrske naloge je oblikovati in razviti sistem, ki bo končnemu uporabniku omogočil prilagajanje ambienta v bivalnih prostorih. To je mogoče doseči s prilagoditvijo barve in svetlosti luči, predvajanjem ustrezne glasbe v ozadju in s širjenjem prijetnega vonja v prostoru. Razviti sistem naj bi pozitivno vplival na čustveno stanje in razpoloženje končnega uporabnika.

Za delovanje sistema so potrebne tri glavne komponente. Prva je strojna oprema, sestavljena iz čipa z vgrajenim brezžičnim modulom in naprav, ki so povezane z njim (svetilke LED, zvočniki in ventilatorji). Druga je programska oprema, ki se izvaja na strežniku in uporabniku omogoča, da na daljavo nadzira delo čipa in njegovih naprav. In tretja je ustrezen komunikacijski protokol za povezovanje prvih dveh komponent. Uporabnik bo sistem nadziral preko spletne aplikacije v realnem času. Storitev, ki izvaja spletno aplikacijo, bo posredovala uporabniške ukaze čipu, ki bo obdelal ukaze in skladno z njimi upravljal priključene naprave.

Predlagani sistem bo sestavni del širšega sistema za upravljanje in nadzor pametnega doma. Sistem bo vgrajen v pohištvo (npr. postelja, garderobna omara, pisalna miza itd.), njegovo upravljanje pa bo potekalo preko osebnega računalnika.

II Kratek pregled sorodnih del

Osnovo za naše delo predstavljata dve vrsti študij. V prvo vrsto spadajo študije, ki preučujejo vpliv osvetlitve, glasbe in vonjav na počutje posameznika.

V študiji [1] so raziskali vpliv svetlobe na kognitivno uspešnost z dvema svetlobnima poskusoma. Rezultati obeh poskusov so pri vsakem udeležencu pokazali, da jakost svetlobe in temperatura barv vplivata na izboljšanje dolgoročnega spomina ter boljše prepoznavanje in reševanje problemov.

Avtorji [2] so izvedli raziskavo za merjenje stresa med udeleženci. Anketa je pokazala, da je glasba, ki jo je izbral udeleženec, najučinkovitejša za kratkoročno izboljšanje razpoloženja. Prav tako so poudarili, da poslušanje glasbe lahko izboljša kakovost življenja in spanja, sposobnost in koncentracijo posameznika ter krepi družbeno povezanost.

V študiji [3] so avtorji analizirali učinek ambientalnega vonja pri odločanju. Poskus je pokazal, da uporaba dišav na dražbah, zlasti v primerih, ko prevladuje vonj mete, zmanjša občutek tveganja, zato ljudje ponavadi ponudijo višje cene.

Drugo vrsto predstavljajo študije, ki preučujejo arhitekturo in implementacijo sistemov za spremljanje in nadzor IoT naprav.

Avtorji študije [4] so predlagali arhitekturo za spremljanje in nadzor IoT naprav, sestavljeno iz treh glavnih plasti: omrežja brezžičnih naprav, prehoda (angl. gateway) in oblačne storitve. Prehod pretvori sporočila, ki prihajajo iz brezžičnega omrežja, v oblačni protokol. Transformirana sporočila se obdelajo v oblačni storitvi, ki preveri stanje sistema in pošlje ustrezne ukaze na naprave v nadzorovanem omrežju.

V prispevku [5] je predstavljena praktična implementacija IoT prehodov, ki je namenjena daljinskem nadzoru bazena. Avtorji prispevka so ustvarili prehod, ki lahko uporablja več komunikacijskih protokolov ter žičnih in brezžičnih senzorjev. Prehod temelji na računalniku Raspberry Pi, ki omogoča dvosmerno komunikacijo in izmenjavo podatkov med uporabnikom in senzorji v omrežju. Sistem zagotavlja popolnoma oddaljen in varen nadzor senzorjev preko spletne applikacije.

III Arhitektura sistema

Glavni del sistema, ki smo ga razvili v okviru magistrske naloge, je cenovno ugoden čip ESP8266 z 32-bitni sistemom in brezžično povezljivostjo. Za prilagajanje ambienta v bivalnem prostoru so na čip povezane svetilke LED, zvočniki in ventilatorji. Upravljanje sistema preko oddaljenega dostopa poteka s pomočjo spletne aplikacije, ki temelji na osnovi HTML5. Uporabljen je protokol MQTT [6], ki zagotavlja, da posamezne aplikacije in sistemi brezhibno komunicirajo z relativno visoko stopnjo varnosti z uporabo robustnega varnostnega protokola spletne storitve.

Celotno arhitekturo, predlagano v tej magistrski nalogi, sestavljajo naslednje komponente:

1. **Podatkovni strežnik**, ki je odgovoren za shranjevanje, dostopanje in upravljanje podatkov v elektronski obliki.

2. **Varnostna storitev** zagotavlja varen dostop do uporabniškega vmesnika spletnih aplikacij in posledično varen nadzor nad napravo ESP8266.

3. **Spletni strežnik** upravlja zahteve, ki prihajajo od končnega uporabnika, izvaja različne ukaze in logiko za pravilno delovanje celotnega sistema in posreduje komunikacijo med končnim uporabnikom, strežnikom MQTT in napravo ESP8266.

4. **Strežnik MQTT** je komunikacijski most med napravo ESP8266 in spletnim strežnikom, ki omogoča uporabo protokola MQTT.

5. **Spletna aplikacija** zagotavlja uporabniški vmesnik, preko katerega končni uporabnik upravlja z oddaljeno napravo ESP8266.

6. **Naprava ESP8266** je čip, ki vsebuje 32-bitni sistem. Čip sprejema in izvršuje ukaze, ki jih zahteva uporabnik.

7. **Svetilke LED, zvočnik in majhen ventilator** so električne naprave, ki so vezane na čip ESP8266 in poskrbijo za nastavljanje želenega ambienta v prostoru.

V primerjavi s študijo [5] ima naš sistem enostavnejšo arhitekturo, saj ne potrebuje vmesnega agregacijskega vozlišča med oddaljenim strežnikom in nadzorovano enoto. Na ta način smo zmanjšali kompleksnost namestitve sistema in njegovo ceno.

IV Eksperimentalna evaluacija

Za ovrednotenje rešitve smo izvedli poskus, ki je raziskoval korelacijo med čustvenim stanjem udeležencev in uporabo sistema. Sistem, ki smo ga razvili, smo dali za 4 dni v uporabo desetim različnim uporabnikom in izvedli anketo o njihovem počutju pred in po uporabi sistema. Za pridobivanje podatov o čustvenem stanju vsakega udeleženca smo uporabili vprašalnik PANAS [7], ki vsebuje 10 vprašanj za pozitivna počutja in 10 vprašanj za negativna. Vsak udeleženec je vprašalnik izpolnil dvakrat: enkrat pred in enkrat po uporabi našega sistema. Z uporabo t-testa za odvisna vzorca smo pokazali, da obstaja pomembna statistična razlika v pozitivnih in negativnih čustvih uporabnika pred in po uporabi sistema. Moč pozitivnih čustev se je po uporabi sistema še povečala, moč negativnih pa zmanjšala.

V Sklep

V okviru te magistrske naloge smo načrtali in razvili sistem, ki omogoča končnemu uporabniku prilagajanje ambienta v bivalnih prostorih. Združili smo vse ambientalne dražljaje, obravnavane v Poglavju II, v en sistem in končnemu uporabniku omogočili, da v bivalnih prostorih uspostavi sebi prilagojeno okolje z ustrezno glasbo, osvetlitvijo in prijetnimi vonjavami.

Sistem je sestavljen iz čipa, svetilke LED, zvočnikov in majhnih ventilatorjev. S pomočjo spletne aplikacije uporabnik lahko prilagaja ambient in upravlja z celotnim sistemom. Rezultati rešitve so pokazali, da se po uporabi našega sistema pozitivna čustva uporabnika povečajo, negativna pa zmanjšajo.

V prihodnosti načrtujemo integracijo našega sistema kot sestavni del širšega sistema za nadzor in upravljanje pametnega doma. S pomočjo uporabe dodatnih senzorjev in algoritmov strojnega učenja želimo tudi avtomatizirati nekatere funkcije sistema, kot na primer vklop samodejnega predvajanja pomirjujoče glasbe in primernih osvetlitvenih vzorcev, ko gre uporabnik v posteljo.

Chapter 1

Introduction

1.1 Motivation

There are many studies [1, 2, 3] exploring the effects of lights, music and scent on people's mood. Results of these studies confirm that a specific adjustment of the ambience positively influences the people's mood, can enhance the person's well-being and quality of life and can improve the sleep quality and concentration. Existing studies explore these effects while people are dining in restaurants, shopping in shopping malls and working in their work offices.

We can use the same concepts to explore the effects of the ambient on the people's mood while they are staying in their own homes and living spaces. A possible solution is to integrate them into the concept of a smart home, where a smart home is defined as a residence equipped with technology that provides proactive services to the residents [8, 9]. The idea is to design and develop a system, which will allow the end user to adjust the ambient in the living spaces. This is achievable by adjusting the light color and brightness of the lights, playing appropriate music in the background and spreading a pleasant scent in the space. A developed system should positively influence the emotional state and the mood of the end user.

1.2 Proposed solution

Within the scope of this master's thesis, we will design and develop a remotely controllable system, which will be an integral part of a wider management system for controlling a smart home. The controllable part of the system will be built into the usual furniture (eg. bed, wardrobe, office desk, etc.) and will consist of a Wi-Fi enabled chip, LED strip lights, speakers and small fans. In that way we will open a new perspective of combining technology and furniture.

The main functionality of the system will be enabling the user to adjust the ambient in the living space. A user will be able to play audio files and stream live radio stations, create and play custom patterns of changing RGB colors on the LED strip lights and start fans spreading pleasant scents in the living space. For example, the user can simultaneously project a wood burning pattern on the LED strip lights, play wood crackling sounds in the background and spread a scent of burning wood in the space.

For the system to work, three major components are required. The first is a hardware consisted of a Wi-Fi enabled chip and actuators connected to it (LED strip lights, speakers and fans). The second is a service running on a server, which will allow the user remotely to control the Wi-Fi enabled chip. And the third is an appropriate communication protocol for connecting the first two components. The user will control the system through a web application in real-time. The service running the web application will forward the user's commands to the Wi-Fi enabled chip which will process the commands and start controlling the connected actuators.

1.3 Structure

In Chapter 2 we give an overview of the problem domain, study the related work on smart homes and effects of music, lights and scents on people's mood, and study different designs and architectures for monitoring and controlling IoT (Internet of Things) devices. In Chapter 3 we describe the overall architecture of our system developed as a result of this thesis. In details we explain every component used in the system and communication protocol used for intermediate communication. We also describe the storing and relationships of the data required for working of the system. Chapter 4 discusses the implementation of the system's components and the communication flow between each of them.

The analysis and results of this work are evaluated in Chapter 5 while Chapter 6 presents our findings and conclusions, the work's contributions and proposals for further work and improvements.

Chapter 2

Related work

2.1 Smart homes and taxonomy

Article [9] develops concepts of what the home is and reflects on smart home technology in relation to these concepts. On the aspects of smart home technologies, four aspects of a home are distinguished: a place for *security and control*, for *activity*, for *relationships and continuity*, and for *identity and values*. Authors explain that there is no fixed definition of a smart home, but an understanding that smart homes incorporate digital sensing and communication devices. These devices communicate with each other seamlessly in the smart home ideal, in order to provide one or more of the following services: more sophisticated control of energy, greater security against break-ins, innovations in home entertainment and ambience, health monitoring and independent/assisted living arrangements. The paper concludes that a broader understanding of the home in all aspects is needed when conducting research in smart homes.

Authors of [10] conducted a focused search for every article related to *smart homes*, *apps*, and *IoT* in three major databases (Web of Science, ScienceDirect, and IEEE Explore), to map the research landscape of smart home applications based on Internet of Things into a coherent taxonomy. They divided articles into four classes. The first class comprises review and survey articles related to smart home IoT applications. The second class includes papers on IoT applications and their use in smart home technology. The third class contains proposals of frameworks to develop and operate applications. The final class includes studies with actual attempts to develop smart home IoT applications. Authors then identified the basic characteristics of this emerging field in the following aspects: motivation for the use of IoT in smart home applications, open challenges hindering utilization, and recommendations to improve the acceptance and use of smart home applications in literature.

2.2 Effects of lighting, music and environmental scents on person's mood

The general aim of study [1] is to investigate the hypothesized line of affect from luminous milieu on cognitive performance via mood, by two light exposure experiments. Results of both experiments have shown enhanced performance in the long-term recall, recognition and problem-solving tasks, in each participant respectively. The experimental study [11] showed that there are systematic influences of lighting on mood from lighting parameters within the range of those encountered in everyday interior conditions.

The authors of another study [2] conducted a survey to measure the current life stress and psychological distress among the participants. The survey showed that music chosen by the listener was most effective for short-term mood enhancement and that music-listening has great potential to rehabilitate mood, and thus enhance well-being and quality of life. The study states that not only can music invoke a large spectrum of emotions, but it can enhance executive skills and concentration, improve sleep quality, and strengthen social connectedness.

The authors of [12] showed that environmental scents have long been produced and widely used in business outlets to make hotel rooms or stores have a pleasant scent. With such environmental stimuli, the retailers expect to improve the customers' mood and potentially increase their loyalty. In another study [3], the authors conducted an experiment and analyzed the effect of ambient scent type and intensiveness on decision making. The experiment has shown that making judgments under scented conditions, especially in case when peppermint scent is prevalent, the feeling of risk reduces, and people tend to bid higher. The results of [3] are in line with the reported characteristics of peppermint scent, since it increases alertness, captures attention, and speeds up physiological processes [13].

2.3 Monitoring and controlling IoT devices

The authors of [4] have proposed an architecture for monitoring and controlling IoT devices composed of three main layers: wireless device network, gateway and Cloud service. The wireless device network corresponds to the set of sensors and actuators attached to the system to be monitored and controlled. The gateway converts messages coming from the wireless network to the Cloud protocol. The transformed messages are consumed by the Cloud Service, which in turn, after evaluating the overall system's conditions, can send back commands to the actuators on the controlled network [4].

The paper [5] presents a practical implementation of an IoT gateway dedicated to real-time monitoring and remote control of a swimming pool. The authors of the paper created a gateway which is capable of using multiple communication protocols and both wired and wireless sensors. The gateway is based on Raspberry Pi single-board computer. It allows bidirectional communication and data exchange between the user and the sensor network implemented on the environment using an Arduino micro-controller. The system provides full remote and secure control and monitoring of sensor networks, via an online platform. Figure 2.1 presents the high level system architecture developed by the authors of the paper. As shown and discussed in [5], the system consists of a hardware, a software and implemented communication features.

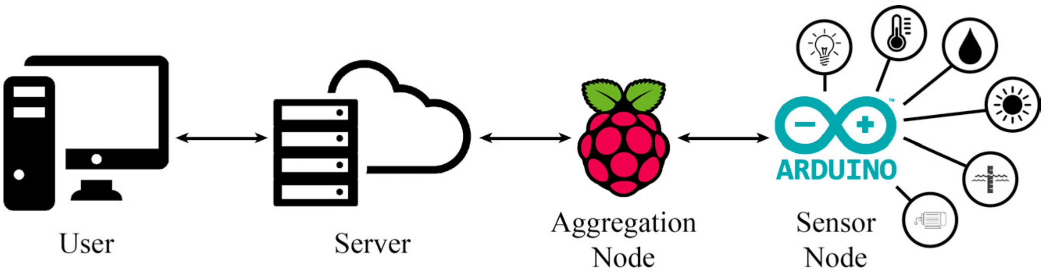

Figure 2.1: System architecture of the work presented in study [5].

Chapter 3

System architecture

In this chapter we will explain the architecture of our system, which follows the architectures of the systems applying the IoT technology discussed in Chapter 2.3. Our system architecture applies simplified design protocols for developing a robust home ambient adjustment system to deal with the problems of complexity, multiple incompatible standards and the resulting expenses in the existing systems. The embedded system features the ubiquitous low-cost 32-bit ESP8266 System-on-chip (SoC) module interfaced to some actuators for interaction in the living space. Flexibility in the remote access, operation and management is achieved through HTML5 based intuitive web GUI application. Message Queuing Telemetry Transport (MQTT) protocol is deployed to ensure that individual applications and systems seamlessly communicate with a relatively high level of security using robust web service security protocol.

The overall architecture proposed in this master's thesis is depicted by Figure 3.1 and is composed of:

1. **Database Server**, responsible for storing, accessing and managing data electronically.

2. **Web Service Security** secures the access to the Web application UI and consequently secures the controlling of the ESP8266 device and its actuators.

3. **Web Server** manages the requests coming from the end user, executes various commands and logic for proper working of the whole system and intermediates commands between the end user, MQTT server and ESP8266 device.

4. **MQTT Server** is the communication bridge between the ESP8266 device and the web server.

5. **Web application** is responsible for delivering a user interface to the end user which, by clicking, fires actions on the remote device and actuators.

6. **ESP8266** is a system-on-chip device, which receives and executes commands requested from the user. It is a connection point between the actuators.

7. **Controllable actuators** are actuators connected to the ESP8266 device (LED strip lights, speaker and small fan) which perform the commands requested from the user.

3.1 Database Server

A database is an organized collection of data, stored and accessed electronically. The database management system (DBMS) is the software that interacts with end users, applications, and the database itself to capture and analyze data. A general-purpose DBMS allows the definition, creation, querying, update, and administration of databases [15]. A database server is a server which houses a database application that provides database services to other computer programs or to computers [16].

In our system we use PostgreSQL, often simply Postgres, which is an object-relational database management system (ORDBMS). We also use Hibernate ORM (Object-relational mapping) which is a framework for mapping a Java object-oriented domain model to a relational database.

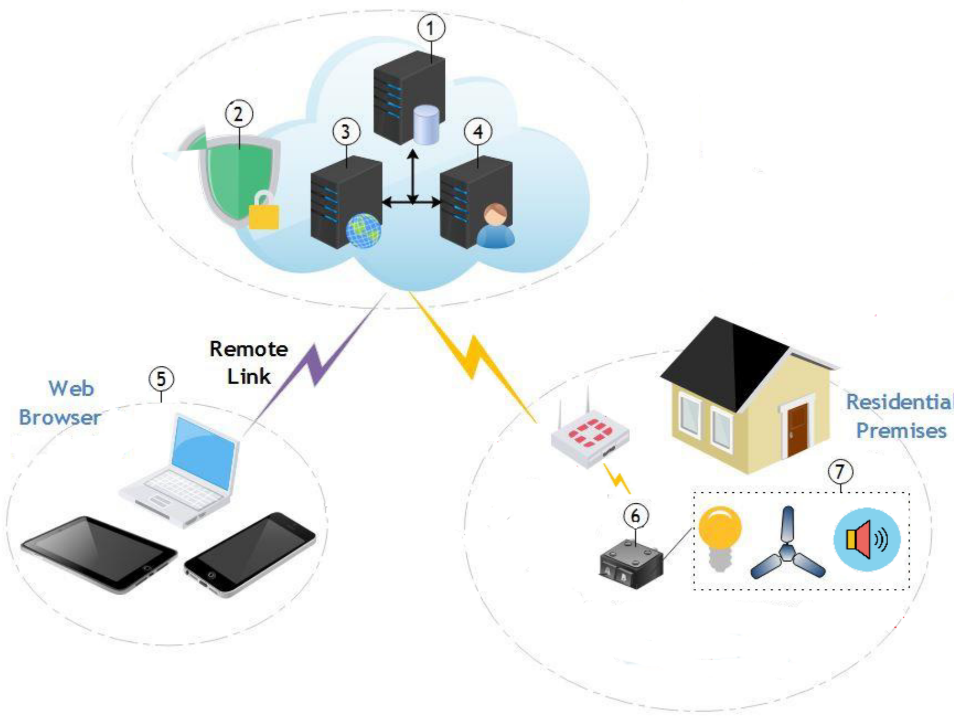

Figure 3.1: System architecture *(figure is acquired from [14] and adapted to map the design of our system architecture)*

3.1.1 PostgreSQL

PostgreSQL is a powerful, open source object-relational database system that uses and extends the SQL (Structured Query) language combined with many features that safely store and scale the most complicated data workloads.

PostgreSQL has earned a strong reputation for its proven architecture, reliability, data integrity, robust feature set, extensibility, and the dedication of the open source community behind the software to consistently deliver performance and innovative solutions. PostgreSQL runs on all major operating systems, has been ACID-compliant since 2001, and has become the open source relational database of choice for many people and organizations [17].

3.1.2 Hibernate ORM

Hibernate ORM (Hibernate in short) is an object-relational mapping tool for the Java programming language. It is a free software that provides a framework for mapping an object-oriented domain model to a relational database [18]. Hibernate's primary feature is mapping from Java classes to database tables, and mapping from Java data types to SQL data types. Hibernate also provides data query and retrieval facilities. It generates SQL calls and relieves the developer from the manual handling and object conversion of the result set.

3.1.3 Database schema

As mentioned in Section 3.1 we use a PostgreSQL database which consists of 9 data tables. The database schema and entity relationship are depicted in Figure 3.2.

Hereinafter, we will shortly explain each data table in the database schema.

- `DEVICE` data table stores data about ESP8266 devices. Each device is unique by its `mac_address` field. This data table is in many-to-one relationship with the `USER` data table, meaning that one end user has the opportunity to control multiple ESP8266 devices.

- DEVICE_PAIRING_CODE is a data table containing pairs of the mac_address and pairing_code of a device. This table is populated by the administrator of the system. Missing entry of mac_address - pairing_code pair prevents adding a device to the DEVICE data table.

- RESOURCE data table stores meta data about audio files uploaded by the end user to the web server. For each uploaded audio file, system generates unique string uuid and saves its file path to the database.

- RADIO_STATION contains names and URLs of radio stations manually added by the end user.

- SCENT data table stores information about scent installed on each particular fan. It is in many-to-one relationship with the device data table, meaning that one device can have multiple fans connected to it.

- LIGHT data table contains information about the pattern of RGB colors showed on LED strip lights. Field duration indicates the time (in milliseconds) showing a particular pattern on the lights strip. Field data is an array of integer values representing the RGB color of each LED light on the strip.

- LIGHT_TEMPLATE data table is in a one-to-many relationship with the LIGHT data table meaning that one template is composed of multiple lighting patterns.

- USER data table stores information about the end users. Each user is in one-to-one relationship with an entity in the ROLE data table, which means that a user has only one role.

- ROLE data table contains data of roles with which the end users access the system. Our system contains only two roles: ADMINISTRATOR and USER.

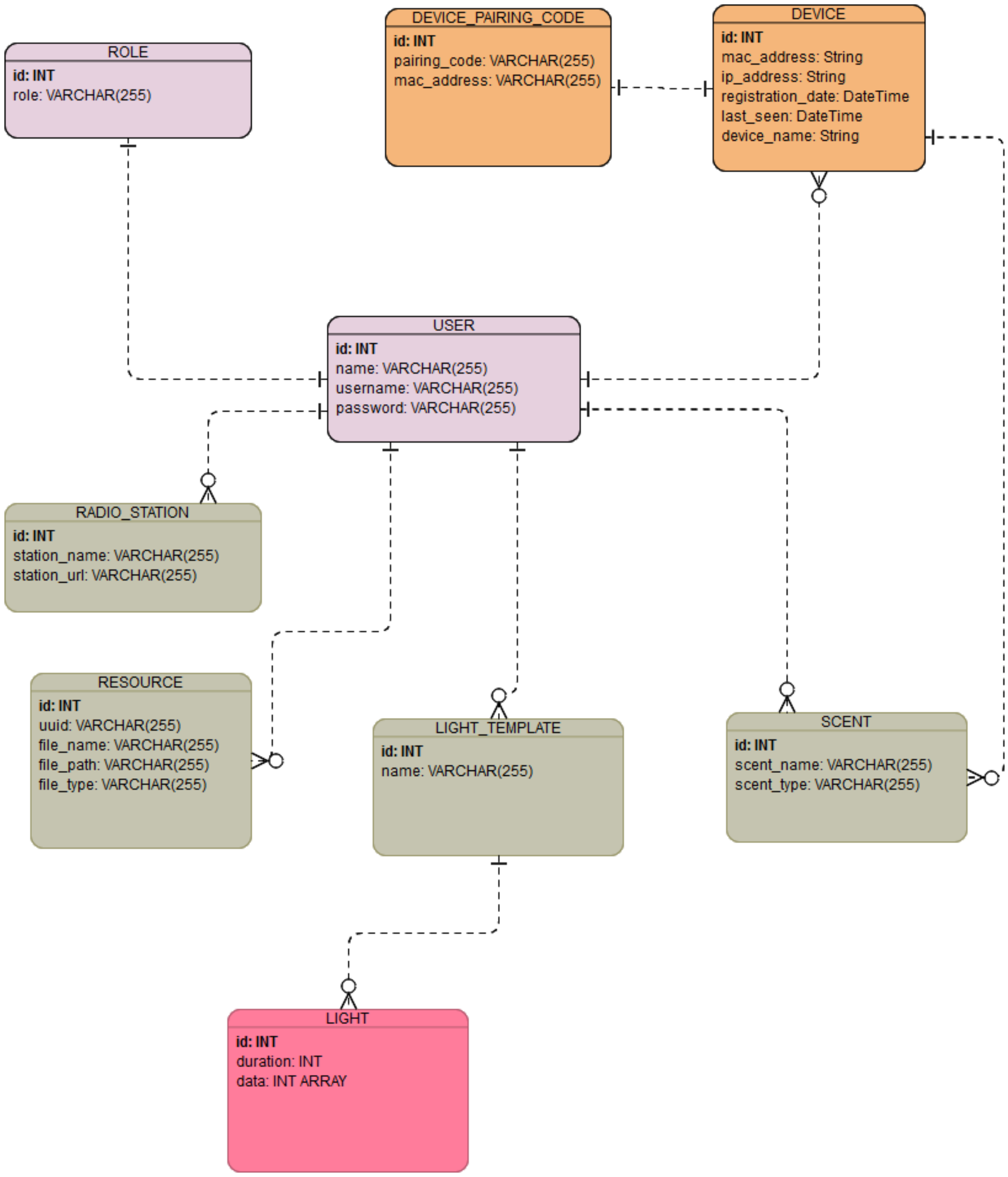

Figure 3.2: Database schema

3.2 Web Service Security

For securing our web application and REST API endpoints we use the Spring Boot Security 2.0 dependency [19], which is a part of Spring Boot framework (which will be discussed in Chapter 3.3). Using a Spring Boot Security is an easy way to secure an application requiring only two tables (`USER` and `ROLE` explained in Chapter 3.1.3) for auto-configuration of the web security.

3.3 Web server

A Web server is a program that uses HTTP (Hypertext Transfer Protocol) to serve the files that form Web pages to users, in response to their requests, which are forwarded by their computers' HTTP clients [20]. Our system's web application and REST API endpoints run on Apache Tomcat server [21] which can be easily included in the Spring Boot framework [22] and can be run directly. As suggested in article [23], we choose to build our web application using Spring Boot framework because it has several advantages. Spring has supported MVC (Model–View–Controller architectural pattern) and provides a RESTful Web Service feature. A database connection is also provided in the Spring package. Spring Boot framework also supports a dependency injection. Dependency injection is the process of supplying a resource that a given piece of code requires [24].

3.4 MQTT Server

3.4.1 Communications protocols

As mentioned in the beginning of this chapter, communication is one of the main elements of IoT. In the system there are two key communication points, between the web server and the web browser application and from the MQTT broker to the ESP8266 device. In the first, an open-source, secure and good performance application layer Internet protocol is required, that can also connect a HTML5/JavaScript Client to a Java EE servlet. Therefore, the communications are done using HTTPS protocol. In the second, we searched for a low-cost, low-power, high range, multi-node communication protocol with low hardware complexity and wireless capabilities. We chose Message Queuing Telemetry Transport (MQTT) protocol [6], a protocol designed for IoT projects.

3.4.2 MQTT protocol

MQTT stands for Message Queuing Telemetry Transport. It is a publish/-subscribe, extremely simple and lightweight messaging protocol, designed for constrained devices and low-bandwidth, high-latency or unreliable networks. The design principles are to minimize network bandwidth and device resource requirements whilst also attempting to ensure reliability and some degree of assurance of delivery. These principles also turn out to make the protocol ideal for the emerging "machine-to-machine" (M2M) or "Internet of Things" world of connected devices, and for mobile applications where bandwidth and battery power are at a premium.

A MQTT system consists of clients communicating with a server, often called a "broker". A client may be either a publisher of information or a subscriber. Each client can connect to the broker [25].

Information is organized in a hierarchy of topics. When a publisher has a new item of data to distribute, it sends a control message with the data to the connected broker. The broker then distributes the information to clients that have subscribed to that topic. The publisher does not need to have any data on the number or locations of subscribers, and subscribers in turn do not have to be configured with any data about the publishers [26].

If a broker receives a topic for which there are no current subscribers, it will discard the topic unless the publisher indicates that the topic is to be retained. This allows new subscribers to a topic to receive the most current value rather than waiting for the next update from a publisher.

When a publishing client first connects to the broker, it can set up a default message to be sent to subscribers if the broker detects that the publishing client has unexpectedly disconnected from the broker.

Clients only interact with a broker, but a system may contain several broker servers that exchange data based on their current subscribers' topics.

A minimal MQTT control message can be as little as two bytes of data. A control message can carry nearly 256 megabytes of data if needed. There are fourteen defined message types used to connect and disconnect a client from a broker, to publish data, to acknowledge receipt of data, and to supervise the connection between client and server.

MQTT relies on the TCP protocol for data transmission. A variant, MQTT-SN, is used over other transports such as Bluetooth.

MQTT sends connection credentials in plain text format and does not include any measures for security or authentication. This can be provided by the underlying TCP transport using measures to protect the integrity of transferred information from interception or duplication.

3.5 Web Application

In order the end user to control the ESP8266 device and its actuators, we developed a Web application user interface (UI) which is shown in Figure 3.3. We built web application using Java EE, Thymeleaf [27] and JavaScript language.

UI consist of multiple tabs:

- **Home** tab consists of multiple fragments.
  - **Device** fragment allows the user to select the device he wants to control.
  - **Audio** fragment allows the user to play his favourite audio songs or radio stations on the selected device.
  - **Light** fragment allows the user to show different lighting patterns on the LED strip lights connected to the selected device.

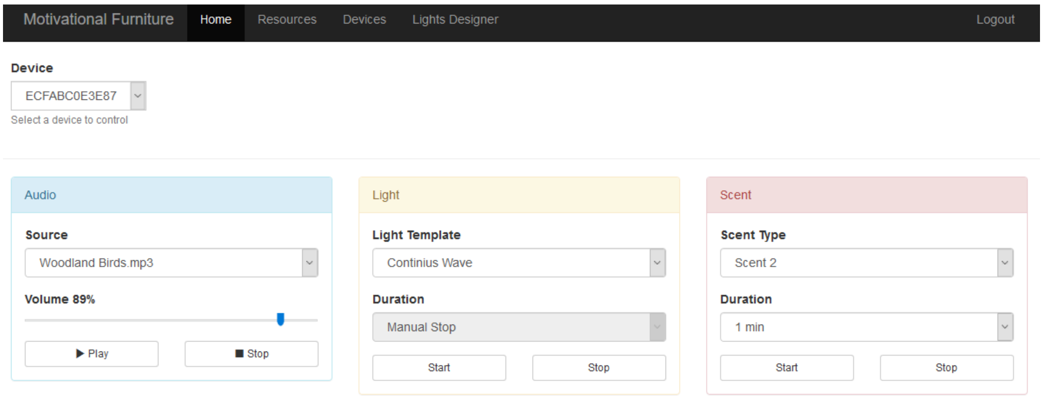

Figure 3.3: Web application user interface for controlling the ESP8266 device and its actuators.

  - **Scent** fragment allows the user to start fan spreading a pleasant scent in the space.

- **Resources** tab allows the user to upload audio songs and add internet radio stations.
- **Devices** tab allows the user to pair an ESP8266 device. The pairing process between the user and the device is explained in Chapter 4.1.
- **Lights Designer** tab allows the user to create his own lighting patterns.

3.6 ESP8266 device

The device showed in Figure 3.4 is a ubiquitous low cost and low power 32-bit CPU based ESP8266 Wi-Fi module. It is a self-contained SoC (System on a Chip) with integrated Transmission Control Protocol/Internet Protocol (TCP/IP) stack used as an application processor [28]. We chose this module because of its powerful on-board processing, storage capabilities, minimal development upfront and minimal loading during runtime. The module

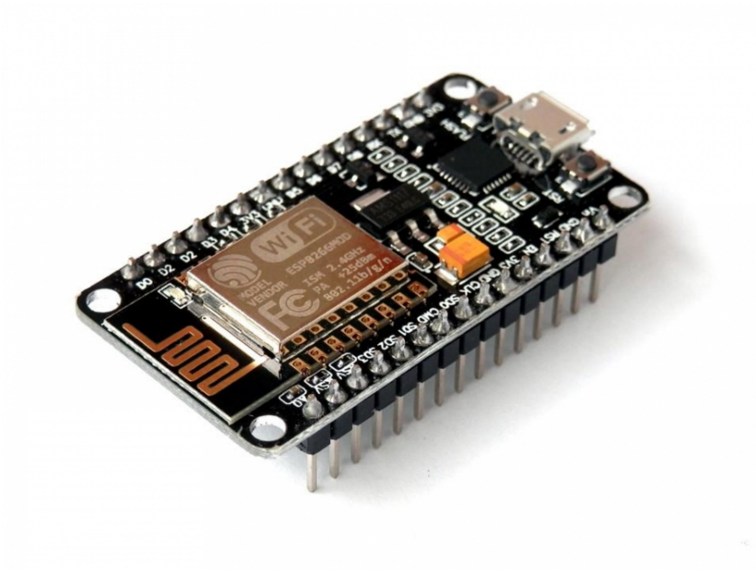

Figure 3.4: ESP8266 SoC [29]

also allows interfacing with the sensors and other application specific devices through its General-purpose input/output (GPIOs). Its high degree of on-chip integration allows for minimal external circuitry while it also contains a self-calibrated RF allowing it to work under all operating conditions, and requires no external RF parts. It has features such as 802.11 b/g/n, Wi-Fi direct (P2P), soft-AP, 81 Mb RAM, up to 160 MHz speed, 1 Mb flash memory and+19.5 dBm output power [14].

3.7 Controllable actuators

To the ESP8266 device discussed in Chapter 3.6 and used in our system, we connected different actuators: LED strip lights, a speaker and a small fan.

LED strips lights are addressable and color-changing and include a tiny chip in between each and every LED, allowing us to control them all individually. In Chapter 4.4.2, we provide an example of the firmware running the color-changing command of the LED lights.

To the audio output of the ESP8266 chip we connected a small speaker which is easily replaceable by any other speaker an end user desires.

Between the ESP8266 device and the fan there is a relay, which is an electrically operated switch [30], and in our system used for controlling the working of the fans. When the ESP8266 device receives a command for turning on/off the fan, it signals the relay to start/stop the fan.

Chapter 4

Implementation

The system we developed is depicted in Figure 4.1. It consists of an ESP8266 device and a speaker together wrapped in a box, LED strip lights and a relay connected to the device and a fan connected to the relay. In order the fan to spread a pleasant scent in the space, we connected the air output of the fan to an essential oil tube.

In the following sections of this chapter we will discuss the implementation of the system's components and the way they communicate between each other.

4.1 Registration and pairing of the ESP8266 device

In order an ESP8266 device to become controllable by the user, the device should be firstly registered by the administrator of the system and then paired by the user.

Figure 4.2 illustrates the registration and pairing flow of the ESP8266 device, while Figure 4.3 shows the UI (user interface) of the web pairing form used by the user.

Registration and pairing of the ESP8266 device consist of the following steps (Figure 4.2):

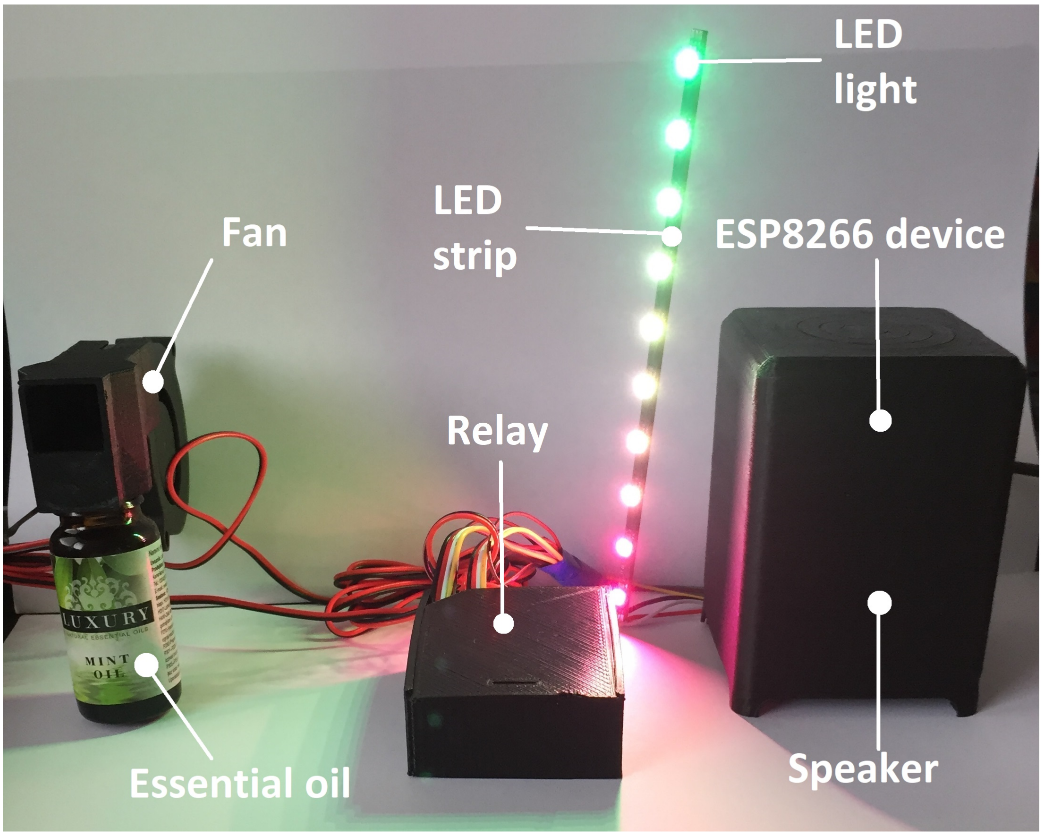

Figure 4.1: Controllable part of the system consisted of: LED light, LED strip, ESP8266 device, Speaker, Relay, Essential oil and Fan. Every component is wrapped in a plastic box.

1. Registration

    (a) Administrator of the system sends a POST request to the server's API requesting a device registration. The body of the POST request contains a JSON formatted string filled with MAC address of the registering device and a pairing code.

    (b) Server API receives the request, deserializes it and submits an insert transaction to the database.

    (c) Database receives the insert transaction. If the entry is not present in the database table, database performs an insert query and returns a success code to the server's API.

    (d) If the insert query is successful, the server API returns a HTTP 201 Created status code to the system's administrator.

2. Pairing

    (a) The ESP8266 device connects to the internet.

    (b) The ESP8266 device publishes an "alive" message to the MQTT broker.

    (c) The Server API saves the device's MAC address in the database, but does not pair it with any end user.

    (d) The end user through the web application form shown in Figure 4.3, sends a pair request to the server's API. The pair request sent contains a JSON string with the pairing device's MAC address and a pairing code.

    (e) Server receives the request and pair the device with the end user.

    (f) Device is now paired with the end user and subscribed to the MQTT broker.

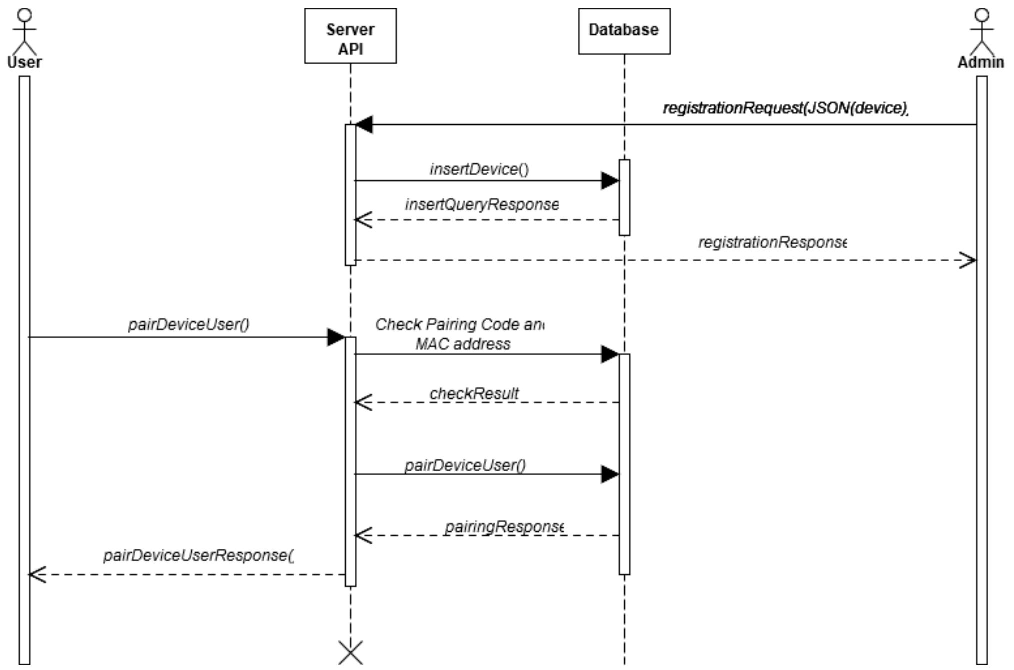

Figure 4.2: Registration and pairing of a ESP 8266 device

4.2 ESP8266 device receiving and executing commands

User adjusts the ambient through sending commands to the ESP8266 device, which receives the commands and starts controlling the actuators connected to it. Receiving and executing commands on the device are sequential operations, which means one cannot run if the other is still processing. To prevent deadlocks, we define a timeout (time in which one operation should finish) for each executing command. If the command fails to finish within the predefined timeout, the execution of the command is terminated and the command is returned in the command queue or it is removed. Only the commands for playing audio can be returned to the command queue (eg. when streaming audio from URL). The other commands are removed and the user is notified about the exception occurred.

Figure 4.4 shows the flow of receiving and executing commands on the

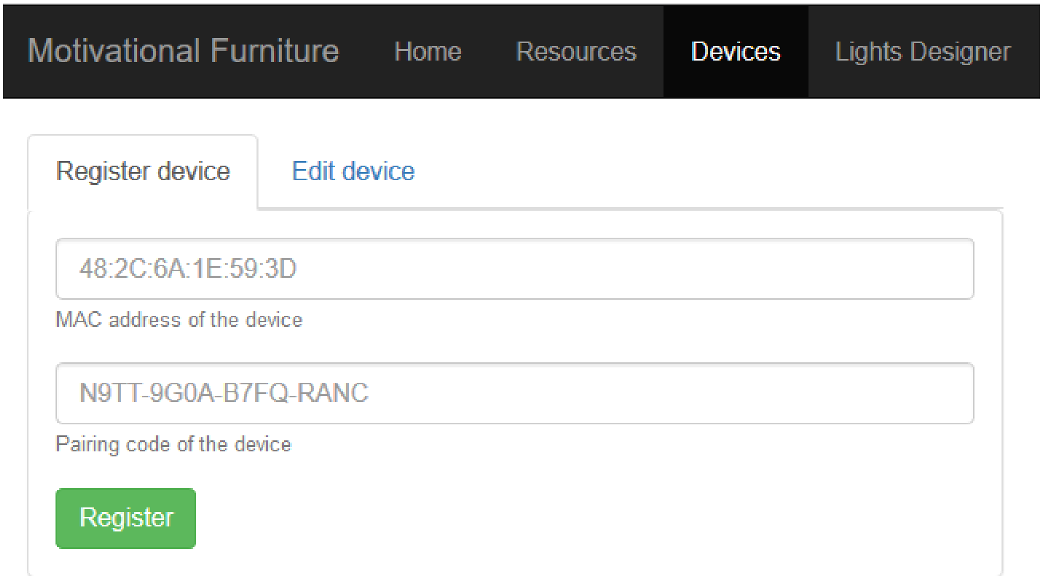

Figure 4.3: The device pairing form. Each ESP8266 device used in our system contains information about its unique MAC address and pairing code. The user enters the MAC address of the pairing device and its pairing code and clicks button *Pair*. After the successful pairing action, the user will be able to control the paired device.

ESP8266 device.

1. When device boots up it firstly tries to connect to a WiFi network.
2. After connecting to the WiFi network it checks if there is any data available in the InputStream. If yes, device executes step 3, otherwise step 5.
3. Device tries to read the data and convert it into an executable command.
4. Device adds the converted command to a command queue.
5. Device checks if there is any available command in the queue. If no command is available, executing flow goes back to step 2, otherwise to step 6.
6. Flow checks if the command is an instance of the command responsible for controlling the fans or the lights.
7. If the command is a fan-controlling command, device turns on/off the particular fan and returns to step 2.
8. If the command is a lights-controlling command, device turns on/off a sequence of LED lights and returns to step 2.
9. If the command is neither a fan-controlling nor a lights-controlling command, the device starts executing the command for playing audio.
10. When device executes a playing audio command, it firstly reads data from the InputStream and writes that data to a buffer.
11. Device reads data from the buffer and writes that data to the AudioOutput port of the device.
12. If the timeout for executing a single command is up, the flow returns the current command to the command queue. Otherwise it returns to

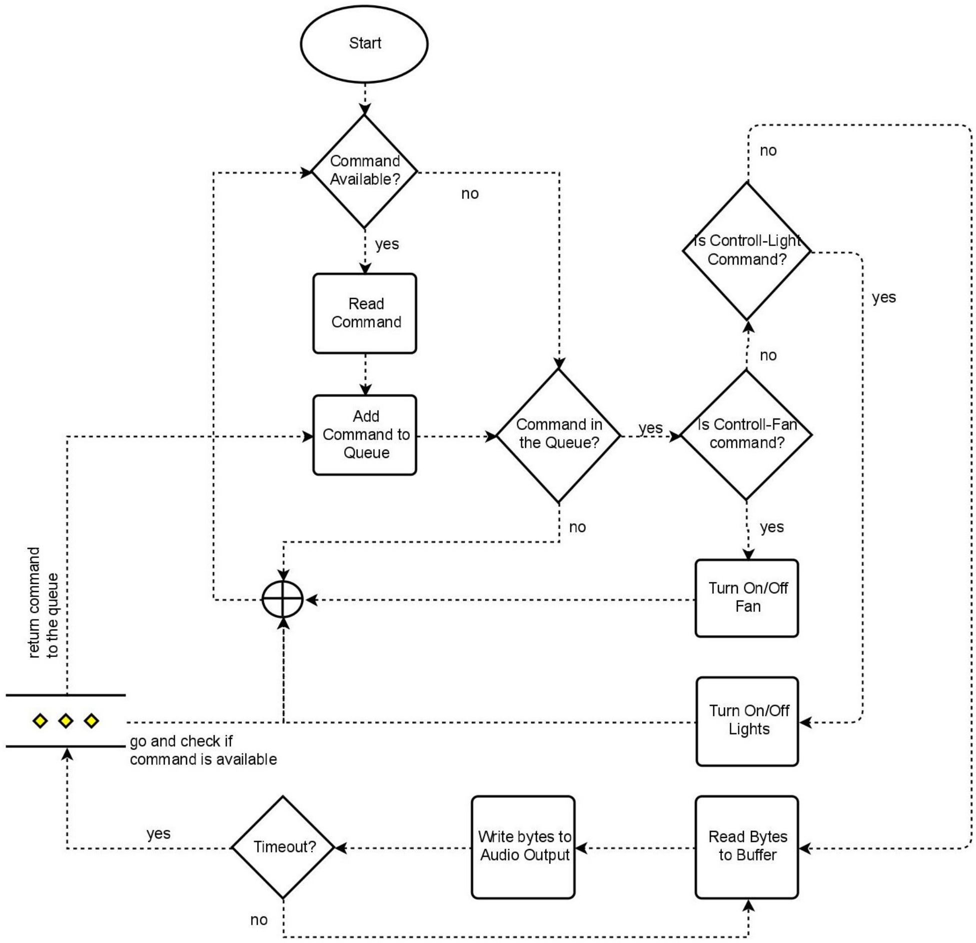

Figure 4.4: The ESP8266 receiving and executing commands

the previous step reading incoming data and writing it to the audio output.

4.3 Server API

We developed a REST API service, which is responsible for the communication between the Web UI application used by the end user, the MQTT broker and the Database server (all discussed in Chapter 3). The solution is based on the Java Spring Boot framework using different integrations for communication with the Database server and the MQTT broker. Hereinafter, an example of using the MQTT Spring Integration Support [31] in our Spring Boot application is provided.

In order our REST API service to communicate with the MQTT broker, we implemented input and output flow for receiving and sending messages to the broker. Listing 4.1 shows the implementation of the input flow. By creating and using an adapter, the service implementation subscribes to a topic on the MQTT broker and waits for incoming messages. In that way the service receives various messages from the ESP8266 device (eg. alive message) throughout the MQTT broker.

On the other hand, Listing 4.2 shows the implementation of the output flow which is responsible for sending and forwarding commands to the MQTT broker and then to the ESP8266 device.

```
// Consumer
@Bean
public IntegrationFlow mqttInFlow(MqttMessageHandler handler) {
    return IntegrationFlows.from(mqttInbound())
            .handle(handler)
            .get();
}

@Bean
public MessageProducerSupport mqttInbound() {
    MqttPahoMessageDrivenChannelAdapter adapter = new
        MqttPahoMessageDrivenChannelAdapter(clientId,
            mqttClientFactory(), defaultTopic);
    adapter.setCompletionTimeout(5000);
    adapter.setConverter(new DefaultPahoMessageConverter());
    return adapter;
}
```

Listing 4.1: Implementation of the input flow in the service API

```
// Publisher
@Bean
@ServiceActivator(inputChannel = "mqttOutboundChannel")
public MessageHandler mqttOutbound() {
    MqttPahoMessageHandler messageHandler = new
       MqttPahoMessageHandler(clientId, mqttClientFactory());
    messageHandler.setAsync(true);
    messageHandler.setDefaultTopic(defaultTopic);
    return messageHandler;
}

@Bean
public MessageChannel mqttOutboundChannel() {
    return new DirectChannel();
}
```

Listing 4.2: Implementation of the output flow in the service API

Our service implementation supports two types of command executions: *immediate* and *delayed*. For that purpose, we defined an `MqttGateway` interface which is responsible for sending messages to the specific topic on the MQTT broker. Listing 4.3 shows our implementation of executing *immediate* and *delayed* commands throughout the gateway interface depicted in Listing 4.4.

```
@Override
public void executeCommand(Device device, Command command) {
    gateway.sendToMqtt("motivationalfurniture/" + device.
       getMacAddress() + "/command", command.getEspCommand());
}

@Override
public void executeCommand(Device device, Command command, Long
   delay) {
    if (delay == null) {
        executeCommand(device, command);
    } else {
```

```
        if (scheduler == null) {
            scheduler = Executors.
                newSingleThreadScheduledExecutor();
        }
        if(future != null) {
            future.cancel(true);
        }
        task = () -> executeCommand(device, command);
        future = scheduler.schedule(task, delay, TimeUnit.
            SECONDS);
    }
}
```

Listing 4.3: *Immediate* and *delayed* execution of commands.

```
@MessagingGateway(defaultRequestChannel = "mqttOutboundChannel")
public interface MqttGateway {
    void sendToMqtt(@Header(MqttHeaders.TOPIC) String topic,
        String data);
}
```

Listing 4.4: MQTT gateway

4.4 Executing the ambient adjusting commands

User adjusts the ambient by controlling the ESP8266 device and the actuators connected to it. Figure 4.5 represents the landing page of the web application after the user successfully logs in into the application. The page consists of three fragments, each responsible for controlling a specific actuator.

- *Audio* fragment controls the playing of audio files and internet radio stations.
- *Light* fragment controls the showing of lighting patterns on the LED strip lights.

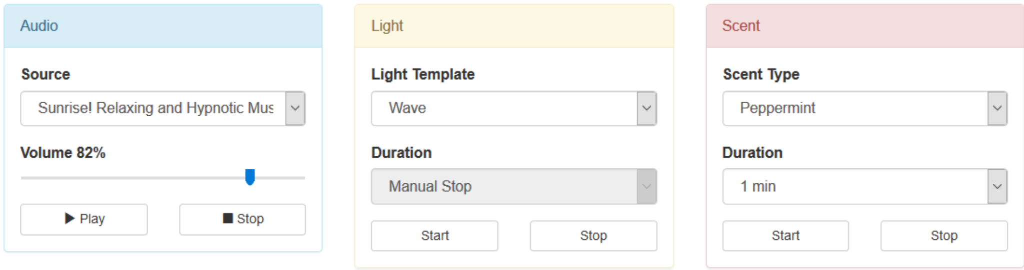

Figure 4.5: Home page of the web application. It contains sections for playing audio files and radios stations, playing lights templates and starting fans for spreading pleasant scents.

- *Scent* controls the working of the fans and spreading of the pleasant scents.

In the sections bellow we will explain the execution flow of each command for ambient adjustment.

4.4.1 Playing audio files and internet radio stations

User is able to play audio files and internet radio stations on the speakers connected to the ESP8266 device. To play an audio file, the user should firstly upload that file on the server. User uploads the file through the web application form showed in Figure 4.6. On the other hand to play an internet radio station, the user should firstly insert the name and the URL link of the desired station through the web application form showed in Figure 4.7.

Figure 4.8 shows the flow diagram of the user requesting audio file to play on the speakers of the system. Bellow we explain the flow in more details.

1. The user chooses an audio file to play in the *Audio* fragment showed in Figure 4.5 and clicks button *Play*.

2. Server API receives the command and starts the process of command execution. It requests Database server to return the selected audio file's path on the server.

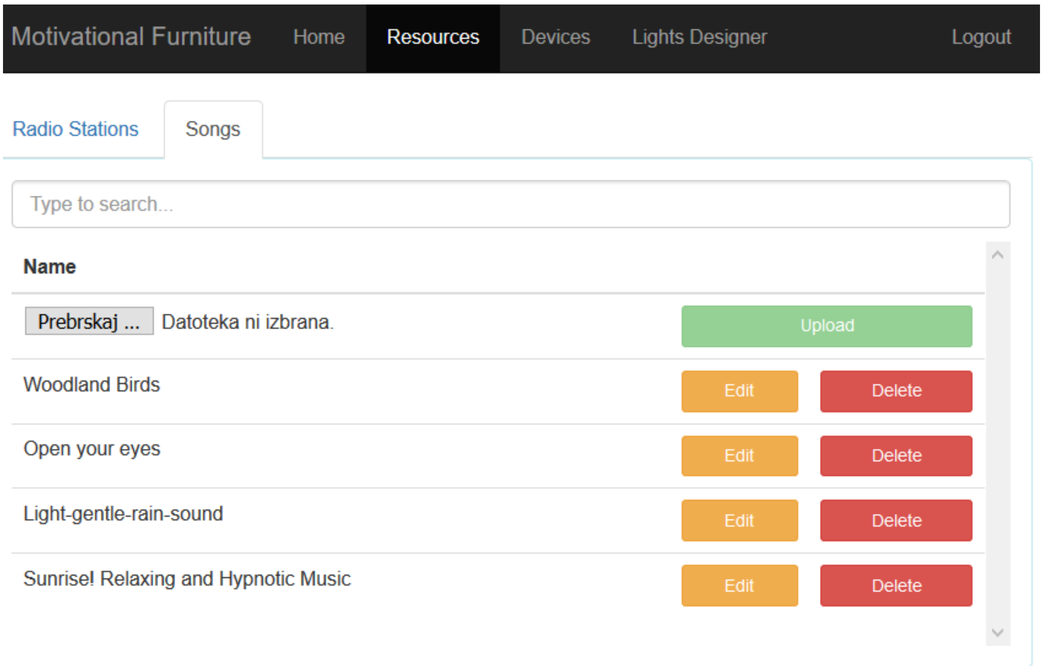

Figure 4.6: Web form for uploading and managing audio files.

3. Database queries for the audio file's path and returns it to the server API.

4. Server creates a command for playing the audio file and publishes it on the MQTT broker.

5. MQTT broker redirects the command to the ESP8266 device.

6. Device receives the command and request the server to start streaming the specified file.

7. Server API receives the command and as a response sends a stream of the requested audio file to the device.

8. Device starts reading from the buffer and simultaneously sends bytes to the audio output where the external speakers are wired to.

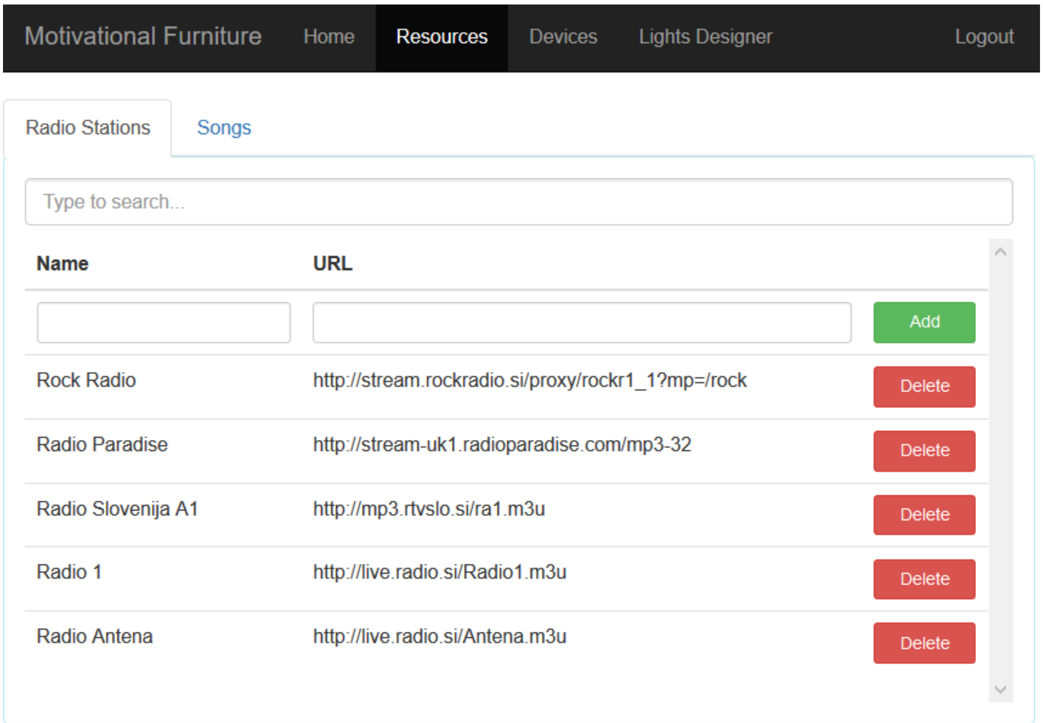

Figure 4.7: Web form for adding and managing internet radio stations.

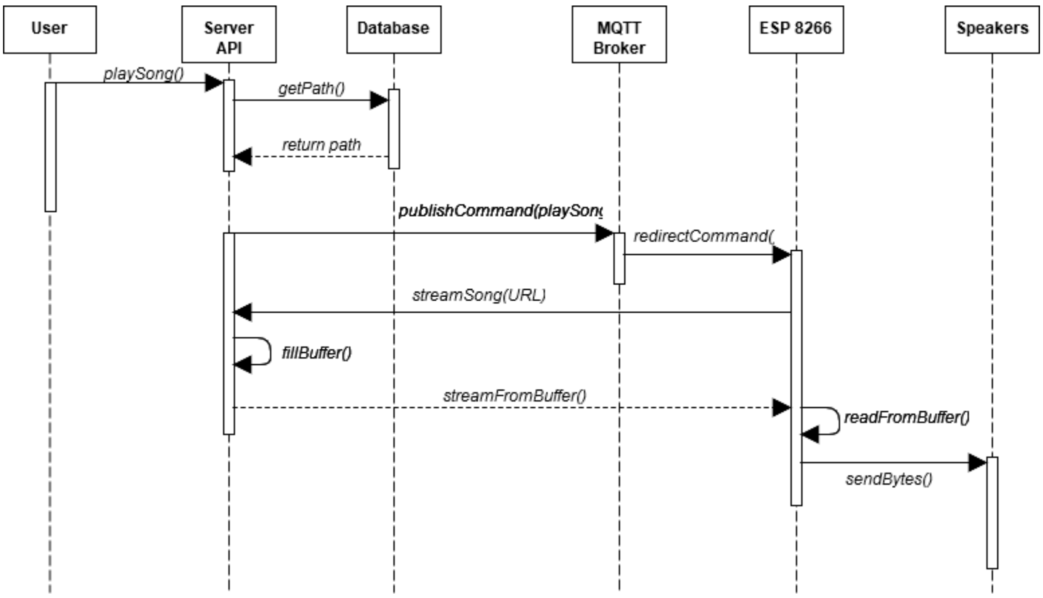

Figure 4.8: Playing song flow

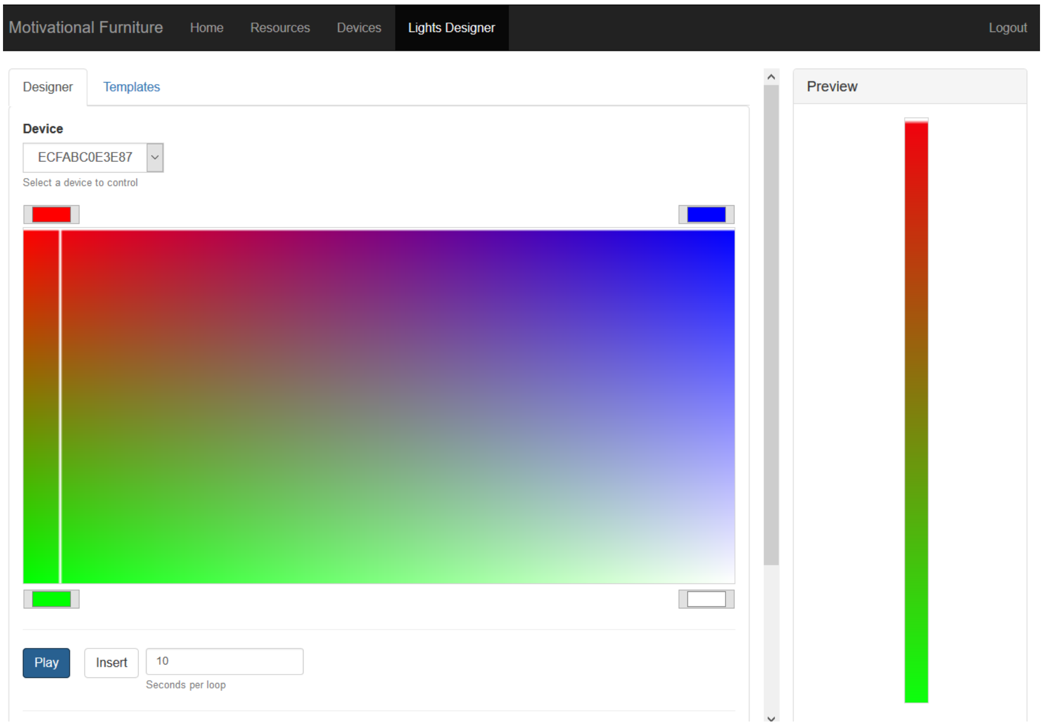

Figure 4.9: Lighting pattern designing form.

4.4.2 Playing lighting patterns

Using the web application, the user can create custom lighting patterns and show them on the LED lights strip. Figure 4.9 shows the UI of the form for creating lighting patterns. The form consists of a canvas and four color picker elements located in the each corner of the canvas. By picking different colors in each corner, the user creates different color patterns as illustrated in the Figure 4.9. The user can directly show the lighting pattern by clicking the button *Play* or can save the pattern for later use. The saved lighting pattern can be shown by clicking *Play* button in the fragment *Light* shown in the Figure 4.5.

Below, we explain the flow of showing the LED lighting patterns depicted in Figure 4.10.

1. The user, through the web application, chooses a light template to be

shown on the LED strip lights and clicks a button for executing the command.

2. Server API receives and starts executing the command. For the selected template it requests Database server to return sequence of RGB values for every LED light on the strip lights.

3. Database queries RGB values for the selected template and returns array of integer values to the server API.

4. Server creates a command for showing light template and publishes it on the MQTT broker. Command is wrapped in JSON string and contains an array of RGB sequence values and the execution time of the sequence.

5. MQTT broker redirects the command to the ESP8266 device.

6. The device receives the command and calls a method (listed in Listing 4.5) for setting and create the lights sequence.

7. Device periodically reads values from the sequence and change the RGB color of each LED light accordingly.

In Listing 4.5, we show the source code running on the ESP8266 device that is responsible for creating the LED lighting pattern.

```
void Lights::setLights(JsonArray& _settings) {
  uint8_t tmp_num_set = number_of_settings;
  for (uint8_t i=0; i< _settings.size(); i++){
    light_data[i+tmp_num_set].time = _settings[i]["duration"];

    for (uint8_t j=0; j < _settings[i]["data"].size(); j++){
      light_data[i+tmp_num_set].data[j] =
            _settings[i]["data"][j];
    }

    light_data[i+tmp_num_set].id = 0;
```

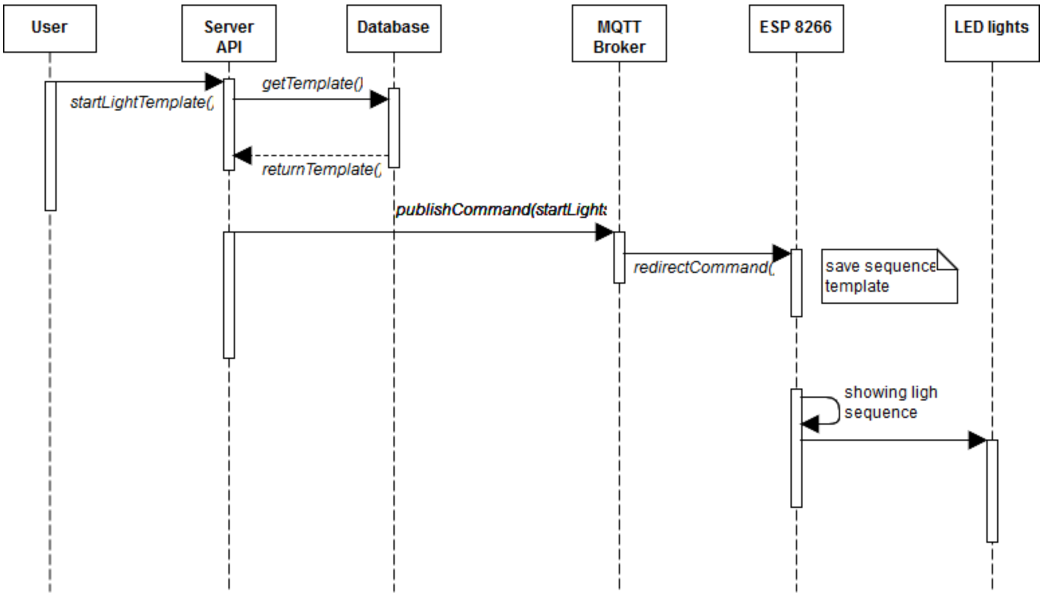

Figure 4.10: Showing lights sequence

```
      if (strcmp(_settings[i]["function"], "blind") == 0){
        light_data[i+tmp_num_set].id = 1;
      } else if (strcmp(_settings[i]["function"], "fixed") == 0){
        light_data[i+tmp_num_set].id = 2;
      }
      number_of_settings++;
    }
}
```

Listing 4.5: Source code of the method, which creates the lighting pattern, written in C++ programming language and running on the ESP8266 device.

4.4.3 Controlling fans and spreading of pleasant scents

Fragment *Scent* in the Figure 4.5 illustrates the UI web form for controlling the fans spreading pleasant scents in the space.

Between the ESP8266 device and the fan there is a relay which, when signaled from the device, turns on/off the working of the fan. The flow of the controlling the working of the fan is explained bellow and depicted by the Figure 4.11.

1. The user, through the web application chooses the scent to be spread and clicks a button for executing the command.

2. Server API receives the command and starts the process of command execution. It requests the Database server to return the ID of the output pin on which the relay controlling the fan is wired to.

3. Database queries the ID of the pin which will activate the fan spreading the selected scent.

4. Server creates a command for starting the selected fan. If the user selects automatic stop of fans, the server also creates a delayed task to execute the stop fan command.

5. MQTT broker redirects the command to the ESP8266 device.

6. Device receives the command and signals the relay to start the selected fan. Fan starts working, producing air flow directed to the scent source and spreading a pleasant scent.

7. After the delay has passed, server sends another command for stopping the fan. MQTT broker intercepts and redirects the command to the ESP8266 device. The device signals the relay to stop the working of the fan.

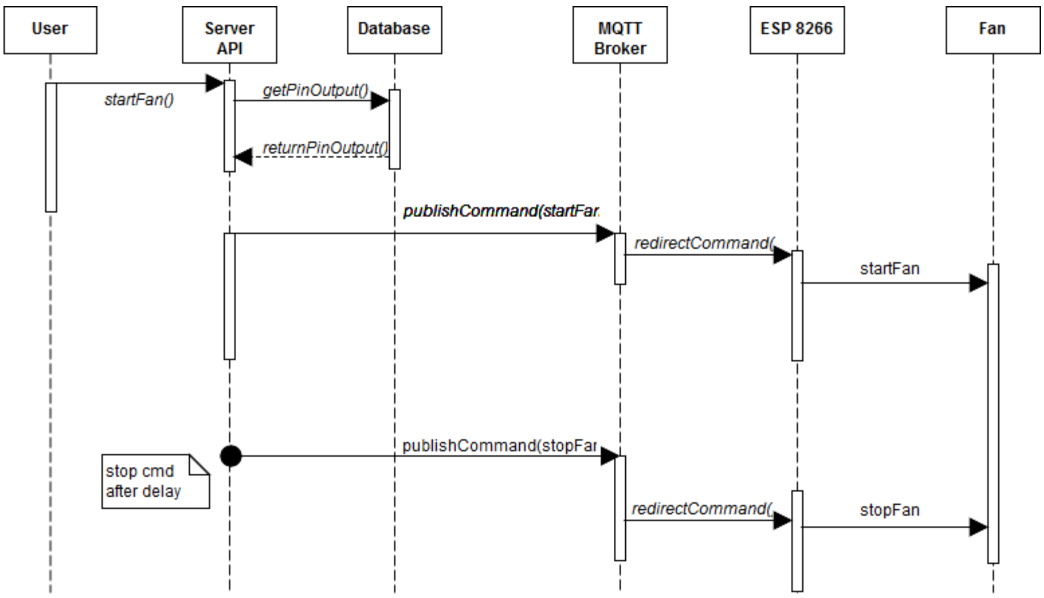

Figure 4.11: Starting and stopping fans

Chapter 5

Results and Analysis

To evaluate the results of this thesis and to compare them with the results of the studies in Chapter 2.2, we carried out an experiment that explores the possible correlation between the participants' emotional state and the use of the system. System, which is depicted in Figure 4.1, was tested in a working office (among our coworkers) and a home environment (among students of the Faculty of Computer and Information Science and a graduate psychologist). Every participant had a 4 days period to test the system at his/her home. Information on the impact of the system on the participants' emotional state was collected using a questionnaire, which was completed by each participant twice: before and after using the system. Using a statistical analysis, we then tried to determine whether the participants' emotional state was affected by the use of the system. The complete questionnaire is provided in Appendix A.

5.1 Method

The questionnaire is based on the Positive and Negative Affect Schedule (PANAS) developed by Watson, Clark, and Tellegen [32], which is one of the most widely used affect scales in psychology and other disciplines [33].

PANAS is a self-report questionnaire composed of two ten-item mood scales: one to measure positive affect (PA) and the other to measure negative

Table 5.1: PANAS scales

1	+	Interested	11	-	Irritable
2	+	Distressed	12	-	Alert
3	+	Excited	13	-	Ashamed
4	-	Upset	14	+	Inspired
5	+	Strong	15	-	Nervous
6	-	Guilty	16	+	Determined
7	-	Scared	17	+	Attentive
8	-	Hostile	18	-	Jittery
9	+	Enthusiastic	19	+	Active
10	+	Proud	20	-	Afraid

affect (NA). As shown in Table 5.1 there are 10 positive emotion items (1, 2, 3, 5, 9, 10, 14, 16, 17, 19) and 10 negative emotion items (4, 6, 7, 8, 11, 12, 13, 15, 18, 20) in the original full form PANAS scales [32].

PA refers to the tendency of experiencing good feelings. Conversely, NA has been defined as the degree to which individuals exhibit negative emotionality, manifest high level of psychological symptoms, and react negatively to stressful situations [32]. The PA and NA scores are the sums of the ratings of the PA items and the NA items, respectively. The PA and NA scores can be used to measure two primary dimensions of mood – Positive and Negative Affect.

5.2 Participants

Tables 5.2, 5.3 and 5.4 include basic information about the participants. Participants of this experiment included 10 people of which 7 males and 3 females. The average age of the participants was 24,9 with standard deviation (SD) of 4,677. Half of the participants were students of the Faculty of Computer and Information Science, University of Ljubljana. The other half were employed.

Table 5.2: Gender

	Number	Percent
Male	7	70%
Female	3	30%
Total	10	100%

Table 5.3: Age

Min :	19
Max :	33
Average :	24,9
SD :	4,677

Table 5.4: Profession

	Number	Percent
Student	5	50%
Employed	5	50%
Total	10	100%

5.3 Measures

Each participant was requested to take the test two times: once before using our system and second after using our system. Participants were also requested to rate these positive and negative adjectives on a 5-point scale (*not at all* to *extremely*) to the extent to which each describes the way they have felt before and after using our system. The higher scores on both PA and NA items indicate the tendency to experience a positive and negative mood.

5.4 Results

Table 5.5 shows the mean values and standard deviation of PA and NA before and after using the system. PA scores range from 10 to 50, with higher scores representing higher levels of positive affect. NA scores range from 10 to 50, with lower scores representing lower levels of negative affect.

The results of our test are depicted in Table 5.6. There are two pairs of results:

- *Pair 1* represents PA before and after using the system
- *Pair 2* represents NA before and after using the system

In the case of PA (*Pair 1*) the negative value of the difference of means indicates that positive affect improved, while in the case of NA (*Pair 2*) the positive value of the difference of means indicates that the negative affect decreased.

To confirm the statistical significance of the results before and after the condition, we used *paired sample t-test* for paired samples. The *paired sample t-test*, sometimes called the dependent sample t-test, is a statistical procedure used to determine whether the mean difference between two sets of observations is zero. In a paired sample t-test, each subject or entity is measured twice, resulting in pairs of observations [34].

The results in our test have shown that there is a significant statistical difference in the both positive and negative emotional scales of PANAS questionnaire before and after the use of the system. There is a statistical difference in the results of PA scale (sig=0,00 and df=9) between the first and the second measurement. According to mean results (M=-5,8 and SD=1,69), positive affect among the participants in the second measurement, increased. On the other hand, we found a significant statistical difference in the results of the NA scale (sig=0,019 and df=9). According to mean results, the negative affect among the participants in the second measurement, decreased.

We tested the results against the *null hypothesis*, which presumes that there is no significant statistical difference in PA and NA before and after

Table 5.5: Statistical data of the PA and NA before and after using the system. With the letter B we denote the affect before using the system, where with the letter A the affect after using the system.

	PA-B	NA-B	PA-A	NA-A
$Mean$	27,8	27,6	33,6	23,9
SD	3,68	3,89	2,95	4,77

Table 5.6: t-paired test

	Difference of Means	SD	t	df	sig.
$Pair1$	-5,8	1,69	-10,88	9	0,000
$Pair2$	3,7	4,08	2,87	9	0,019

the use of the system.

We can conclude that, after using the system, there is a difference in the participant's emotional state. We found that the positive affect increases, on the other hand, negative affect decreases. According to these results, we can reject the *null hypothesis*, and confirm that, there is significant statistical difference in the participant's emotional state, before and after the use of the system.

Chapter 6

Conclusion and future work

In this work we designed and developed a remotely controllable system, which consists of three major components. The first is a hardware consisted of a Wi-Fi enabled chip and actuators connected to it (LED strip lights, speakers and fans). The second is a service running on a server, serving a web application which allows the user remotely to control the Wi-Fi enabled chip. And the third is a communication protocol for connecting the first two components. The user controls the system through a web application in real-time. The service running the web application forwards the user's commands to the Wi-Fi enabled chip which process the commands and start controlling the connected actuators.

The main functionality of the system is enabling the user to adjust the ambient in the living space by playing audio files, showing custom patterns of changing RGB colors on the LED strip lights and spreading pleasant scents in the living space. Our system has better both technical and user experience advantages compared to studies discussed in Chapter 2. Compared to study [5], our system has simpler architecture removing the intermediate aggregation node between the remote server and the controllable unit. In that way we reduce the system's installation complexity and the economic costs. We also combined all environmental stimuli discussed in Chapter 2.2 into one system and enable the end user to experience custom ambient adjustment of music, lighting and pleasant scents in the living spaces.

We carried out an experiment that explores the correlation between the emotional state of the participants and the use of the system. Each participant answered a 20-question PANAS questionnaire two times: that is, before and after using our system. Results have shown that there is a significant statistical difference in the both positive and negative emotions of the user before and after the use of the system. Positive affect among the participants after the use of the system increased, while negative decreased.

In the future we are planning to integrate this system as an integral part of a wider management system for controlling a smart home. Using some sensors and machine learning algorithms we are planning to automate some functions of the system. For example, the system will automatically play calm music and show sunset lighting pattern when user goes in bed.

Appendix A

Questionnaire about the influence of the system to the user's mood and emotional state

1. Gender
   O Man
   O Woman

2. Age
   ----

3. Profession
   O Student
   O Employed

4. Indicate to what extend you feel this way right now:

   (a) Interested
       O Very slightly or not at all
       O A little
       O Moderately
       O Quite a bit
       O Extremly

(b) Distressed
O Very slightly or not at all
O A little
O Moderately
O Quite a bit
O Extremly

(c) Excited
O Very slightly or not at all
O A little
O Moderately
O Quite a bit
O Extremly

(d) Upset
O Very slightly or not at all
O A little
O Moderately
O Quite a bit
O Extremly

(e) Strong
O Very slightly or not at all
O A little
O Moderately
O Quite a bit
O Extremly

(f) Guilty
O Very slightly or not at all
O A little
O Moderately
O Quite a bit
O Extremly

(g) Scared
O Very slightly or not at all
O A little
O Moderately
O Quite a bit
O Extremly

(h) Hostile
O Very slightly or not at all
O A little
O Moderately
O Quite a bit
O Extremly

(i) Enthusiastic
O Very slightly or not at all
O A little
O Moderately
O Quite a bit
O Extremly

(j) Proud
O Very slightly or not at all
O A little
O Moderately
O Quite a bit
O Extremly

(k) Irritable
O Very slightly or not at all
O A little
O Moderately
O Quite a bit
O Extremly

(l) Alert
O Very slightly or not at all
O A little
O Moderately
O Quite a bit
O Extremly

(m) Ashamed
O Very slightly or not at all
O A little
O Moderately
O Quite a bit
O Extremly

(n) Inspired
O Very slightly or not at all
O A little
O Moderately
O Quite a bit
O Extremly

(o) Nervous
O Very slightly or not at all
O A little
O Moderately
O Quite a bit
O Extremly

(p) Determined
O Very slightly or not at all
O A little
O Moderately
O Quite a bit
O Extremly

(q) Attentive
O Very slightly or not at all
O A little
O Moderately
O Quite a bit
O Extremly

(r) Jittery
O Very slightly or not at all
O A little
O Moderately
O Quite a bit
O Extremly

(s) Active
O Very slightly or not at all
O A little
O Moderately
O Quite a bit
O Extremly

(t) Afraid
O Very slightly or not at all
O A little
O Moderately
O Quite a bit
O Extremly

Bibliography

[1] I. Knez, Effects of indoor lighting on mood and cognition, Journal of Environmental Psychology 15 (1) (1995) 39 – 51.

[2] E. Lynar, E. Cvejic, E. Schubert, U. Vollmer-Conna, The joy of heartfelt music: An examination of emotional and physiological responses, International Journal of Psychophysiology 120 (Supplement C) (2017) 118 – 125.

[3] A. Gagarina, I. Pikturnienė, The effect of ambient scent type and intensiveness on decision making heuristics, Procedia - Social and Behavioral Sciences 213 (2015) 605 – 609, 20th International Scientific Conference "Economics and Management 2015 (ICEM-2015)".

[4] A. F. da Silva, R. L. Ohta, M. N. dos Santos, A. P. Binotto, A cloud-based architecture for the internet of things targeting industrial devices remote monitoring and control, IFAC-PapersOnLine 49 (30) (2016) 108 – 113, 4th IFAC Symposium on Telematics Applications TA 2016.

[5] A. Glória, F. Cercas, N. Souto, Design and implementation of an iot gateway to create smart environments, Procedia Computer Science 109 (2017) 568 – 575, 8th International Conference on Ambient Systems, Networks and Technologies, ANT-2017 and the 7th International Conference on Sustainable Energy Information Technology, SEIT 2017, 16-19 May 2017, Madeira, Portugal.

[6] MQTT (Message Queue Telemetry Transport) protocol, http://mqtt.org/, accessed: 01.12.2017.

[7] J. Karim, R. Weisz, S. U. Rehman, International positive and negative affect schedule short-form (I-PANAS-SF): Testing for factorial invariance across cultures, Procedia - Social and Behavioral Sciences 15 (2011) 2016 – 2022, 3rd World Conference on Educational Sciences - 2011.

[8] D. Ding, R. A. Cooper, P. F. Pasquina, L. Fici-Pasquina, Sensor technology for smart homes, Maturitas 69 (2) (2011) 131 – 136.

[9] K. Gram-Hanssen, S. J. Darby, "Home is where the smart is"? Evaluating smart home research and approaches against the concept of home, Energy Research & Social Science 37 (Supplement C) (2018) 94 – 101.

[10] M. Alaa, A. Zaidan, B. Zaidan, M. Talal, M. Kiah, A review of smart home applications based on Internet of Things, Journal of Network and Computer Applications 97 (Supplement C) (2017) 48 – 65.

[11] C. L. B. McCloughan, P. Aspinall, R. Webb, The impact of lighting on mood, International Journal of Lighting Research and Technology 31 (3) (1999) 81–88.

[12] R. A. Rahadi, Factors related to repeat consumption behaviour: A case study in traditional market in bandung and surrounding region, Procedia - Social and Behavioral Sciences 36 (Supplement C) (2012) 529 – 539, aSEAN Conference on Environment-Behaviour Studies (AcE-Bs), Savoy Homann Bidakara Hotel, 15-17 June 2011, Bandung, Indonesia.

[13] B. Raudenbush, N. Corley, W. Eppich, Enhancing athletic performance through the administration of peppermint odor, Journal of Sport and Exercise Psychology 23 (2) (2001) 156–160.

[14] T. Abdulrahman, O. Isiwekpeni, N. Surajudeen-Bakinde, A. Otuoze, Design, specification and implementation of a distributed home automation system, Procedia Computer Science 94 (2016) 473 – 478, the

11th International Conference on Future Networks and Communications (FNC 2016) / The 13th International Conference on Mobile Systems and Pervasive Computing (MobiSPC 2016) / Affiliated Workshops.

[15] Database, `https://en.wikipedia.org/wiki/Database`, accessed: 11.08.2018.

[16] Database server, `https://en.wikipedia.org/wiki/Database_server`, accessed: 11.08.2018.

[17] PostgreSQL, `https://www.postgresql.org/about/`, accessed: 12.08.2018.

[18] Hibernate ORM, `http://hibernate.org/orm/`, accessed: 11.08.2018.

[19] Spring Boot Security 2.0, `https://github.com/spring-projects/spring-boot/wiki/Spring-Boot-Security-2.0`, accessed: 10.07.2018.

[20] Web server, `https://whatis.techtarget.com/definition/Web-server`, accessed: 10.10.2018.

[21] Apache Tomcat, `http://tomcat.apache.org/`, accessed: 10.07.2018.

[22] Spring Boot, `https://github.com/spring-projects/spring-boot`, accessed: 10.07.2018.

[23] H. Suryotrisongko, D. P. Jayanto, A. Tjahyanto, Design and development of backend application for public complaint systems using microservice spring boot, Procedia Computer Science 124 (2017) 736 – 743, 4th Information Systems International Conference 2017, ISICO 2017, 6-8 November 2017, Bali, Indonesia.

[24] Dependency Injection, `https://searchmicroservices.techtarget.com/definition/dependency-injection`, accessed: 10.10.2018.

[25] MQTT, https://en.wikipedia.org/wiki/MQTT, accessed: 01.12.2017.

[26] Publish–subscribe pattern, https://en.wikipedia.org/wiki/Publish%E2%80%93subscribe_pattern, accessed: 01.12.2017.

[27] Thymeleaf, https://www.thymeleaf.org/, accessed: 09.10.2018.

[28] ESP8266, https://en.wikipedia.org/wiki/ESP8266, accessed: 09.09.2018.

[29] ESP8266 NodeMCU CP2102 board, https://robu.in/product/nodemcu-cp2102-board/, accessed: 09.09.2018.

[30] Relay, https://en.wikipedia.org/wiki/Relay/, accessed: 19.10.2018.

[31] Mqtt Support, https://docs.spring.io/spring-integration/reference/html/mqtt.html, accessed: 07.06.2018.

[32] D. Watson, L. Clark, A. Tellegen, Development and validation of brief measures of positive and negative affect: the PANAS scales., Journal of personality and social psychology 54 6 (1988) 1063–70.

[33] C. Kuesten, P. Chopra, J. Bi, H. L. Meiselman, A global study using PANAS (PA and NA) scales to measure consumer emotions associated with aromas of phytonutrient supplements, Food Quality and Preference 33 (2014) 86 – 97.

[34] Paired sample t-test, http://www.statisticssolutions.com/manova-analysis-paired-sample-t-test/, accessed: 14.10.2018.