NOVA SCHOOL OF
SCIENCE & TECHNOLOGY

Sistemas de Computação Móvel e Ubíqua

Prof. Vitor Duarte e Prof. Carmen Morgado

Final Report

Project done by:

Rodrigo Mesquita 57372

Ana Carolina Gadelha 59943

Ruben Andrade 58116

1. Introduction

For any driver, it is essential to have a place to park the vehicle, and normally the driver wants it to be a quick and safe process.

With SpotFinder, these concerns will no longer exist. Our self-managed service uses sensors to detect empty parking spots, keeps track of how many spots are available, automatically controls the barrier gate and the air quality in scenarios such as underground parking lots. Beyond this, it lets people reserve parking spots in advance on their phones.

It's all about making parking simple and convenient for everyone involved.

2. Overview

Below is an overview of the possible functionalities available to the end user:

- Registration into the system, creating a new user.
- Login, to use the system.
- Search for parking lots in the map, to more easily find the desired one.
- Select a parking lot to check how many parking spaces are still available, as well as the hourly rate and the amount of spots for reservation.
- Make a reservation to book a spot in a specific parking lot in a given time frame.
- Report an issue with a parking lot, leaving a message telling us what's wrong.
- Stop the current stay at the parking lot, where the price will be calculated and the user will be prompted to pay

For the application, it is built with React Native using a backend developed with Express API and a Firebase Firestore as a database.

For the ubiquitous part, we are using a proximity sensor and an RFID reader for sensing the empty parking spot and allowing entrance, and for actuators we're using an LCD display to show the number of available spots and a servo motor to control the entrance barrier.

3. Architecture

For the development of our prototype we chose to use a cardboard box to simulate an underground parking lot. As a makeshift of the barrier gates (entrance and exit gates) we will use paper straws for the barriers and servo motors to lift and lower the barriers. To simulate the availability lights we will use a LED above each spot, these will change accordingly to a

proximity ultrasonic sensor that will detect if a car is in the spot or not and change the LED from green to red. For the entrance we want to display to the drivers if the park is full or not so we will use an LCD display to show how many spots are available. Besides this, for the barrier to open we will use an RFID scanner to read the NFC signal from the user's card allowing him entrance which would start the timer for the payment.

Regarding the reserved spots we would have a select area with the same entrance system as the main park, available to customers who pay extra to have a guaranteed spot.

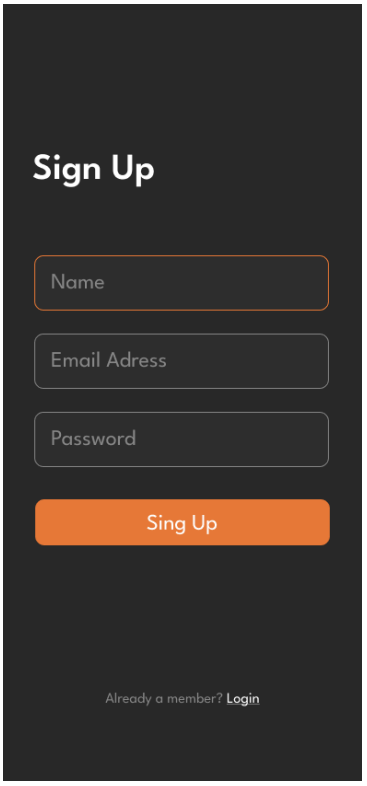
We will use an ESP32-WROVER kit as the controller for the ubiquitous part.

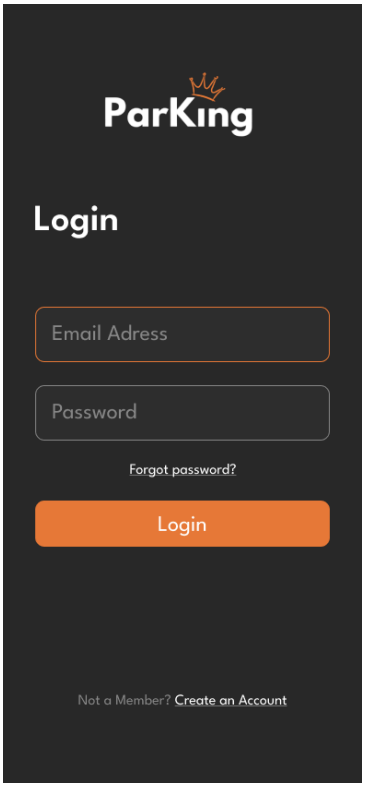
4. Application

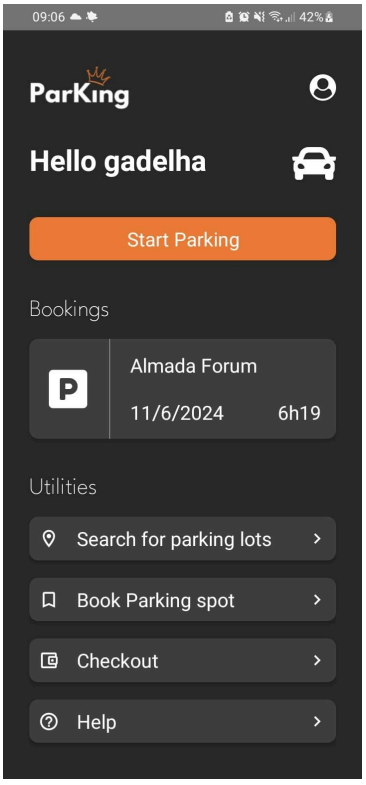
The application was implemented in React Native using VS Code. There is also a backend service to communicate with both the mobile application, the ubiquitous part and the database. It is able to process requests from both sides and offload most of the logic from the application and the ubiquitous side.

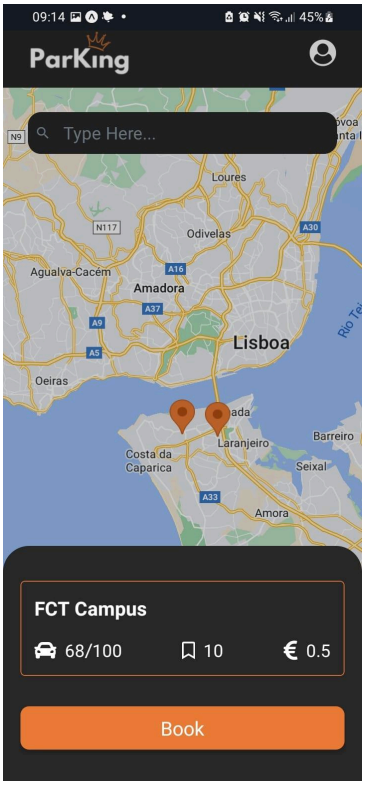
5. Design

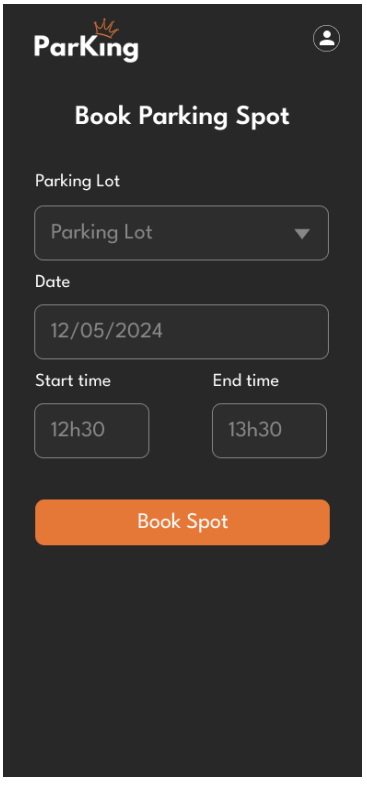

ParKing

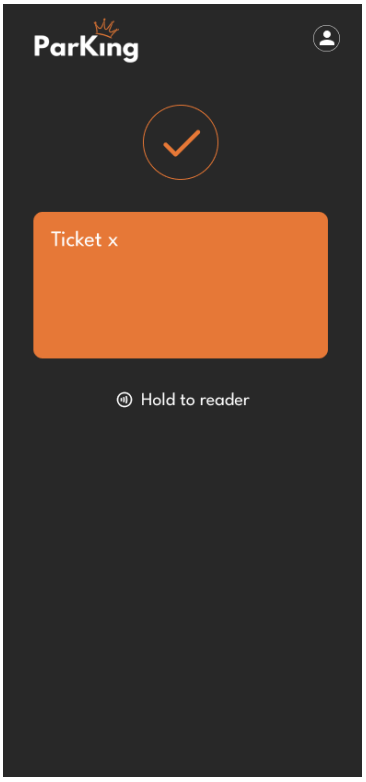

Sign Up
Name
Email Adress
Password
Sing Up
Already a member? Login

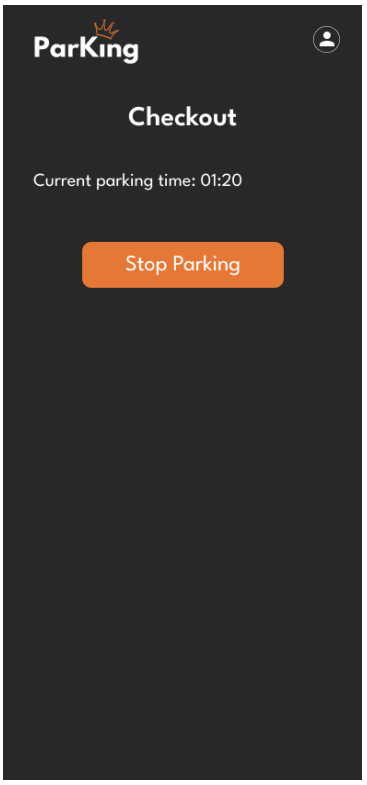

ParKing
Login
Email Adress
Password
Forgot password?
Login
Not a Member? Create an Account

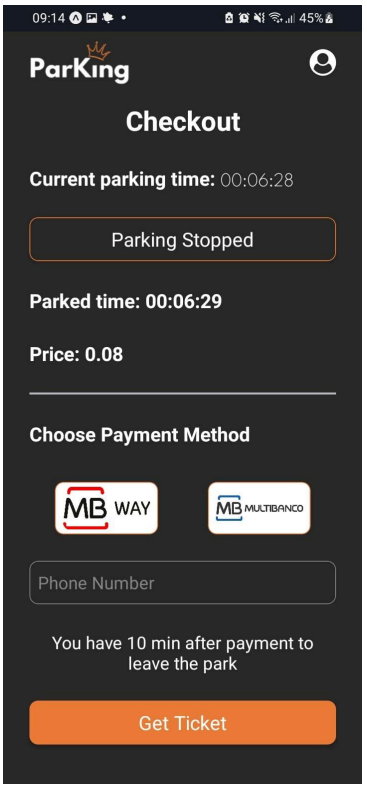

09:06
42%
ParKing
Hello gadelha
Start Parking
Bookings
Almada Forum
11/6/2024
6h19
Utilities
Search for parking lots
Book Parking spot
Checkout
Help

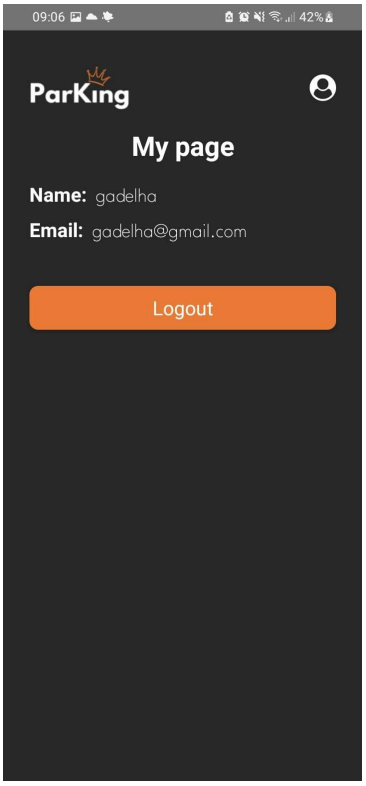

09:14
45%
ParKing
Type Here...
Loures
Odivelas
Agualva-Cacém
Amadora
Lisboa
Oeiras
Laranjeiro
Barreiro
Seixal
Costa da Caparica
Amora
FCT Campus
68/100
10
0.5
Book

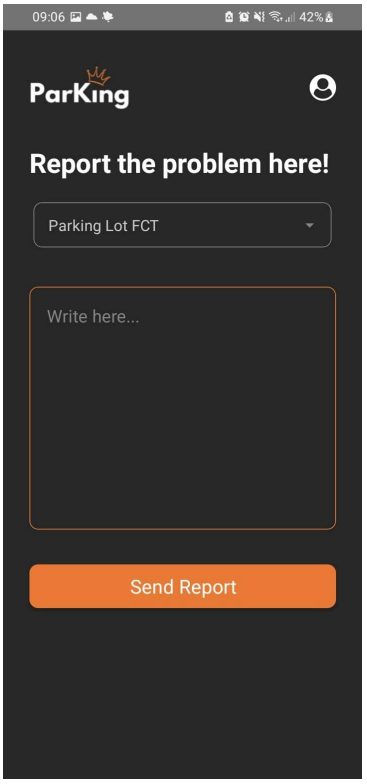

ParKing
Book Parking Spot
Parking Lot
Parking Lot
Date
12/05/2024
Start time
End time
12h30
13h30
Book Spot


ParKing
Ticket x
Hold to reader


ParKing
Checkout
Current parking time: 01:20
Stop Parking


09:14
45%
ParKing
Checkout
Current parking time: 00:06:28
Parking Stopped
Parked time: 00:06:29
Price: 0.08
Choose Payment Method
MB WAY
MB MULTIBANCO
Phone Number
You have 10 min after payment to leave the park
Get Ticket


09:06
42%
ParKing
My page
Name: gadelha
Email: gadelha@gmail.com
Logout


09:06
42%
ParKing
Report the problem here!
Parking Lot FCT
Write here...
Send Report

6. Sensors and Actuators

Proximity Sensor:

```
31  // Proximity Ultrasonic Sensor
32  #define trigPin 12
33  #define echoPin 13
34  #define MAX_DISTANCE 700
35  float timeOut = MAX_DISTANCE * 60;
36  int soundVelocity = 340;
37  float sonarResult;
```

setup()

```
74    // Proximity Ultrasonic Sensor setup
75    pinMode(trigPin, OUTPUT);  // set trigPin to output mode
76    pinMode(echoPin, INPUT);   // set echoPin to input mode
```

loop()

```
134   delay(500);
135
136   getSonarDistance();
```

```
void getSonarDistance() {
  Serial.printf("Distance: ");
  sonarResult = getSonar();
  Serial.print(sonarResult); //
  Serial.println("cm");
}
```

```
162  float getSonar() {
163   unsigned long pingTime;
164   float distance;
165   // make trigPin output high level lasting for 10μs to triger HC_SR04?
166   digitalWrite(trigPin, HIGH);
167   delayMicroseconds(10);
168   digitalWrite(trigPin, LOW);
169   // Wait HC-SR04 returning to the high level and measure out this waitting time
170   pingTime = pulseIn(echoPin, HIGH, timeOut);
171   // calculate the distance according to the time
172   distance = (float)pingTime * soundVelocity / 2 / 10000;
173   return distance; // return the distance value
174  }
```

7. Experiments

The best way we found to test our prototype was to experiment with different scenarios and use cases, to cover as much as we can of our expected operations and make sure they work correctly. One of the cases where this was helpful was in a situation where when a user was entering the park and they scanned their card, we needed to know if he was

there as a normal user or to make use of his reservation. As such, we then had to implement additional logic to check for this case.

There were many other situations like the one above, where we tried something, found a check we were not doing, and fixed it right after, providing us with a much more robust system.

8. Conclusion

Overall, this was a great learning experience. We had little to no experience whatsoever with microcontrollers, sensors and actuators, even ubiquitous computing in general, and this was a great opportunity to learn hands-on what it's all about.

It had its ups and downs, some times more fun than others (getting some of these sensors to work can be frustrating) but in sum it was a good experience.

As much as we would like to, this would not be as easy to implement in a real world scenario. The architecture in the ubiquitous side would need to be vastly different, since we can only attach a limited number of sensors to one Arduino module, we would need to have multiple Arduinos to deal with the generally large number of parking spots in a regular parking lot. This would also mean that additional logic would need to be added to facilitate maintenance in case one of them malfunctioned (like identifying each arduino to narrow down the search for the malfunctioning one).

On the other hand, a real parking lot would require a much better prepared infrastructure for the central service, including but not limited to cloud replication and division into microservices (instead of monolith) to deal with the vast amounts of data being sent/processed.

Despite all of this, we think that creating this project and bringing it to life, even if on a small scale, is already a great achievement that we can be proud of.