Universidade de Aveiro Departamento de Engenharia Mecânica
2021

Fabrício Gabriel Vaz de Souza

Concepção e desenvolvimento de um sistema de bloqueio interconectado para bicicleta

Universidade de Aveiro Departamento de Engenharia Mecânica
2021

Fabrício Gabriel Vaz de Souza

Concepção e desenvolvimento de um sistema de bloqueio interconectado para bicicleta

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestrado em Engenharia Mecânica, realizada sob orientação científica de António Manuel Godinho Completo, Professor Associado c/ Agregação do Departamento de Engenharia Mecânica da Universidade de Aveiro e de José Paulo Oliveira Santos, Professor Auxiliar do Universidade de Aveiro da Universidade de Aveiro.

O júri / The jury

Presidente / President	**Prof. Doutor Marco Paulo Soares dos Santos** Professor Auxiliar da Universidade de Aveiro
Vogais / Committee	**Prof. Doutor António Manuel Godinho Completo** Professor Associado c/ Agregação da Universidade de Aveiro (orientador)
	Prof. Doutor José Paulo Oliveira Santos Professor Auxiliar da Universidade de Aveiro (co-orientador)
	Prof. Doutor Rui Manuel Escadas Ramos Martins Professor Auxiliar da Universidade de Aveiro (Arguente)

Agradecimentos

Em primeiro lugar gostaria de agradecer a Deus, que até aqui me tem sustentado e que, graças a Ele, consegui terminar esta dissertação. A seguir gostaria também de agradecer aos meus orientadores a disponibilidade e apoio ao longo da tese que muitas vezes não foi aproveitada por mim da melhor maneira. Gostaria também de deixar os meus sinceros agradecimentos aos meus pais por todo o apoio e preocupação ao longo desta caminhada; ao meus amigos Filipe, Filipa, Cristiana, Raquel, Daniel que me apoiaram ativamente neste projeto e sempre me encorajaram mesmo quando nem eu acreditava em mim; Aos irmãos da IMW Estarreja por todas as orações e palavras de carinho; aos colegas que fizeram parte desta caminhada, me apoiaram ao longo do curso em noites infinitas de trabalho; e finalmente à pessoa mais importante, que me tem aturado nestes dias de cansaço e má disposição, obrigado Beatriz!

Palavras-chave Cadeado Inteligente; Bicicleta; Bluetooth; PWA; ESP-32; GPS; Parafuso Sem-Fim; QFD; Mudge

Resumo

A crescente procura por bicicletas tradicionais e elétricas é uma oportunidade para o mercado dos sistemas de bloqueio e antirroubo destes veículos. Nesta dissertação pretende-se estudar o mercado de cadeados inteligentes, perceber os seus pontos fortes e as suas limitações; criar um conceito de cadeado inteligente que tente preencher as lacunas deste mercado tendo em conta as necessidades do cliente; e finalmente, executar o projeto mecânico e automação do sistema.O estudo de mercado dos cadeados inteligentes, permitiu perceber as principais características destes sistemas de bloqueio, vantagens, limitações e extras. Identificaram-se as características necessárias para o utilizador e recorrendo ao diagrama de Mudge ordenaram-se estas necessidades por ordem decrescente de importância para o cliente. Utilizou-se o diagrama QFD para identificar as características críticas para o cliente, especificar a correlação entre as necessidades do cliente e os requisitos de projeto e ordenar os requisitos do projeto por ordem decrescente de atuação.O conceito do cadeado desenvolvido é de fácil montagem, com elevada compatibilidade entre bicicletas e sem necessidade de um objeto externo para o bloquear, com um mecanismo de bloqueio do tipo coroa-parafuso sem-fim. A comunicação entre o cadeado e o utilizador é feita por Bluetooth e GSM/GPRS para permitir um alcance global. O desbloqueio da bicicleta pode ser remoto, desbloqueando o cadeado com a aproximação do utilizador ou manual, no caso de falta de bateria. A aplicação possibilita o bloqueio e desbloqueio da bicicleta permitindo a partilha da mesma com terceiros; efetua a monitorização de estatísticas como a distância percorrida, velocidade média e calorias queimadas e consegue localizar a bicicleta por GPS. O dimensionamento mecânico do sistema com parafuso sem-fim permitiu a determinação dos parâmetros característicos da coroa e do parafuso. A coroa possui um diâmetro primitivo de 133 mm, com um diâmetro externo de 136 mm e um diâmetro interno de 129 mm, com 76 dentes. O parafuso sem-fim tem um diâmetro primitivo de 10 mm, diâmetro externo de 13 mm e diâmetro interno de 6 mm. O comprimento máximo do sem-fim é de 34 mm e o comprimento mínimo é de 18 mm. A pesquisa e desenvolvimento de novas tecnologias de bloqueio automático permite a melhoria dos cadeados inteligentes aumentando a sua segurança e as alternativas disponíveis no mercado proporcionando uma melhor e mais personalizada experiência ao utilizador.

Keywords Smart Padlock; Bicycle; Bluetooth; PWA; ESP-32; GPS; Worm Drive; QFD; Mudge

Abstract

The growing demand for traditional bicycle and e-bikes is an opportunity for the market of locking and antitheft systems for these types of vehicles. This dissertation intends to study the smart padlocks market, understand its strengths and limitations; create an intelligent padlock concept that tries to fill the gaps in this market, considering the customer's needs; and finally, perform the mechanical design and automation of the system.The benchmarking of smart padlocks allowed us to understand the main characteristics of these locking systems, advantages, limitations and extras. User's main features were identified and, using the Mudge diagram, these needs were ranked in descending order of value to the client. The QFD diagram was used to identify critical features for the customer, specify the correlation between customer needs and project requirements, and order the project requirements in descending order of improvement priority. The generated padlock's concept is easy to mount on a typical bike, with high compatibility between bicycles and without the need for an external object to lock it, with a worm drive locking mechanism. The communication between the lock and the user is made by Bluetooth and GSM/GPRS to allow a global reach. The bicycle can be unlocked remotely, unlocking the padlock with the user's approach or manual, in case of lack of battery. The application makes it possible to lock and unlock the bike, allowing it to be shared with third parties; tracks statistics such as the distance travelled, average speed and calories burned and can track the bike by GPS.The mechanical dimensioning of the worm drive system allowed the determination of the parameters of the worm wheel and the worm. The worm wheel has a prime diameter of 133 mm, with an outer diameter of 136 mm and an inner diameter of 129 mm, with 76 teeth. The worm has a prime diameter of 10 mm, an outer diameter of 13 mm and an inner diameter of 6 mm. The maximum worm length is 34 mm and the minimum length is 18 mm.The research and development in this field allows the improvement of smart padlocks, increasing security and alternatives available on this market, providing a better and more personalized experience for the user.

Índice

Lista de Tabelas

Lista de Figuras

Lista de Acrónimos

APP - Aplicação Móvel

ASCII - American Standard Code for Information Interchange

BLE - Bluetooth Low Energy

CSS - Cascading Style Sheets

DC - Direct Current

EPA - Environmental Protection Agency

GPRS - General Packet Radio Service

GPS - Global Positioning System

GSM - Global System for Mobile Communication

HTML - HyperText Markup Language

LED - Light-Emitting Diode

NFC - Near Field Communication

NMEA - National Marine Electronics Association

PWA - Progressive Web Application

QFD - Quality Function Deployment

RFID - Radio Frequency Identification

RSSI - Received Signal Strength Indicator

SMS - Short Message Service

USB - Universal Serial Bus

Capítulo 1

Introdução

Neste capítulo vão ser enumerados os objetivos deste projeto, as suas motivações e o que levou ao desenvolvimento de um cadeado inteligente. No fim é descrita a forma como esta dissertação está organizada e exposto o conteúdo de cada capítulo.

1.1 Objetivo

Esta dissertação tem como principais objetivos estudar o mercado de cadeados inteligentes, perceber os seus pontos fortes e as suas limitações; criar um conceito de cadeado inteligente que tente preencher as lacunas deste mercado tendo em conta as necessidades do cliente; e, finalmente, a execução de projeto mecânico e automação do sistema.

1.2 Motivação

Segundo a *Environmental Protection Agency* (`EPA`) a emissão de gases de efeito estufa para a atmosfera como o dióxido de carbono, o metano e os óxidos de nitrogénio, têm vindo a aumentar desde a revolução industrial [1]. Em 2020 a concentração anual de CO_2 na atmosfera registou o valor de 412.5 ppm, cerca de 50 % superior ao valor detetado no início do século XVIII [2]. O aumento da concentração destes gases na atmosfera provoca um aumento da poluição do ar, com consequências para a saúde das populações, nomeadamente doenças respiratórias e cardiovasculares, formação de chuvas ácidas e aumento da temperatura na Terra com efeitos destrutivos nos ecossistemas. Na Figura 1.1 representa-se as emissões de gases de efeito estufa desde 1190 até 2019. Da análise gráfica é possível concluir que dióxido de carbono é o gás de efeito estufa mais emitido para a atmosfera [1].

Comparando os diferentes setores que emitem gases de efeito estufa é possível verificar que os setores dos transportes e da geração de eletricidade são os maiores emissores. Na Figura 1.2 apresenta-se um gráfico das quantidades de gases de efeito de estufa emitidos por diferentes setores desde 1990 a 2019.

Devido ao aumento da concentração de gases de efeito estufa na atmosfera, às consequências causadas por esse acréscimo e pelo crescimento da preocupação da poluição com estes efeitos, começaram-se a implementar medidas para a diminuição da emissão destes gases. Estas medidas contemplam a regulação das emissões permitidas às indústrias, a utilização de fontes de energia alternativas como as energias renováveis, a

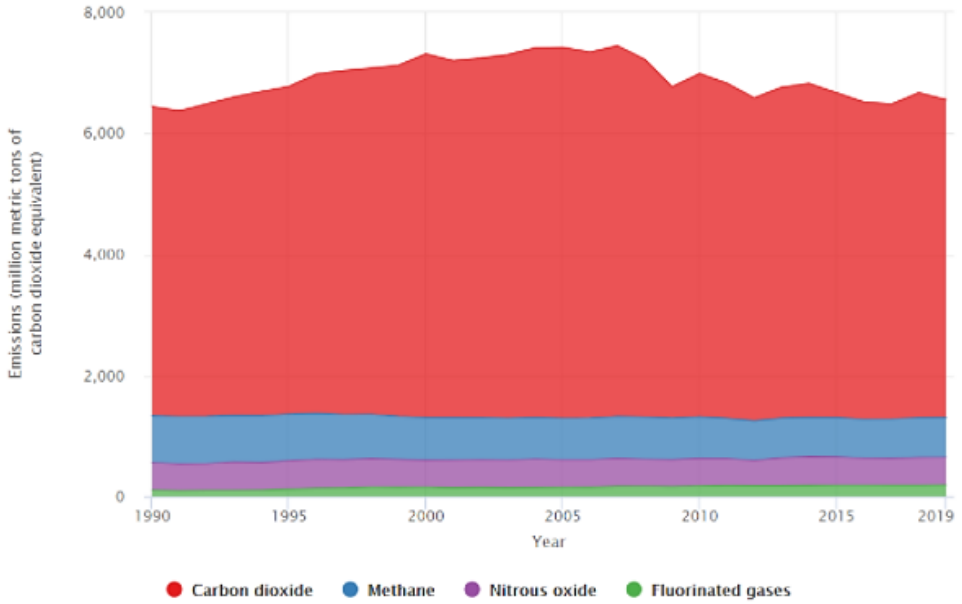


Figura 1.1: Emissões de gases de efeito estufa desde 1190 até 2019 [1].

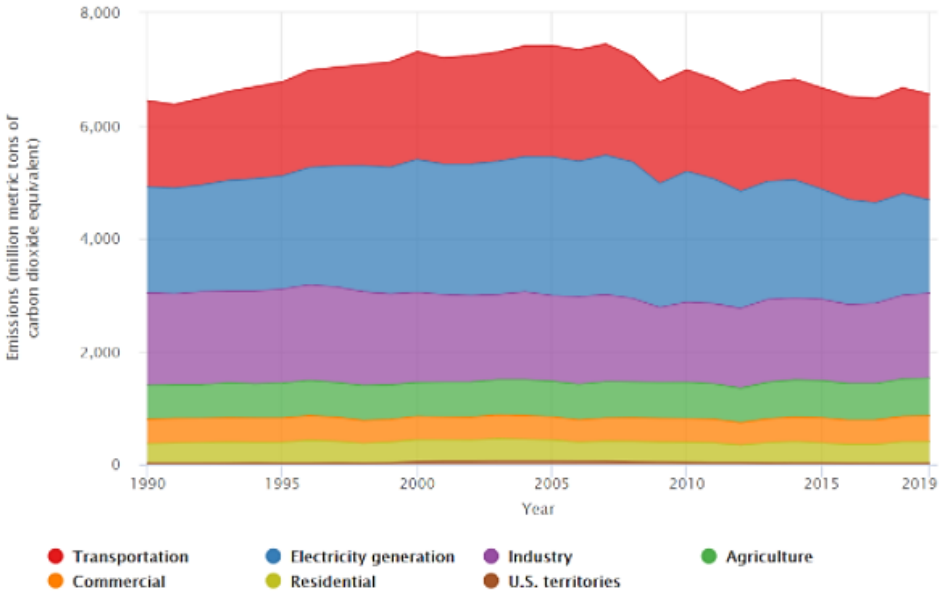


Figura 1.2: Quantidades de gases de efeito de estufa emitidos por diferentes setores desde 1990 a 2019 [1].

consciencialização da população para a utilização maioritariamente de veículos coletivos e de velocípedes, etc. A utilização da bicicleta ao invés do carro diminui as emissões de CO_2 em 150 gramas por quilómetro. Cada 7 km de bicicleta economiza uma emissão de 1 quilo de CO_2 em comparação com a mesma distância percorrida de carro [3].

O mercado das bicicletas tradicionais e das bicicletas elétricas tem vindo a aumentar nos últimos anos, verificando-se uma crescente utilização das bicicletas elétricas partilháveis em diferentes cidades. Segundo Matt Powell, vice-presidente do grupo NPD, as vendas de bicicletas subiram 55 % entre dezembro de 2020 e fevereiro de 2021, quando comparadas com o mesmo período do ano anterior [4]. Segundo um estudo de 2015 para o *Journal of Transport & Health* 42 % das famílias em todo o mundo possuem pelo menos uma bicicleta, correspondendo a mais de 580 milhões de bicicletas [5]. De acordo com *UN Environment's Global Report*, nas cidades chinesas até 60 % das deslocações são efetuadas de bicicleta [3].

O crescimento do mercado das bicicletas tradicionais e elétricas é uma motivação para o desenvolvimento de novas formas de bloqueio das bicicletas, como os cadeados automáticos, que além de serem mecanismos eficazes antirroubo, possuem outras características atrativas para os utilizadores.

1.3 Organização

Esta dissertação encontra-se dividida em 6 capítulos, passando-se a descrever os assuntos abordados nos mesmos.

- No capítulo 1, apresentam-se os objetivos deste trabalho, a sua motivação e a organização da dissertação;
- No capítulo 2, efetua-se um estudo comparativo dos cadeados para bicicletas disponíveis no mercado;
- No capítulo 3, são identificadas as necessidades do utilizador e a sua ordem de importância e estas são correlacionadas com os requisitos de projeto de forma a perceber-se quais as especificações imprescindíveis e as menos relevantes. É, posteriormente gerado um conceito tendo em conta todas as informações adquiridas, do cliente e do mercado;
- No capítulo 4, é exposto o projeto de automação incluindo material utilizado, esquemas de montagem e programação do microcontrolador e da aplicação e o seu funcionamento;
- No capítulo 5, é apresentado o dimensionamento mecânico do sistema com parafuso sem-fim e exposta metodologia utilizada;
- No capítulo 6, são abordadas as principais conclusões do trabalho e sugeridas algumas propostas para trabalhos futuros.

Capítulo 2

Benchmarking

2.1 Introdução

Neste capítulo é elaborado o conceito do cadeado a desenvolver, a sua tipologia e características principais. A seguir, é também avaliada a situação do mercado, isto é, que tipos de sistemas de bloqueio de bicicletas existem, quais as suas principais características, de que forma tentam responder às necessidades do utilizador, as suas principais falhas e os seus pontos positivos. Por fim, é apresentada e analisada uma tabela resumo, que tenta comparar os vários produtos apresentados no *benchmarking* de modo retirar os melhores aspetos de cada um e evitar os seus pontos negativos, de modo a obter um produto superior aos já existentes.

2.2 Enquadramento

Como abordado no capítulo 1, o objetivo deste trabalho passa por criar um cadeado inteligente, de uso pessoal, para bicicletas. Deste modo, é necessário assegurar ao utilizador, que o sistema de bloqueio oferece uma garantia de que o veículo em que está instalado não corre risco de furto com a utilização deste equipamento.

Idealmente, um cadeado de bicicleta inteligente deveria reconhecer o seu utilizador e bloquear ou desbloquear o veículo conforme este se afasta ou aproxima, desresponsabilizando o proprietário desta tarefa. Além disso, este sistema, deveria ter formas autónomas de lidar com tentativas de furto e um sistema de alerta do utilizador aquando de alguma atividade suspeita, solicitando a sua atenção apenas em situações limite.

2.3 *Benchmarking*

Neste sub-capítulo serão apresentados os cadeados inteligentes já existentes no mercado e as suas caracteristicas principais.

2.3.1 Noke U-Lock

Esta opção de bloqueio, Figura 2.1, permite desbloquear a bicicleta remotamente ou no próprio cadeado. Remotamente, este cadeado é bloqueado pelo utilizador aquando da

montagem do equipamento no local desejado e desbloqueado quando o utilizador pressionar um botão presente na lateral do cadeado e possuir um *smartphone* reconhecido pelo sistema de bloqueio através de *bluetooth*. Caso o utilizador não possua um *smartphone* ou este tenha ficado sem bateria é possível desbloquear o cadeado por meio de um código pessoal de toques, longos ou curtos, na lateral do cadeado. Em caso de tentativa de furto, isto é, caso seja detectado um movimento do cadeado por mais de 3 segundos, é emitido um alarme de 105 dB durante 30 segundos. Este produto é resistente a água e poeiras, possui materiais especiais de modo a evitar sabotagens, dispõe de um módulo `GPS` que regista a última atividade do cadeado e baterias de lítio capazes de alimentar o sistema por, pelo menos, 1 ano. Este dispositivo pesa 1474 g e encontra-se à venda no mercado por cerca de 149.99 € [6].

Figura 2.1: Modelo Noke U-Lock em utilização. [6]

2.3.2 Bisecu

Este dispositivo possui um *design* inovador e, para ser utilizado, necessita de ser montado no cubo dianteiro da bicicleta, Figura 2.2. Uma vez montado, não existe necessidade de desmontar ao longo da sua utilização, uma vez que pode ser carregado na própria bicicleta através de uma entrada micro-USB. Apesar do seu design ser um ponto forte é também problemático uma vez que, devido a isso, este dispositivo não é compatível com todos os tipos de bicicleta possuindo imensas restrições dimensionais e geométricas. Durante o seu funcionamento, este aparelho não necessita de qualquer intervenção humana, isto é, através de *bluetooth* este sistema de bloqueio deteta se o seu utilizador está afastado ou próximo, bloqueando e desbloqueando a bicicleta, respetivamente. Claro que não é obrigatória a utilização deste sistema por *bluetooth* existindo a possibilidade de bloquear e desbloquear o cadeado através da aplicação disponibilizada pelo fabricante.

Através desta aplicação é também possível verificar o estado do cadeado, ou seja, se está aberto ou fechado, partilhar a bicicleta e monitorizar as estatísticas recolhidas pelos quatro sensores presentes neste sistema de bloqueio como por exemplo, distância percorrida, velocidade média, calorias queimadas e etc. É também através destes quatro sensores (um giroscópio, dois acelerómetros e um velocímetro) que são detetadas as tentativas de furto e a atividade da bicicleta. Ao identificar uma tentativa de sabotagem ou roubo este sistema de bloqueio envia uma notificação ao utilizador e emite um alarme de 105 dB. Além disso o seu sistema de fecho tem uma arquitetura que evita tentativas de corte ou serra do fecho, uma vez que este está protegido pela estrutura não sendo sequer visível. Este cadeado é resistente a humidade e poeiras, possui uma autonomia de 6 meses e encontra-se no mercado por 169.99 € [7].

Figura 2.2: Bisecu montado no cubo frontal de uma bicicleta comum [7].

2.3.3 Sentinel

Esta alternativa, Figura 2.3 necessita de ser aparafusada à parte traseira do quadro e, o seu mecanismo bloqueia a roda traseira. Este sistema de bloqueio precisa de intervenção humana para ser bloqueado no entanto, o seu desbloqueio pode ser feito a partir da aplicação disponibilizada pelo fabricante ou através de um porta-chaves `RFID`. Além disso admite uma garantia adicional por meio de um acoplamento de corrente de segurança ao próprio cadeado. Possui um alarme de 100 dB que é acionado quando é detetada uma tentativa de sabotagem ou roubo e permite, através de `GPS` localizar a bicicleta e monitorizar a sua posição. Possibilita, também, a partilha do velocípede e da sua localização. Em caso de acidente é enviada, através da aplicação, uma notificação ao utilizador informando-o do mesmo. Este dispositivo permite que o utilizador guarde dados de manutenção da bicicleta, possui uma autonomia de 4 meses e encontra-se a venda no mercado por 192.00 € [8].

Figura 2.3: Dispositivo de bloqueio Sentinel [8].

2.3.4 LINKA

Este cadeado, `LINKA Original` (Figura 2.4) tem um funcionamento automático, tanto no bloqueio como no desbloqueio e, à semelhança do `Sentinel` bloqueia a roda traseira precisando de ser acoplado ao quadro. O seu modo de funcionamento é idêntico ao do `Bisecu`, ou seja, através da intensidade de sinal *bluetooth* é detetada a proximidade do utilizador e, consoante o seu valor, é bloqueado ou desbloqueado o cadeado. Este sistema pode ser desativado através da aplicação do fabricante e, nesse caso, o fecho e a abertura do mecanismo de bloqueio é feito através da ordem do utilizador pela interface da aplicação.Este dispositivo admite a utilização de uma corrente para segurança adicional e aquando de uma tentativa de furto emite um alarme na ordem dos 110 dB. É também possível partilhar o velocípede com família e amigos através da aplicação e em caso de falta de bateria no *smartphone* o cadeado pode ser desbloqueado através de um código pessoal previamente configurado. Este sistema pesa cerca de 650 g, é resistente a água, possui uma autonomia de até 16 meses e está disponível para venda ao público por 169.00 € [9].

Esta marca possui também, em alternativa ao modelo original apresentado no parágrafo anterior, o `LEO` (Figura 2.4) cujo funcionamento é semelhante ao primeiro modelo no entanto, comparativamente a este, apresenta alguns extras tais como: sistema de rastreamento através de `GPS`; comunicação com o utilizador não apenas por *bluetooth* mas também por *GSM/GPRS*, permitindo um alcance global; e um aumento da autonomia para um máximo de 2.5 anos. Este cadeado pode ser utilizado em *e-bikes* e em bicicletas tradicionais sendo, por isso, um pouco maior do que o cadeado abordado no parágrafo anterior. O sistema de alarme teve um pequeno decremento de decibéis, quando comparado com o modelo original, passando de 110 dB para 90 dB e está disponível no mercado por 269.00 € [9].

Figura 2.4: Modelos `LINKA LEO` à esquerda e `LINKA Original` à direita [9].

2.3.5 Ellipse

Esta solução, Figura 2.5, possui um design e modo de funcionamento muito idênticos ao da opção `Noke U-Lock`, ou seja, o seu formato em "U"e o seu sistema fecho duplo destrancado através de um toque do utilizador com um *smartphone* conhecido pelo mecanismo de bloqueio por meio de *bluetooth* ou, em alternativa, por via de um código pessoal a inserir no seu sensor de toque capacitivo. No entanto, possui um modo de carregamento distinto dos outros cadeados inteligentes do mercado, através de painéis solares presentes no seu exterior aumentando a sua autonomia. Contudo, é também possível carregá-lo através de `micro-USB`. Este sistema admite a partilha do velocípede através da aplicação do fabricante, onde também é possível consultar a localização da bicicleta e o seu estado (bloqueado ou desbloqueado). Através de um acelerómetro, cuja sensibilidade pode ser ajustada na aplicação disponibilizada, o cadeado deteta qualquer tentativa de furto e notifica o utilizador da ocorrência. Por meio deste sensor é também detetado qualquer acidente e, nesse caso, os "contactos de emergência"previamente selecionados pelo utilizador, são notificados do incidente. Este sistema de bloqueio é resistente a água, possui um peso de 1134 g e esta disponível apenas nos Estados Unidos por 199 $ [10].

2.3.6 LOCK8

Este produto (Figura 2.6), à semelhança dos já apresentados, dispensa a utilização de qualquer tipo de chave tradicional ao invés disso utiliza uma aplicação onde o utilizador pode bloquear ou desbloquear a bicicleta. No entanto, além de ser fixo num este sistema possui um sistema passivo e ativo de bloqueio, onde o sistema passivo se baseia

Figura 2.5: Cadeado inteligente `Ellipse` [10].

apenas na deteção de qualquer tentativa de furto, através do acelerómetro instalado, e consequente acionamento de alarme de 120 dB e notificação do utilizador por `SMS`, *e-mail* ou na aplicação; e um sistema ativo o qual consiste num sistema parecido com um sistema tradicional de corrente que necessita de um objeto externo, como um poste ou uma árvore, e cujas extremidades são bloqueadas pelo cadeado inteligente. Assim, através da aplicação disponibilizada, o utilizador pode bloquear ou desbloquear a corrente sem necessidade de uma chave tradicional. As deteções de tentativas de furto são efetuadas por meio de um conjunto de sensores como um acelerómetro, um termómetro (no caso de tentativas de congelamento ou incêndio do cadeado ou da corrente), um localizador `GPS` e um detetor de corte da corrente. Com esta alternativa é também possível partilhar o velocípede ou até alugá-lo. A sua bateria pode ser carregada através de uma porta `USB`, contudo não existirá grande necessidade da sua utilização uma vez que este cadeado possui um sistema de indução magnética, que através de uns refletores magnéticos que podem ser adicionados à roda traseira, recarregarão o cadeado durante a utilização da bicicleta. Este cadeado é leve quando comparado aos anteriormente expostos, apresentado um peso de 350 g,excluindo a corrente, e encontra-se à venda por 199 $ [11].

2.3.7 BITLOCK

Esta opção disponível do mercado, a semelhança das apresentadas anteriormente, tem como premissa principal pôr de lado as chaves tradicionais. Para isso, tem implementado um sistema de deteção de *bluetooth* que, com a aproximação do utilizador seguido de um toque deste, desbloqueia o cadeado permitindo a utilização do velocípede. Em caso de falta de bateria no *smartphone* é possível desbloquear o cadeado através de um código pessoal de 4 dígitos. Este cadeado tem uma forma em "U"idêntica ao `Noke U-Lock` e ao `Ellipse`. Para a sua devida utilização não necessita de ser fixo à bicicleta no entanto o

Figura 2.6: Cadeado inteligente `Lock8` montado numa bicicleta comum [11].

sitio de bloqueio precisa de ser estrategicamente escolhido no velocípede pelo utilizador podendo ser necessário um elemento externo, como um poste [12].

Este dispositivo admite *bike sharing* através de uma aplicação para *smartphone* onde é também possível partilhar e consultar a posição do sistema de bloqueio por meio de um módulo de `GPS` neste inserido. Os fabricantes deste sistema tem plena confiança na solidez do seu mecanismo, construído com aço endurecido, e por isso não apresenta soluções alternativas à ocorrência de uma tentativa de furto além da consulta da sua localização. Esta alternativa é resistente a água, possui uma bateria com autonomia até 5 anos e está presente no mercado por 129 $ [12].

Figura 2.7: Cadeado inteligente `BITLOCK` [12].

2.3.8 I LOCK IT

Este modelo possui uma forma e funcionamento semelhante aos modelos da `Linka` apresentados anteriormente. Para o seu devido funcionamento é necessário aparafusá-lo à parte traseira quadro, tendo sido este desenhado para bloquear a roda traseira. Este cadeado inteligente possui um modo automático que deteta o afastamento e a aproxima-

ção de um *smartphone* conhecido por meio da intensidade do sinal *bluetooth* e, em caso de falta de bateria no telemóvel, um modo manual que recorre a um código de cores pré-configurado pelo utilizador. Neste sistema de bloqueio estão presentes um acelerómetro, para detetar eventuais tentativas de furto; um alarme de 110 dB ; uma bateria de lítio com autonomia de até 3 meses; e é suportada a utilização de uma corrente que garantirá uma segurança adicional ao velocípede. Pela aplicação é possível partilhar a bicicleta e monitorizar o estado do cadeado, alem disso,em cenário de furto, é acionado o alarme e notificado o utilizador através desta mesma aplicação. Este dispositivo é construído em aço endurecido e está disponível por 99 € [13].

Figura 2.8: Cadeado inteligente `I LOCK IT` em utilização numa bicicleta convencional [13].

2.3.9 KADALOCK

Esta alternativa possui uma solução distinta das já apresentadas. Está projetada para ser montada num suporte que acompanha o produto e que pode ser utilizado para guardar uma garrafa de água, Figura 2.9. Deve ser montado onde normalmente está um suporte parecido, que apenas serve para guardar bidões de água. O seu funcionamento depende na totalidade do utilizador, quer para puxar um fio de aço colocando-o na posição de fecho, fazendo-o passar pelo meio da roda; quer para o abrir. No entanto o bloqueio ou desbloqueio desse mesmo fio apenas acontece no afastamento ou aproximação de um utilizador, respetivamente, reconhecido pelo sistema através de *bluetooth*. O fio de aço, responsável pela proteção do velocípede, está integrado num sistema anti-furto que aciona o alarme e notifica o utilizador quando este é cortado ou tencionado ou quando a bicicleta é movida (por meio de um acelerómetro). Este cadeado é a prova de água, possui um peso que ronda as 350 g e encontra-se a venda no mercado por 199 $ [14].

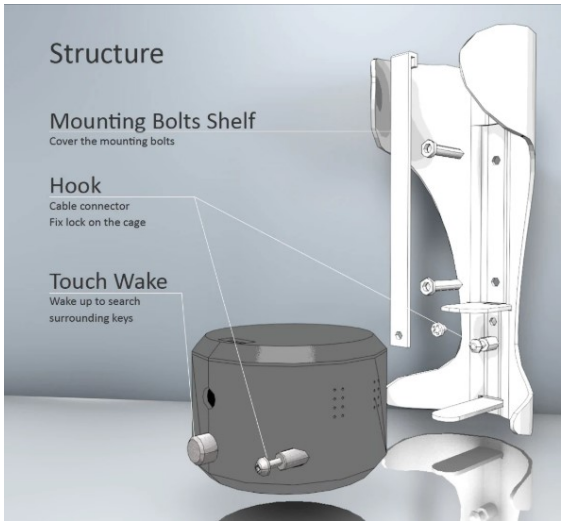


Figura 2.9: Cadeado inteligente KADALOCK e suporte [15].

Figura 2.10: Cadeado inteligente KADALOCK montado apropriadamente numa bicicleta de passeio [15].

2.3.10 Deeperlock

Esta opção oferece um sistema de bloqueio que precisa de ser fixo a bicicleta. Possui uma forma e funcionamento semelhantes aos modelos `Linka` e `I LOCK IT` dispondo de um modo automático que bloqueia e desbloqueia o sistema segundo a proximidade do utilizador [16].

Alternativamente pode monitorizar-se e alterar-se o estado do cadeado numa aplicação para *smartphone* ou por meio de um código pessoal. Qualquer tentativa de furto é detetada por meio de um acelerómetro embutido, notificando imediatamente o utilizador via `GSM` e acionando um alarme de 110 dB. Para além disto, permite ao utilizador consultar a qualquer momento a posição do velocipede [16].

2.4 Análise de *Benchmarking*

Cadeado	Montagem	Bloqueio	Compatibilidade	Desbloqueio	Sistema antifurto	GPS	Carregamento	Autonomia	Extras	Preço
Noke U-Lock	Fácil	Externo à bicicleta	Com todas as bicicletas	Remoto (Bluetooth) No cadeado	✓	✓	Baterias de lítio	12 meses	Resistente à água e poeiras	149.99 €
Bisecu	Cubo dianteiro Difícil	Roda dianteira	Dependa da dimensão e geometria da bicicleta	Remoto (Bluetooth e aplicação)	✓	✓	Micro-USB	6 meses	Partilha da bicicleta Monitorização de estatísticas	169.99 €
Sentinel	Aparafusado ao quadro traseiro	Roda traseira	Elevada	Remoto (app, RFID)	✓	✓		4 meses	Partilha da bicicleta Guarda dados de manutenção	192.00 €
LINKA Original	Acoplado ao quadro	Roda traseira	Elevada	Remoto (Bluetooth e aplicação) No cadeado	✓			16 meses	Partilha da bicicleta Resistente à água	169.00 €
LINKA LEO	Acoplado ao quadro	Roda traseira	Elevada com E-bikes e bicicletas tradicionais	Remoto (Bluetooth, GSM/GPRS e aplicação) No cadeado	✓	✓		30 meses	Partilha da bicicleta Resistente à água	269.00 €
Ellipse	Fácil	Externo à bicicleta	Com todas as bicicletas	Remoto (Bluetooth) No cadeado	✓	✓	Micro-USB Painéis solares		Partilha da bicicleta Notificação em caso de incidente Resistente à água	199.00 $
LOCK8	Roda	Externo à bicicleta	Elevada	Remoto (aplicação)	✓	✓	Sistema de indução magnética USB		Notificação por SMS em caso de tentativa de furto	199.00 $
BITLOCK	Fácil	Externo à bicicleta	Com todas as bicicletas	Remoto (Bluetooth) No cadeado	✓	✓		60 meses	Partilha da bicicleta Construção em aço	129.00 $
I LOCK IT	Aparafusado ao quadro traseiro	Roda traseira	Elevada	Remoto (Bluetooth) No cadeado	✓		Baterias de lítio	3 meses	Partilha da bicicleta Construção em aço endurecido	99.00 €
KADALOCK	Suporte para a água			Manual + Bluetooth	✓				Resistente à água	199.00 $
Deeperlock	Acoplado ao quadro	Roda traseira		Remoto (GSM, Aplicação) No cadeado	✓	✓				

Tabela 2.1: Resumo das características principais de cada cadeado analisado no *Benchmarking*

Da comparação efetuada entre os diferentes cadeados disponíveis no mercado verifica-se que a maioria é de fácil montagem e compatível com grande parte das bicicletas. É de notar que cadeados como o `Noke U-Lock`, `Ellipse` e `BITLOCK` apesar de serem de fácil montagem apresentam uma desvantagem que é a sua necessidade de um objeto externo para o seu bloqueio. Grande parte dos cadeados possui desbloqueio remoto e no próprio cadeado e a comunicação entre o cadeado e o utilizador é efetuada principalmente por Bluetooth limitando o alcance da comunicação.

Nesta característica destaca-se o cadeado LINKA Leo que também efetua comunicação por `GSM/GPRS` permitindo um alcance global. Todos os cadeados apresentam um sistema antifurto com alarme sonoro e alguns ainda notificam o utilizador na tentativa de roubo e da localização da bicicleta. Na autonomia, mais uma vez, enfoca-se o cadeado

`LINKA Leo`, com uma autonomia esperada de 2.5 anos. Os principais extras presentes nos cadeados são a resistência à água, permitir a localização da bicicleta, notificações em caso de tentativa de furto ou acidente e permitem a partilha do veículo com terceiros.

Idealmente o cadeado a ser desenvolvido combina as melhores características de cada um dos cadeados do mercado e resolve o maior problema destes, o preço .

Deverá ser de fácil montagem, com elevada compatibilidade entre bicicletas e não necessitar de um objeto externo para a bloquear. A comunicação entre o cadeado e o utilizador deve ser por Bluetooth e `GSM/GPRS` para permitir um alcance global. O desbloqueio da bicicleta deve ser remoto, desbloqueando o cadeado com a aproximação do utilizador e ter um desbloqueio manual, no caso de falta de bateria. A aplicação desenvolvida deve permitir o bloqueio e desbloqueio da bicicleta permitindo a partilha da mesma com terceiros, monitorizar estatísticas como a distância percorrida, velocidade média e calorias queimadas e localizar a bicicleta por `GPS`. O utilizador deve ser notificado quando ocorrem tentativas de furto e o cadeado deve emitir um alarme antirroubo. Todo este mecanismo deverá ser construído com componentes electrónicos de preço reduzido e com materiais que respondam às necessidades físicas do uso normal do cadeado, mas que não aumentem significativamente o seu valor.

Capítulo 3

Necessidades do Utilizador e Seleção de Conceito

3.1 Introdução

Neste capítulo identificam-se as necessidades do utilizador e a sua ordem de importância utilizando para isso o diagrama de Mudge. Posteriormente, identificaram-se as características críticas para o cliente e especificou-se a correlação entre as necessidades do utilizador e os requisitos do projeto através da *Quality Function Deployment* (`QFD`). Em seguida, foram comparadas várias alternativas de cada especificação do projeto como sistemas de bloqueio, tipos de aplicações e meios de comunicação entre o cadeado e o utilizador sendo selecionada a mais conveniente de forma a gerar o conceito do cadeado.

3.2 Identificação das necessidades dos utilizadores

No desenvolvimento de qualquer produto é importante conhecer as necessidades de quem o vai utilizar, uma vez que é para estes que o produto está a ser criado. Assim, ter as necessidades do cliente bem presentes durante a etapa de geração de conceito é algo fundamental. Ao avaliar os produtos disponíveis no mercado identificaram-se as principais características mandatórias, isto é, essenciais a este produto do ponto de vista do cliente; esperadas, ou seja, que o cliente espera ou deseja ver executadas no produto; e atrativas que, não sendo características essenciais, acrescentam valor ao produto e surpreendem o utilizador.

3.2.1 Diagrama de Mudge

O diagrama de Mudge é uma ferramenta que permite comparar um conjunto de características duas a duas, estabelecendo uma ordem de relevância entre elas [17]. As características que foram comparadas recorrendo ao diagrama de Mudge foram a eficiência de bloqueio da bicicleta (A), o design do cadeado (B), o tempo de fecho (C), possuir fecho automático (D), a facilidade de montagem (E), possuir alarme (F), possuir GPS (G), possuir aplicação móvel (H), monitorizar atividade (I), autonomia (J), preço (K), peso (L) e possibilidade de partilha do acesso ao cadeado (M). Para comparação das características, definiu-se a seguinte escala de pontuação:

Muito mais importante – 5
Mais importante – 3
Pouco mais importante – 1
Igual importância – 0

Na Figura 3.1 apresenta-se o diagrama de Mudge do projeto de desenvolvimento de um cadeado com fecho automático.

A	B	C	D	E	F	G	H	I	J	K	L	M	TOTAL	%
A	A5	A5	A5	A5	A5	A5	A5	A5	A5	A5	A5	A5	60	38.22
B		C1	D1	B1	F3	G3	H3	0	0	K1	0	M1	1	0.64
C			D1	C1	F3	G3	H3	C1	0	K1	C1	M1	4	2.55
D				D1	F3	G3	H3	D1	0	D1	D1	0	6	3.82
E					F3	G3	H3	I1	J1	K1	L1	M1	0	0.00
F						G1	F1	F3	F1	F1	F3	F3	24	15.29
G							G1	G3	G1	G1	G3	G3	25	15.92
H								H3	H1	H1	H1	H1	19	12.10
I									J1	K1	I1	M1	2	1.27
J										J1	J1	0	4	2.55
K											K1	M1	5	3.18
L												M1	1	0.64
M													6	3.82
TOTAL													157	100

A	Eficiência de bloqueio da bicicleta	H	Possuir app
B	Design do cadeado	I	Monitorizar atividade (velocidade média, distância percorrida)
C	Tempo de fecho	J	Autonomia
D	Fecho automático	K	Preço
E	Facilidade de montagem	L	Peso
F	Possuir alarme	M	Possibilidade de partilha de acesso ao cadeado
G	Possuir GPS		

5	Muito mais importante
3	Mais importante
1	Pouco mais importante
0	Igual importancia

Figura 3.1: Diagrama de Mudge

O diagrama de Mudge permitiu a construção da Tabela 3.1 com a ordenação por ordem decrescente do grau de relevância de cada uma das características especificadas anteriormente para o cadeado. Desta forma é possível, concluir, por exemplo que o cliente não se importa de pagar mais desde que o cadeado garanta uma maior segurança à bicicleta, com um design atrativo e automático. Acredita-se que esta preferência do cliente não está na desvalorização do fator preço mas da valorização do fator segurança e interatividade uma vez que para porcurar este tipo de mercado o cliente abandonou o mercado dos cadeados tradicionais onde são praticados preços significativamente mais baixos .

3.3 Especificações técnicas

Conhecidas as necessidades do cliente é importante definir-se de que forma se dará resposta a essas necessidades e com que especificações do produto as conseguiremos colmatar, por exemplo, no caso da eficiência de bloqueio, esta será tanto melhor quanto melhores os materiais usados no cadeado e quanto melhor for o mecanismo de bloqueio. Além disso, pode também ser percebido que especificação é dispensável do produto, quando esta não responde a nenhuma das necessidades do cliente. Este estudo foi feito por meio da matriz QFD apresentada na secção seguinte.

Posição	Característica	Descrição
1	A	Eficiência de bloqueio da bicicleta
2	G	Possuir GPS
3	F	Possuir alarme
4	H	Possuir app
5	M	Possibilidade de partilha de acesso ao cadeado
5	D	Fecho automático
7	K	Preço
8	C	Tempo de fecho
8	J	Autonomia
10	I	Monitorizar atividade
11	B	Design do cadeado
11	L	Peso
13	E	Facilidade de montagem

Tabela 3.1: Necessidades do cliente por ordem decrescente de relevância

3.3.1 Matriz QFD (*Quality Function Deployment*)

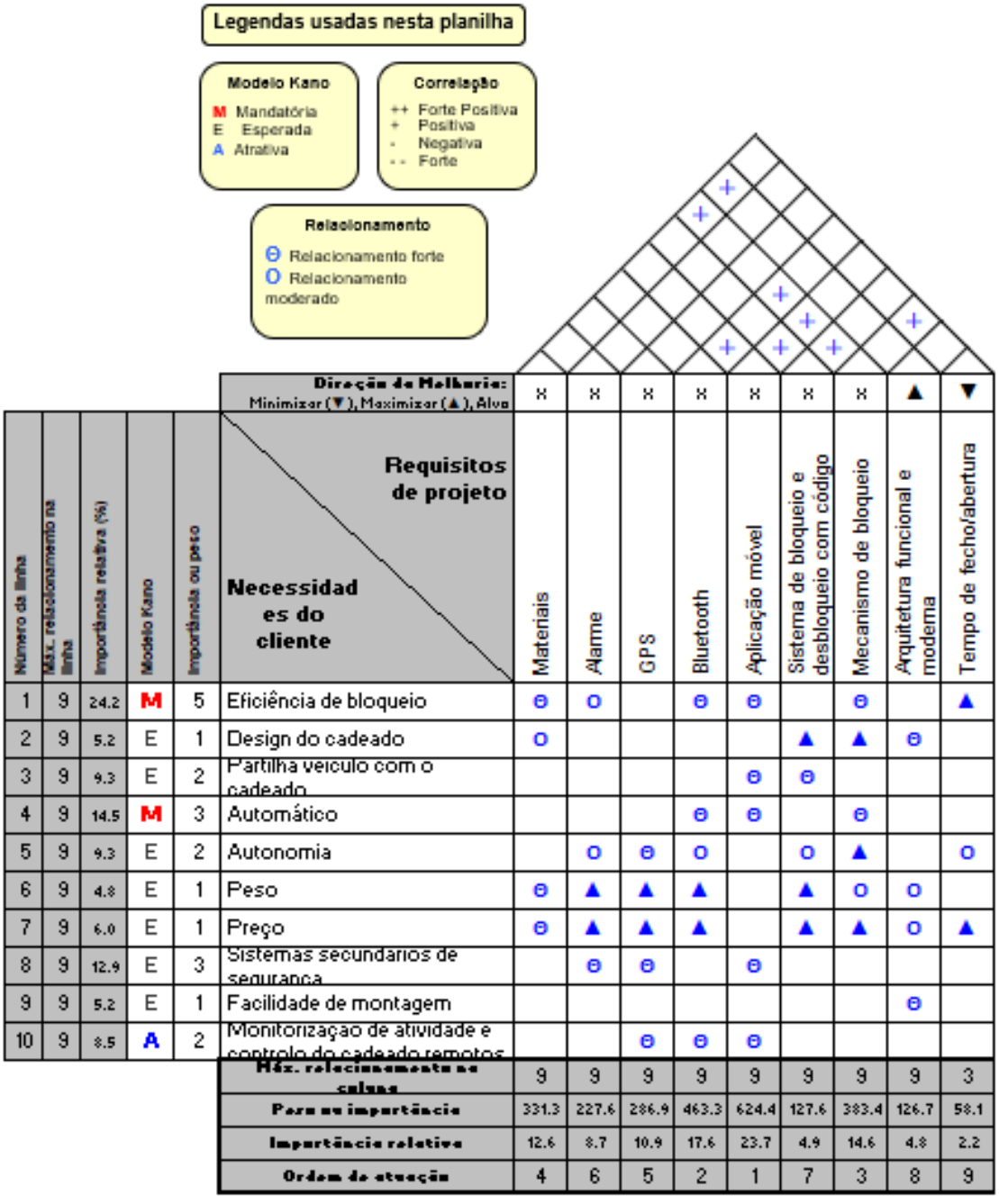
Quality Function Deployment (QFD) é um método utilizado para identificar as características críticas para o cliente e especificar a correlação entre as necessidades do cliente e os requisitos de projeto [17]. As necessidades do cliente listadas por ordem decrescente de importância são a eficiência de bloqueio, ser automático, possuir sistemas secundários de segurança, permitir a partilha da bicicleta com o cadeado, ter elevada autonomia, fazer monitorização de atividade e controlo remoto do cadeado, preço, peso e design do produto. Destas necessidades a eficiência de bloqueio e ser automático são características mandatórias, a monitorização de atividade e controlo remoto do cadeado é uma característica atrativa e as restantes necessidades são esperadas pelo usuário. Os requisitos de projeto são os materiais, possuir alarme, possuir GPS, comunicação por Bluetooth, possuir aplicação móvel, possuir sistema de bloqueio e desbloqueio com código, mecanismo de bloqueio, possuir uma arquitetura funcional e moderna e tempo de fecho e abertura mínimo. Relacionaram-se as necessidades do cliente com os requisitos do projeto identificando-se o grau de relação entre elas (relacionamento forte, moderado ou fraco) e construi-se o QFD deste projeto apresentado na Figura 3.2.

Da construção do QFD conclui-se que a ordem decrescente de atuação é possuir aplicação móvel, Bluetooth, mecanismo de bloqueio, materiais, GPS, alarme, sistema de bloqueio e desbloqueio com código, arquitetura funcional e moderna e tempo de fecho/abertura.

3.4 Geração e Seleção de Conceitos

Nesta secção será decidido o rumo que o cadeado a desenvolver irá tomar. No entanto, antes, é importante definir alguns conceitos:

- **Bluetooth/ Bluetooth Low Energy**- O Bluetooth começou a ser desenvolvido em 1998 pelo Bluetooth Special Industry Group (SIG), formado pela Ericsson, IBM, Intel, Nokia e Toshiba, com o objetivo de criar uma tecnologia sem fios que



Número da linha	Máx. relacionamento na linha	Importância relativa (%)	Modelo Kano	Importância ou peso	Necessidades do cliente / Requisitos de projeto	Materiais	Alarme	GPS	Bluetooth	Aplicação móvel	Sistema de bloqueio e desbloqueio com código	Mecanismo de bloqueio	Arquitetura funcional e modema	Tempo de fecho/abertura
					Direção de Melhoria: Minimizar (▼), Maximizar (▲), Alvo	x	x	x	x	x	x	x	▲	▼
1	9	24.2	M	5	Eficiência de bloqueio	Θ	O		Θ	Θ		Θ		▲
2	9	5.2	E	1	Design do cadeado	O					▲	▲	Θ	
3	9	9.3	E	2	Partilha veículo com o cadeado					Θ	Θ			
4	9	14.5	M	3	Automático				Θ	Θ		Θ		
5	9	9.3	E	2	Autonomia		O	Θ	O		O	▲		O
6	9	4.8	E	1	Peso	Θ	▲	▲	▲		▲	O	O	
7	9	6.0	E	1	Preço	Θ	▲	▲	▲		▲	▲	O	▲
8	9	12.9	E	3	Sistemas secundários de segurança		Θ	Θ		Θ				
9	9	5.2	E	1	Facilidade de montagem								Θ	
10	9	8.5	A	2	Monitorização de atividade e controlo do cadeado remotos			Θ	Θ	Θ				
					Máx. relacionamento na coluna	9	9	9	9	9	9	9	9	3
					Peso ou importância	331.3	227.6	286.9	463.3	624.4	127.6	383.4	126.7	58.1
					Importância relativa	12.6	8.7	10.9	17.6	23.7	4.9	14.6	4.8	2.2
					Ordem de atuação	4	6	5	2	1	7	3	8	9

Figura 3.2: Matriz QFD

permitisse a comunicação a curto alcance entre dispositivos, substituindo os cabos de dados RS-232. Esta tecnologia de baixo custo permite a troca de dados, sem fios, entre dispositivos fixos e móveis em distâncias curtas, utilizando ondas de rádio UHF na banda ISM (industrial, scientific, medical) dos 2.4 GHz [18].

O Bluetooth Low Energy, tal como o Bluetooth, é uma tecnologia sem fios de personal area network (PAN) que permite a comunicação, sem fios, entre dispositivos, com um consumo energético mais reduzido que o Bluetooth, mas com um alcance de comunicação semelhante. Os dispositivos que utilizam esta tecnologia de baixo consumo possuem uma autonomia de meses ou anos, sem necessidade de carregamento ou troca de bateria [19].

- **GSM**- O Global System for Mobile Communications (GSM) foi desenvolvido na Finlândia em 1991 pelo European Telecommunications Standards Institute (ETSI) para descrever os protocolos para redes móveis digitais de segunda geração (2G) utilizado em dispositivos móveis, como telemóveis e tablets [20].

- **GPRS**- O General Packet Radio Services (GPRS) foi desenvolvido pela European Telecommunications Standards Institute (ETSI). Esta tecnologia de comunicações móveis baseia-se na transferência de dados em pacotes sobre a rede GSM, permitindo uma maior velocidade na transmissão de dados [21].

- **Microcontrolador**- Um microcontrolador é um circuito integrado compacto desenhado para controlar operações específicas e, normalmente, possui um processador, memória e periféricos de entrada/saída. É um dispositivo de tamanho reduzido e baixo consumo energético, utilizado em dispositivos automóveis, médicos, brinquedos, etc. O Esp-32 é um exemplo de microcontrolador de pequenas dimensões que possui protocolos de Bluetooth (Bluetooth v4.2 BR/EDR e BLE) e WiFi (802.11 b/g/n) e funciona numa vasta gama de temperaturas (-40ºC a 85ºC) com um consumo médio de 80 mA [22].

- **PWA**- Uma aplicação PWA (Progressive Web App) é uma aplicação escrita dentro de uma página web, utilizando linguagens como HTML, JavaScript e CSS. O custo de desenvolvimento de uma aplicação PWA é muito menor porque não é necessário criar uma aplicação para cada sistema operativo (Android e iOS) e submetê-la em diferentes App Stores. As PWA podem ser acedidas em qualquer dispositivo que possua um browser, sendo compatíveis com a maioria dos browsers mais utilizados (Chrome, Safari, Firefox e Edge) [23].

- **Parafuso sem-fim**- As engrenagens de parafuso sem-fim são um tipo de engrenagens helicoidais constituídas por um parafuso sem-fim e uma coroa, onde a movimentação circular do parafuso, faz movimentar a coroa. Uma das principais aplicações deste tipo de engrenagem é nos redutores de velocidade, em que as relações de transmissão podem atingir valores da ordem dos 100:1 [24].

- **Pinhão-Cremalheira**- Pinhão-cremalheira é um mecanismo composto por uma engrenagem circular dentada (pinhão) e uma engrenagem linear dentada (cremalheira). Este mecanismo transforma movimento rotacional em movimento linear. Ou seja, a rotação do pinhão aciona a cremalheira linearmente e o acionamento da cremalheira linearmente faz o pinhão girar [25].

Na Figura 3.3 é apresentado um gráfico onde são comparadas as taxas de transferência de dados e o alcance de cada tipo comunicação (*Bluetooth*, *Bluetooth Low Energy*, `GSM` e `GPS`) e onde é possível ver as limitações em termos de alcance dos tipos *Bluetooth* e *Bluetooth Low Energy*.

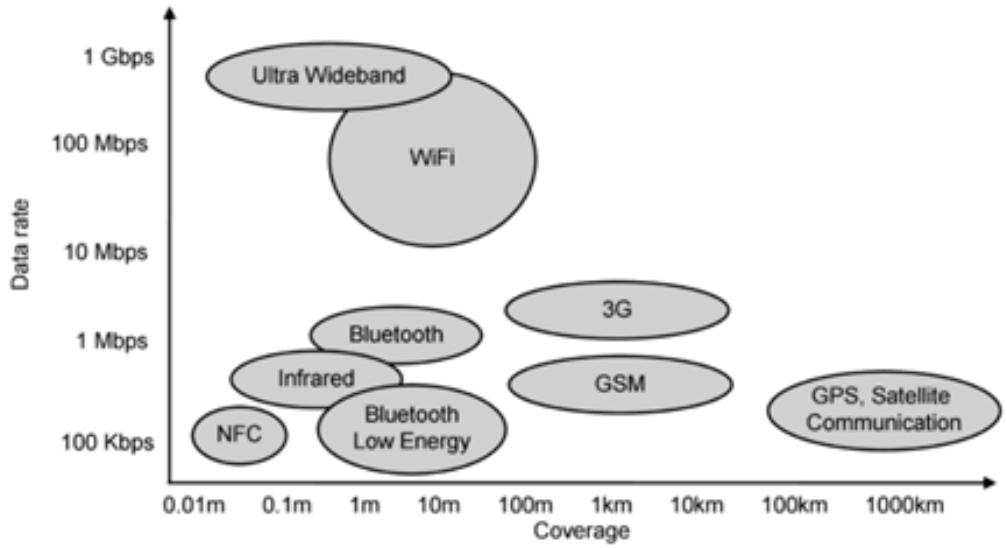


Figura 3.3: Taxa de transferência de dados e alcances típicos de vários tipos de comunicação sem fios [26].

3.4.1 Sistema de bloqueio

Do estudo de mercado efetuado sobre os cadeados de bicicletas, verificou-se que existem três alternativas possíveis para o sistema de bloqueio, apresentadas na Figura 3.4.

A primeira alternativa consiste num sistema de bloqueio manual, onde o utilizador necessita de acionar uma patilha que roda o mecanismo e tranca o cadeado. O desbloqueio do cadeado é automatizado, onde o microcontrolador destranca o cadeado e através de uma mola direciona a patilha à posição inicial. Este sistema de bloqueio é compatível com a maioria das bicicletas, sendo de fácil montagem, no entanto tem a desvantagem de ser manual, precisando da atuação do utilizador no cadeado.

A segunda alternativa é um cadeado automatizado onde o microcontrolador aciona o motor e este faz girar o parafuso sem-fim. Consoante o sentido de rotação do parafuso, sentido horário ou anti-horário, o cadeado é bloqueado ou desbloqueado. Este sistema de bloqueio é de fácil montagem e compatível com a maioria das bicicletas.

A terceira alternativa é um cadeado que utiliza um mecanismo do tipo pinhão cremalheira que é acionado por um microcontrolador, fazendo subir ou descer o perno, que entra no rasgo e bloqueia a roda. Este sistema de bloqueio tem de ser instalado na roda dianteira para evitar a desmontagem de todos os sistemas que estão ligados à roda traseira e não é compatível com todos os tipos de bicicletas.

Das três alternativas apresentadas decidiu-se utilizar um sistema de bloqueio semelhante ao segundo cadeado por este ter um sistema de bloqueio e desbloqueio automatizado, ser de mais fácil montagem do que a terceira alternativa, ser compatível com a

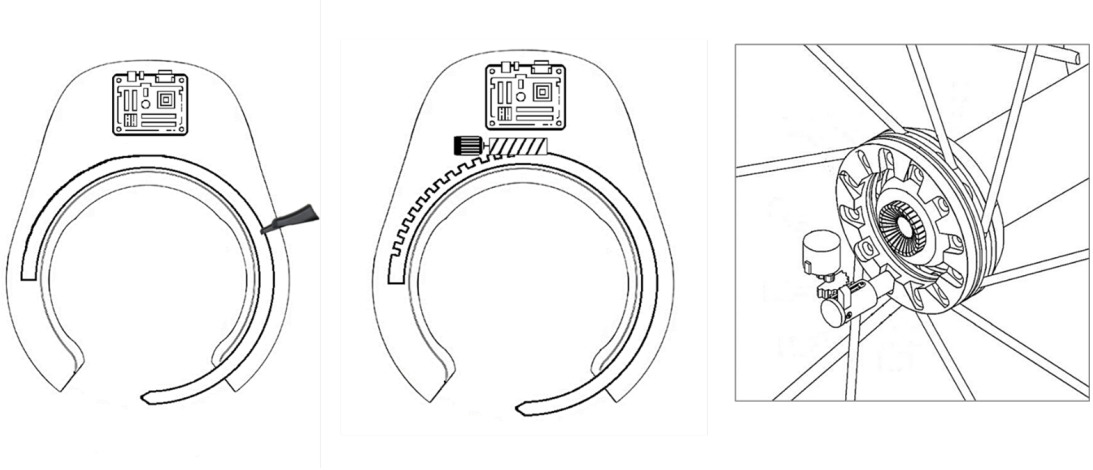


Figura 3.4: À esquerda está representado um cadeado com bloqueio manual; Ao centro um cadeado com mecanismo de bloqueio e desbloqueio automático com parafuso sem-fim; À direita é apresentado um cadeado com mecanismo de bloqueio e desbloqueio automático com pinhão/cremalheira [27].

maioria das bicicletas e não necessitar de nenhum apoio externo.

3.4.2 Aplicação

Pretende-se desenvolver uma aplicação com a função de interface entre o utilizador e o cadeado. Esta aplicação vai permitir ao utilizador bloquear e debloquear o cadeado, aceder à localização da bicicleta, ser alertado em caso de tentativa de furto, permitir a partilha da bicicleta com terceiros e monitorizar a atividade como a distância percorrida, a velocidade média e as calorias queimadas.

Existiam dois tipos de aplicações possíveis de desenvolver: uma aplicação `PWA` (*Progressive Web App*) ou uma aplicação nativa como *Android* ou *iOS*. As aplicações `PWA` são escritas dentro de uma página web, utilizando linguagens como *HTML*, *JavaScript* e *CSS* [19], enquanto as aplicações nativas são desenvolvidas com a linguagem de cada plataforma (*Objective-C* e *Swift* para *iOS* e *Java* para *Android*). Assim, o custo de desenvolvimento de uma aplicação `PWA` é muito menor porque não é necessário criar uma aplicação para cada sistema operativo e submetê-la em diferentes *App Stores*. Além disso, as aplicações `PWA` podem ser acedidas em qualquer dispositivo que possua um *web browser*, sendo compatíveis com a maioria dos *browser* mais utilizados (Chrome, Safari, Firefox e Edge). Deste modo, decidiu-se desenvolver uma aplicação do tipo `PWA`.

O bloqueio e desbloqueio da bicicleta é efetuado pelo acionamento do botão bloqueio/desbloqueio na aplicação, permitindo também a partilha da bicicleta remotamente com terceiros e a localização da bicicleta é identificada no mapa mundo. A velocidade é adquirida a partir de um parâmetro fornecido pelo GPS e as calorias queimadas calculadas com base na velocidade média e tempo do deslocamento e na altura e peso do utilizador. Essas informações poderão ser consultadas num menu especifico na aplicação.

3.4.3 Comunicação cadeado - utilizador

A comunicação entre o utilizador e o cadeado pode ser feita por *Bluetooth* e *Global System for Mobile* (GSM).

O *Bluetooth* apenas permite a comunicação entre o utilizador e o cadeado a curta distância, no entanto possibilita o bloqueio e desbloqueio automático do cadeado recorrendo à intensidade do sinal de *Bluetooth*. Se o sinal vai diminuindo o cadeado é bloqueado, se o sinal vai aumentando o cadeado é desbloqueado. Além disso, não tem custos associados.

O GSM é o sistema de comunicação usado nos telemóveis que utiliza dois conjuntos de frequências na banda dos 900 MHz. A banda dos 890-915 MHz é usada para as transmissões do terminal e a banda dos 935-960 MHz, para as transmissões da rede. Este sistema de comunicação permite, ao contrário do *Bluetooth*, que exista comunicação entre o utilizador e o cadeado globalmente, não dependendo da distância a que se encontram.

Pretende-se implementar estes dois tipos de comunicação entre o cadeado e o utilizador. O GSM vai permitir existir uma comunicação global e recorrendo à intensidade do sinal de *Bluetooth* vai ser possível o cadeado detetar a proximidade ou afastamento do utilizador, bloqueando e desbloqueando a bicicleta consoante esta ação.

3.4.4 Conceito

Tendo em conta os tópicos anteriores deste capítulo pensou-se num mecanismo de bloqueio da roda traseira, com uma forma semelhante à representada ao centro na Figura 3.4, que ligue a roda ao quadro, bloqueando-a. De modo que não seja possível roubar a bicicleta através da desarticulação da roda e eliminando a necessidade de um objeto externo ao veículo para acorrentar a bicicleta, como acontece nos sistemas de bloqueio usuais.

Foi também idealizado um sistema de bloqueio e desbloqueio automático, efetuado por um motor que é acionado por um sinal de *bluetooth*, proveniente do *smartphone* do utilizador, bloqueando a bicicleta quando este se afasta e desbloqueando-a quando este se aproxima; um sistema manual de bloqueio/desbloqueio através de um código pessoal ou uma pulseira que funciona como chave através de RFID, de modo a contornar possíveis falhas como a falta de bateria no telemóvel; a utilização de um acelerómetro responsável por detetar qualquer tentativa de furto quando o cadeado esta bloqueado; um sistema de localização GPS de modo a permitir a fácil localização da bicicleta em caso de furto ou de qualquer outra necessidade por parte do utilizador como, por exemplo, partilhar a localização do veículo com um amigo a fim deste também a poder utilizar. Todo este sistema interage com o utilizador através de uma aplicação para *smartphone* que além de permitir o controlo e a visualização do estado do cadeado (bloqueado/desbloqueado), mostra também estatísticas recolhidas através dos sensores instalados como velocidade máxima, velocidade média, distância percorrida e calorias queimadas, garantindo uma experiência ideal ao utilizador.

Na Figura 3.5 está representado o funcionamento geral do cadeado que estará instalado na bicicleta e poderá ser desbloqueado de três formas: pela aplicação, que estará directamente associada a um *smartphone* e que poderá partilhar o acesso a bicicleta com terceiros; por um código pessoal; e por uma pulseira RFID. Em caso de furto é acionado um alarme, pode ser consultada a posição do cadeado e é enviada uma mensagem a notificar o utilizador.

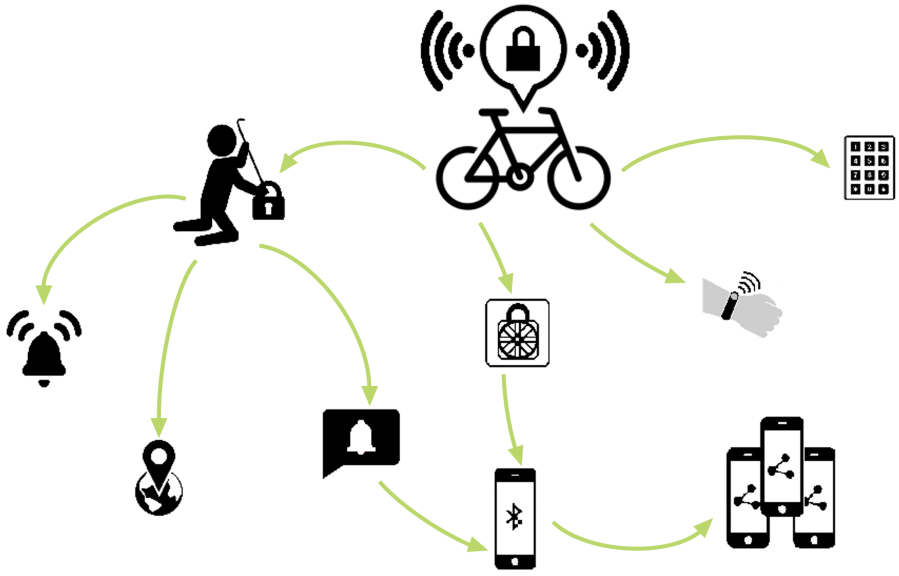


Figura 3.5: Esquema de funcionamento geral do cadeado inteligente

Capítulo 4

Projeto de Automação

Definidos os objetivos do cadeado e as características mais importantes a potencializar procedeu-se ao desenvolvimento da parte eletrónica, que será desenvolvida ao longo deste capítulo, onde serão referidos os componentes e o software utilizados, as suas características e alguns esquemas de montagem. Será também exposta a programação envolvida e o funcionamento da aplicação.

4.1 Solução Proposta

A Figura 4.1, representa a arquitetura do funcionamento da solução proposta. Mostra como os dispositivos interagem entre si, quais as vias de comunicação e com que finalidade. Primeiramente, o módulo GPS envia continuamente, via comunicação em série, frases (NMEA) com as coordenadas e outros valores de interesse para o ESP que, por sua vez, tenta estabelecer uma ligação bluetooth com a PWA de forma e enviar-lhe as frases NMEA provenientes do módulo GPS e a receber ordens de bloqueio e desbloqueio do cadeado. Quando uma dessas ordens é recebida, é desencadeada uma interrupção que acionará o motor no sentido concordante com a ordem recebida.

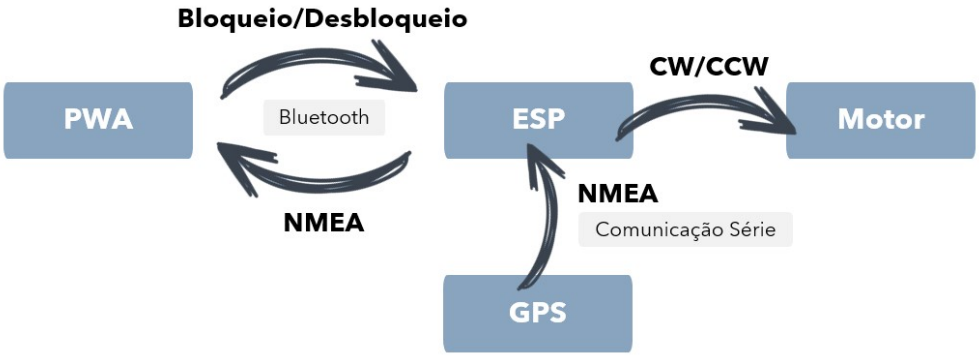


Figura 4.1: Esquema da arquitetura do sistema.

4.2 Componentes

Os componentes electrónicos utilizados neste projeto e algumas das suas características foram enumeradas na Tabela 4.1

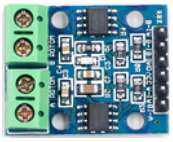
Componente	Descrição
Esp-Wroom-32 	Microcontrolador de pequenas dimensões que possui protocolos de *Bluetooth* (*Bluetooth* v4.2 BR/EDR e BLE) e WiFi (802.11 b/g/n), funciona numa vasta gama de temperaturas (-40 ºC a 85 ºC) e com um consumo médio de 80 mA [28]. **Preço:** 4.14 €
GY-GPS6MV2 	Módulo GPS com uma tensão de alimentação de 3 V a 5 V que possui apenas 4 pinos: dois para alimentação (GND, VCC) e dois para comunicação em série [29]. **Preço:** 3.01 €
L9110 	Driver que admite até 2 motores DC com uma corrente continua máxima de 800 mA e uma tensão de alimentação de 2.5 V a 12 V [30]. **Preço:** 1.18 €
Motor 	Mini motor com redutora, de corrente continua de 3 V e 200 rpm à saída. **Preço:** 2.21 €
Bateria	Aplicou-se uma bateria de 5 V, 2 A e 5000 mAh. **Preço:** 7.99 €

Tabela 4.1: Componentes utilizados e respetivas descrições e preços

4.3 Software e Linguagens de Programação

Ao nível de *software* e linguagens de programação foram utilizados:

- **Arduino IDE (C/C++)**- Arduino Integrated Development Environment (IDE) é uma plataforma open-source para Windows, Linux e macOs, que utiliza linguagem C e C++, permitindo escrever e fazer upload de programas em placas compatíveis com Arduino [31].

- **Visual Studio Code (HTML, CSS, JavaScript)**- O Visual Studio Code é um editor de código simplificado com suporte para operações de desenvolvimento como debugging e execução de tarefas. Foi desenvolvido pela Microsoft para Windows, Linux e macOS e suporta diferentes linguagens de programação como Java, JavaScript, Python e C++ [32].

- **HTML**- HTML (HyperText Markup Language) é uma linguagem de programação criada em 1980 pelo físico Tim Berners-Lee. É utilizada no desenvolvimento de páginas web e utilizada em conjunto com Cascading Style Sheets (CSS) e JavaScript.

- **CSS**- CSS (Cascading Style Sheets) é uma linguagem utilizada para adicional um estilo, como tipo de letra, tamanho, cores, etc. a um documento HTML.

- **JavaScript**- JavaScript é uma linguagem de programação de alto-nível, utilizada no desenvolvimento de páginas Web, juntamente com CSS e HTML.

4.4 Esquema de montagem

O esquema de montagem está representado na Figura 4.2, onde se pode observar que é feita uma alimentação de 5 V ao microcontrolador e ao driver do motor por meio da bateria mencionada no tópico anterior. Esta alimentação ao microcontrolador é possível pois este possui um regulador de tensão quando alimentado ao pino Vin. Através do pino 3V3 do microcontrolador precedeu-se à alimentação do módulo GPS (GY-GPS6MV2) que comunica em serie com o Esp-Wroom-32 por intermédio dos pinos Rx2/Tx2 deste. Os pinos 12 e 13 do microcontrolador foram utilizados como saídas digitais e ligados ao driver do motor nos pinos A-1A e A-1B para ativar a rotação do motor no sentido horário e anti-horário. Para evitar a utilização de sensores ou fins de curso foram também utilizados os pinos capacitivos (32 e 33) do Esp-Wroom-32 que, ao fazerem contacto com uma saliência ligada ao terra na parte funcional do cadeado, indicam ao programa se os limites de abertura e fecho foram atingidos de modo que a rotação do motor seja desabilitada.

4.5 Programação do ESP-32

A programação do microcontrolador foi feita através do Arduino IDE, um programa desenvolvido para facilitar a programação de placas compatíveis com Arduino mas que, através de "board packs" de terceiros, permite também a programação de placas como os Esp-32.

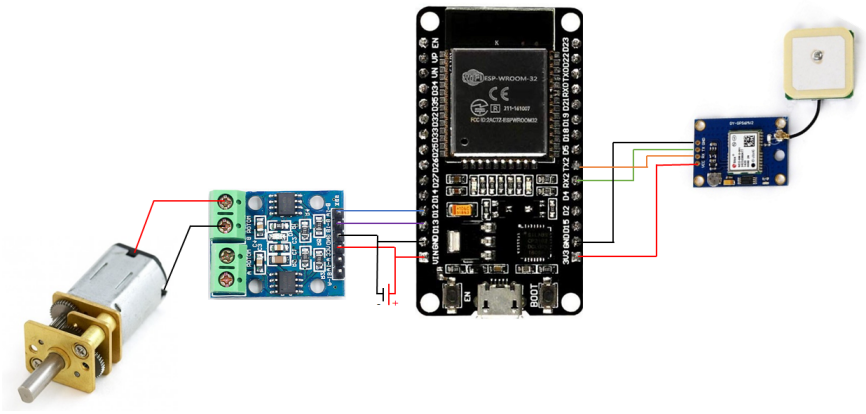


Figura 4.2: Esquema de montagem

4.5.1 Configuração da Placa no Software

Neste caso foi necessário, em primeiro lugar, importar um "board pack" de Esp's-32 fornecido pela Espressif Systems e selecionar no software a placa "ESP32 Dev Module". Depois os parâmetros de transferência de dados devem ser configurados como mostra a Figura 4.3

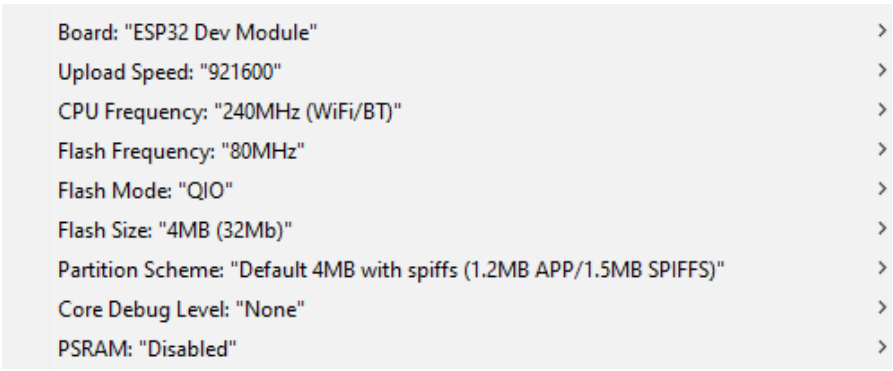


Figura 4.3: Seleção da placa e respetivos parâmetros de comunicação

De modo a facilitar a programação e comunicação dos vários dispositivos utilizados neste projeto tornou-se necessária a importação de algumas bibliotecas como "TinyGPS++" para o tratamento de frases NMEA; "HardwareSerial" e "EEPROM" para comunicação em série; "BLEDevice", "BLEUtils", "BLEServer", "BLE2902" para comunicação *Blueto-oth Low Energy*. A inclusão destas bibliotecas no código original foi feita seguindo a documentação e recomendações do desenvolvedor e, apesar de ter sido despendido muito tempo nessas questões, não parece pertinente desenvolver e explanar essas linhas de código nos próximos tópicos. No entanto, todo o código está disponível para consulta no Anexo A.1.

4.5.2 Configuração de Saídas e Entradas Digitais

Como já referido anteriormente, utilizaram-se apenas os pinos 12 e 13 (para bloquear e desbloquear o cadeado), os pinos 32 e 33 (como sensor de "cadeado bloqueado" e "cadeado desbloqueado") e foi também utilizado um LED já embutido na placa, correspondente ao pino 2, para assinalar quando o motor está em funcionamento. De modo a facilitar a programação e interpretação do código os pinos foram definidos como na Figura 4.4:

```
#define LED_PIN 2
#define Forward 13
#define Reverse 12
#define ClosedSensor 32
#define OpenedSensor 33
```

Figura 4.4: Definição de cada pino utilizado no código

Posteriormente foram declaradas as entradas e saídas digitais e atribuído o valor zero às saídas (Figura 4.5).

```
pinMode(LED_PIN, OUTPUT);
pinMode(Forward, OUTPUT);
pinMode(Reverse, OUTPUT);
pinMode(ClosedSensor, INPUT);
pinMode(OpenedSensor, INPUT);
digitalWrite (LED_PIN, LOW);
digitalWrite (Forward, LOW);
digitalWrite (Reverse, LOW);
```

Figura 4.5: Configuração das saídas e entradas digitais e atribuição do seu valor inicial

Estas linhas de código foram escritas na função de "setup" que é a primeira instrução a ser executada e só o é uma vez. Assim garante-se que, de cada vez que ligamos ou reiniciamos o Esp-32, as saídas que ativam o motor assumem o valor de 0 V e o *Led* está desligado evitando fechos ou aberturas indesejadas. Nesta função têm também de ser iniciados os dois tipos de comunicação necessários ao bom funcionamento do cadeado, *Bluetooth* (para comunicação com outros dispositivos) e série (para comunicação com o módulo GPS).

4.5.3 Rotina Principal

Os principais objetivos da rotina principal são selecionar as mensagens NMEA provenientes do módulo GPS, verificar se algum dispositivo se ligou via *Bluetooth* e, se esta última for verdade, enviar as mensagens NMEA já tratadas e receber os comandos de bloqueio e desbloqueio do cadeado que acionarão uma interrupção responsável por ativar as saídas que acionarão o motor na direção pretendida, Figura 4.6. Idealmente, as mensagens com a localização não deveriam ser enviadas por *Bluetooth* uma vez que este tipo de comunicação é de curta distância e se estamos perto da bicicleta não necessitamos saber a sua posição. Esta informação deveria ser enviada via GSM para uma base

de dados para, posteriormente ser tratada pela aplicação e demonstrada ao utilizador independentemente da distância deste ao veículo. Contudo, não foi possível utilizar uma placa com GSM de modo a permitir esta comunicação e, afim de continuar a programação da *app*, e uma vez que a única alteração a fazer ao código da PWA é alterar a fonte das mensagens NMEA de *Bluetooth* para uma base de dados preferiu-se continuar com este meio de comunicação.

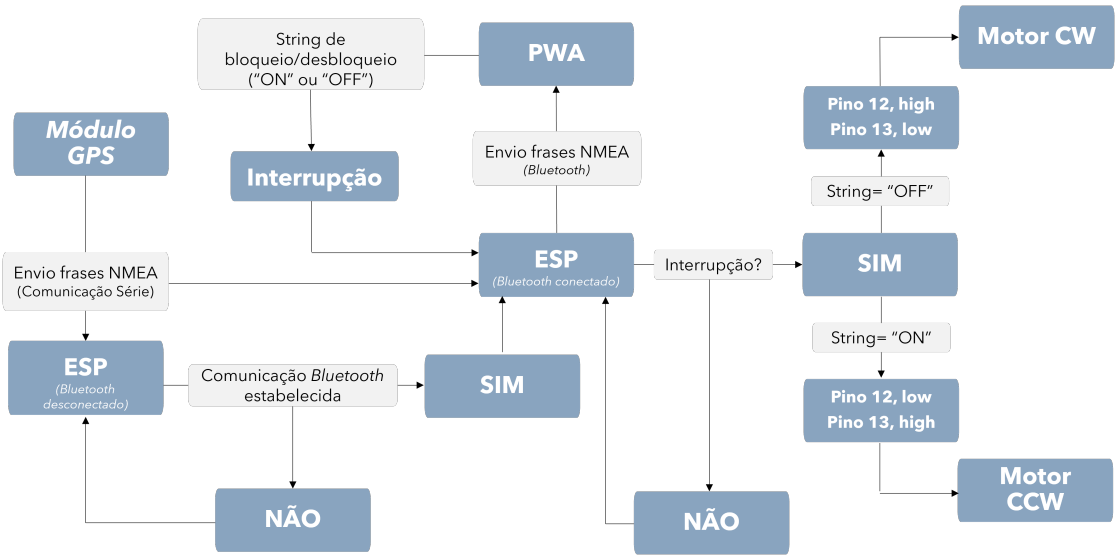


Figura 4.6: Diagrama ilustrativo da rotina principal do Esp.

As mensagens NMEA chegam ao microcontrolador como se observa na Figura 4.7. No entanto, apenas duas destas frases contêm informação relevante a este trabalho, $GPRMC (Figura 4.8) e $GPGGA (Figura 4.9).

```
$GPRMC,205128.00,A,4045.32642,N,00834.07808,W,0.565,,221021,,,A*66
$GPVTG,,T,,M,0.565,N,1.046,K,A*26
$GPGGA,205128.00,4045.32642,N,00834.07808,W,1,06,1.88,10.1,M,49.8,M,,*7C
$GPGSA,A,3,12,25,10,32,29,31,,,,,,,2.80,1.88,2.08*0E
$GPGSV,3,1,12,02,13,094,,$GPRMC,205130.00,A,4045.32550,N,00834.07935,W,0.222,,221021,,,A*64
$GPVTG,,T,,M,0.222,N,0.411,K,A*25
$GPGGA,205130.00,4045.32550,N,00834.07935,W,1,06,1.88,12.8,M,49.8,M,,*71
$GPGSA,A,3,12,25,10,32,29,31,,,,,,,2.80,1.88,2.08*0E
$GPGSV,3,1,12,02,13,094,,06,08,056,,10,08,242,16,11,06,090,*7D
$GPGSV,3,2,12,12,66,035,17,15,00,159,,19,03,031,12,24,42,106,*7C
$GPGSV,3,3,12,25,71,282,20,29,30,178,26,31,08,296,13,32,48,294,20*76
$GPGLL,4045.32550,N,00834.07935,W,205130.00,A,A*7D
```

Figura 4.7: Exemplos de frases NMEA vindas do módulo GPS

Como podemos observar pelas imagens anteriores, com as frases $GPRMC e $GPGGA é possível obter os seguintes dados: hora, data, latitude, longitude, altitude, velocidade instantânea e o número de satélites utilizados para obter esta informação. A hora e a data são importantes para ordenar a informação numa base de dados e para calcular o tempo de viagem. Através da latitude e da longitude é possível mostrar a posição da bicicleta no mapa. A altitude, a velocidade instantânea e o tempo de viagem podem ser utilizados para calcular as calorias queimadas pelo utilizador. O número de satélites pode ser utilizado para saber a qualidade da informação fornecida pelo módulo e decidir

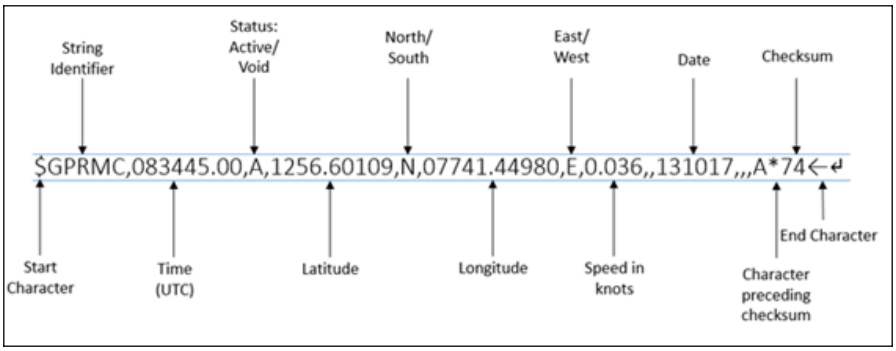


Figura 4.8: Descrição de uma frase $GPRMC exemplo

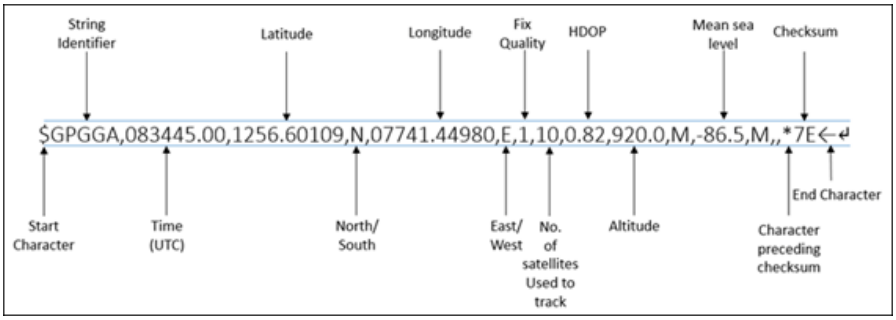


Figura 4.9: Descrição de uma frase $GPGGA exemplo

se a informação é válida ou não.

Para selecionar estas duas frases das restantes foi gerada uma rotina que deteta as posições das *strings* "$GPRMC" e "$GPGGA"na mensagem original do módulo GPS ("Message") e cria duas *strings* a partir do índice dessas posições até ao índice do "\n". O resultado são as duas *strings* desejadas como se observa na Figura 4.10. Esta rotina é ativada todas as vezes que é recebida uma mensagem do módulo GPS (Figura 4.11).

```
$GPRMC,200726.00,A,4045.32634,N,00834.08511,W,0.802,,221021,,,A*6C$GPGGA,200729.00,4045.33021,N,00834.08657,W,1,05,1.25,27.4,M,49.8,M,,*72
$GPRMC,200729.00,A,4045.33021,N,00834.08657,W,1.489,,221021,,,A*6F$GPGGA,200732.00,4045.33118,N,00834.08718,W,1,05,1.25,27.6,M,49.8,M,,*7B
$GPRMC,200732.00,A,4045.33118,N,00834.08718,W,0.525,,221021,,,A*62$GPGGA,200735.00,4045.33207,N,00834.08696,W,1,05,1.25,27.9,M,49.8,M,,*79
$GPRMC,200735.00,A,4045.33207,N,00834.08696,W,0.976,,221021,,,A*65$GPGGA,200738.00,4045.33179,N,00834.08673,W,1,05,1.25,28.1,M,49.8,M,,*72
$GPRMC,200738.00,A,4045.33179,N,00834.08673,W,0.969,,221021,,,A*67$GPGGA,200741.00,4045.33013,N,00834.08493,W,1,05,1.25,28.4,M,49.8,M,,*78
```

Figura 4.10: Exemplo de mensagem enviada o outro dispositivo via *Bluetooth*

Sempre que algum dispositivo se liga via *Bluetooth* ao Esp-32 através da *app* esta mensagem é enviada de modo a ser possível visualizar a posição da bicicleta no mapa, Figura 4.12.

Quando uma mensagem é recebida por *Bluetooth* é ativada uma interrupção que verifica se esta tem um tamanho maior que 0 e, se isto for verdade, posteriormente verifica se o seu conteúdo é igual à string "ON" ou "OFF", que são as palavras-chave para o bloqueio e desbloqueio do cadeado, respetivamente. Se a mensagem for igual a uma destas palavras-chave inicia-se um ciclo que só será desativado quando o sensor de fecho e de abertura passar a apresentar um valor inferior a 5, desativando as saídas digitais responsáveis por ativar o motor Figura 4.13.

```
if (SerialGPS.available()) {
   SerialGPS.setTimeout(450);
   String Message= SerialGPS.readString();

   String GPRMC= Message.substring(Message.indexOf("$GPRMC"),Message.length());
   GPRMC= GPRMC.substring(0,GPRMC.indexOf('\n'));
   String GPGGA= Message.substring(Message.indexOf("$GPGGA"),Message.length());
   GPGGA= GPGGA.substring(0,GPGGA.indexOf('\n'));

   FinalMessage = GPGGA + "\r\n" + GPRMC;
   //Serial.print(FinalMessage.c_str());
   //Serial.print(Message.c_str());
}
```

Figura 4.11: Rotina de seleção de frases NMEA

```
if (deviceConnected) {

  delay(50);
  pCharacteristic->setValue(FinalMessage.c_str());
  pCharacteristic->notify();
 }
```

Figura 4.12: Rotina de envio de mensagens via *Bluetooth*

4.6 Programação da Aplicação

Como já foi referido anteriormente, optou-se por criar uma *Progressive Web App* ou PWA que por meio do *Visual Studio Code* em linguagem *JavaScript*, *HTML* e *CSS*. Esta aplicação é o meio por onde o utilizador comunica com o cadeado e convém que esta seja intuitiva e apelativa. Para isso, utilizou-se uma biblioteca vocacionada para a interface com o utilizador, desenvolvida pelo *Facebook*, "*React*". Desta forma, foi possível criar animações, botões e ambientes visualmente apelativos e de fácil interpretação, Anexo A.2. Infelizmente a leitura do `RSSI` por parte do *web browser* ainda não está implementada [23] e não foi possível continuar com o conceito inicial de desbloquear e bloquear o cadeado através de sinal *Bluetooth* .

É também função desta aplicação permitir a ligação, via *Bluetooth*, do dispositivo onde está a ser executada ao Esp-32; enviar para o microcontrolador as mensagens "*ON*" e "*OFF*" aquando da manifestação do utilizador de bloquear ou desbloquear o cadeado; e receber as mensagens NMEA enviadas pelo Esp-32, tratá-las e mostrá-las num mapa.

A aplicação tem um menu inicial (Figura 4.14) com 4 botões: "*Bluetooth*", que inicia a rotina de emparelhamento; "*Lock*" e "*Unlock*", que só podem ser utilizados quando já se estiver conectado a um dispositivo; e "*Location*", onde é possível visualizar a posição do cadeado num mapa, e que, neste caso também só funciona caso se esteja conectado ao cadeado, uma vez que a comunicação GSM não foi implementada.

Ao ser selecionado, o botão Bluetooth desencadeia a rotina de emparelhamento e, caso esta seja bem-sucedida, começa a receber as mensagens vindas do Esp-32 e está pronto a enviar os comados "*ON*", "*OFF*" para bloqueio e desbloqueio do cadeado como se observa nas Figuras 4.15 e 4.16.

As funções "*Lock*" e "*Unlock*" estão associadas aos botões, isto é, são executadas sem-

```
class MyCallbacks: public BLECharacteristicCallbacks {
    void onWrite(BLECharacteristic *pCharacteristic) {
      std::string value = pCharacteristic->getValue();

      if (value.length() > 0) {
        Serial.println("*********");
        Serial.print("New value: ");
        for (int i = 0; i < value.length(); i++)
          Serial.print(value[i]);

        Serial.println();
        Serial.println("*********");
        if (value== "ON"){
          while (touchRead(ClosedSensor) > 5) {
          digitalWrite (LED_PIN, HIGH);
          digitalWrite (Forward, HIGH);
          digitalWrite (Reverse, LOW);
          }

          digitalWrite (LED_PIN, LOW);
          digitalWrite (Forward, LOW);
          digitalWrite (Reverse, LOW);

        }
        if (value== "OFF"){
          while (touchRead(OpenedSensor) > 5) {
          digitalWrite (LED_PIN, LOW);
          digitalWrite (Forward, LOW);
          digitalWrite (Reverse, HIGH);
          }

          digitalWrite (LED_PIN, LOW);
          digitalWrite (Forward, LOW);
          digitalWrite (Reverse, LOW);

        }
      }
    }
};
```

Figura 4.13: Interrupção responsável por bloquear ou desbloquear o cadeado

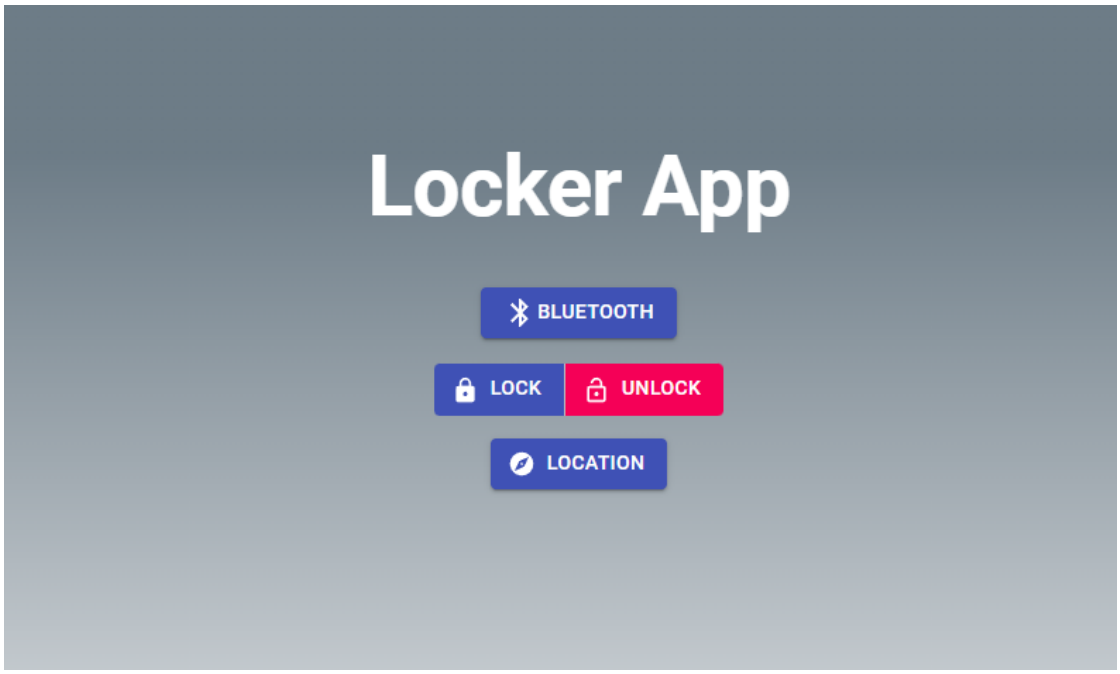


Figura 4.14: Menu principal da aplicação

pre que estes botões forem selecionados. Já a função "handleCharacteristicNotifications" é executada sempre que é recebida uma mensagem proveniente do microcontrolador e descodifica-a, uma vez que esta é enviada em ASCII. Como já foi mencionado acima, essas mensagens correspondem as frases NMEA que contêm informação de localização entre outras.

Na Figura 4.17, a mensagem recebida é separada em duas pelo carater "\n" sendo guardada na variável "NMEA", na posição 1, a frase $GPGGA e, na posição 2, a frase $GPRMC. Posteriormente, se a posição onde são escritas as coordenadas e as suas direções não estiver vazia, é chamada uma função responsável pela conversão das coordenadas de graus minutos e segundos para graus decimais de modo que estas sejam mostradas no mapa. Os valores das coordenadas, hora, data, velocidade, número de satélites e altitude são guardados para futuras possíveis melhorias na aplicação.

4.7 Funcionamento da Aplicação

Em primeiro, é necessário emparelhar o dispositivo móvel ou computador com *Bluetooth* ao cadeado. Para isso basta um toque sobre o botão "*Bluetooth*" que, pedirá ao utilizador que ative o *Bluetooth* se este não estiver ativado e posteriormente apresentará uma lista de dispositivos prontos para emparelhar (Figura 4.18). Neste momento o estado do botão *Bluetooth* passa a uma espera, enquanto o emparelhamento não é realizado. Quando o emparelhamento é realizado com sucesso o botão que antes dizia "*Bluetooth*" agora diz "*Connected*" e possui a cor verde (Figura 4.19), demonstrando que a ligação foi bem-sucedida.

A partir desse momento é possível bloquear e desbloquear o cadeado selecionando os botões "*Lock*" e "*Unlock*" respetivamente, e consultar a localização do cadeado com o

Bluetooth-Service.js > Bluetooth

```
import { DMStoOD } from "./DMStoOD-Service"
import Store from './store';

export default class Bluetooth {
    static device = null;
    static server = null;
    static service = null;
    static bleCharacteristic = null;
    static notifications = null;


    static async connect() {
        const _ = Bluetooth;

        try {
            _.device = await navigator.bluetooth.requestDevice({
                //acceptAllDevices:true,
                filters: [{ services: ["4fafc201-1fb5-459e-8fcc-c5c9c331914b"] }]
            });
            //print("Step 2: Connect to it")
            _.server = await _.device.gatt.connect();
            //device.watchAdvertisements();
            //device.addEventListener('advertisementreceived', GetRSSI);
            //print("Step 3: Get the Service")
            _.service = await _.server.getPrimaryService("4fafc201-1fb5-459e-8fcc-c5c9c331914b");
            //print("Step 4: get the Characteristic")
            _.bleCharacteristic = await _.service.getCharacteristic("beb5483e-36e1-4688-b7f5-ea07361b26a8");

            //print(!!bleCharacteristic.startNotifications);
            _.notifications = await _.bleCharacteristic.startNotifications();
            //print("Step 5: startNotifications")
            _.bleCharacteristic.addEventListener('characteristicvaluechanged', Bluetooth.handleCharacteristicNotifications);

            return Promise.resolve();

        } catch (error) {
            //print( "error: " + error);
            console.error('Connection failed!', error);
        }

        return Promise.reject();
    }
```

Figura 4.15: Rotina de emparelhamento *Bluetooth*

```
43
44      static Lock() {
45          Bluetooth.sendMessage("ON");
46      }
47
48      static Unlock() {
49          Bluetooth.sendMessage("OFF");
50      }
51
52      static sendMessage(message) {
53          let encoder = new TextEncoder('utf-8');
54          Bluetooth.bleCharacteristic.writeValue(encoder.encode(message));
55      }
56
57      static handleCharacteristicNotifications(event) {
58          let value = event.target.value;
59          let decoder = new TextDecoder('utf-8');
60          const message = decoder.decode(value)
61          console.log(message);
```

Figura 4.16: Rotina de receção e conversão das mensagens ASCII e rotinas de envio das mensagens de bloqueio e desbloqueio

```
62
63      if (message[0] == "$") {
64          const NMEA = message.split("\n");
65          const [, , , , , , , Satellite, ,Altitude] = NMEA[0].split(",");
66          const [,Time, , coordinates1, dir_lat, coordinates2, dir_lon, Velocity, , Date] = NMEA[1].split(",");
67
68          if (coordinates1!=""||dir_lat!=""||coordinates2!=""||dir_lon!=""){
69          const coordinate = DMStoDD(coordinates1, coordinates2, dir_lat, dir_lon);
70          Store.addCoordinate(coordinate);
71          Store.addTime(Time);
72          Store.addDate(Date);
73          Store.addVelocity(Velocity);
74          Store.addSatellite(Satellite);
75          Store.addAltitude(Altitude);
76          console.log(Time, Date, Velocity, Satellite, Altitude);
77          }
78      } else {
79          //print(Message);
80      }
```

Figura 4.17: Código de seleção e armazenamento das variáveis presentes nas mensagens NMEA

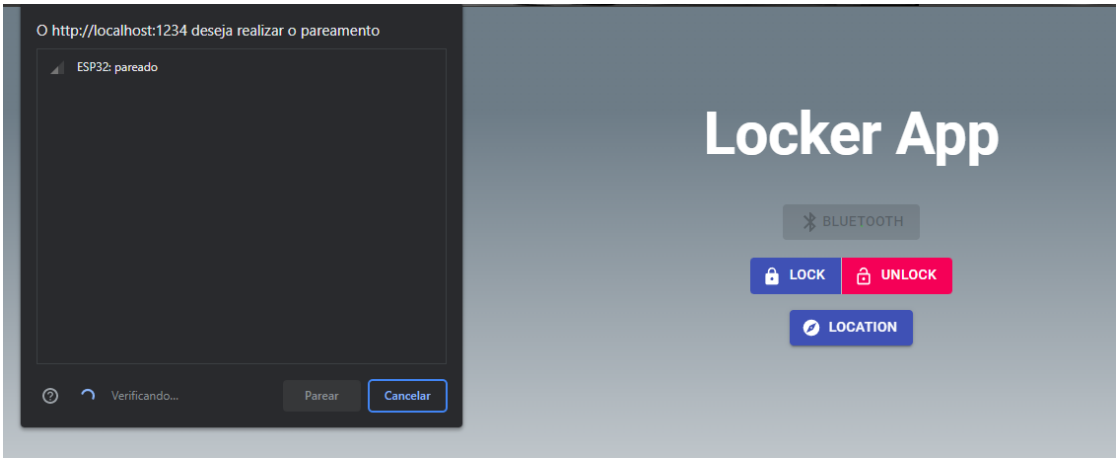


Figura 4.18: Lista de dispositivos disponíveis para emparelhamento

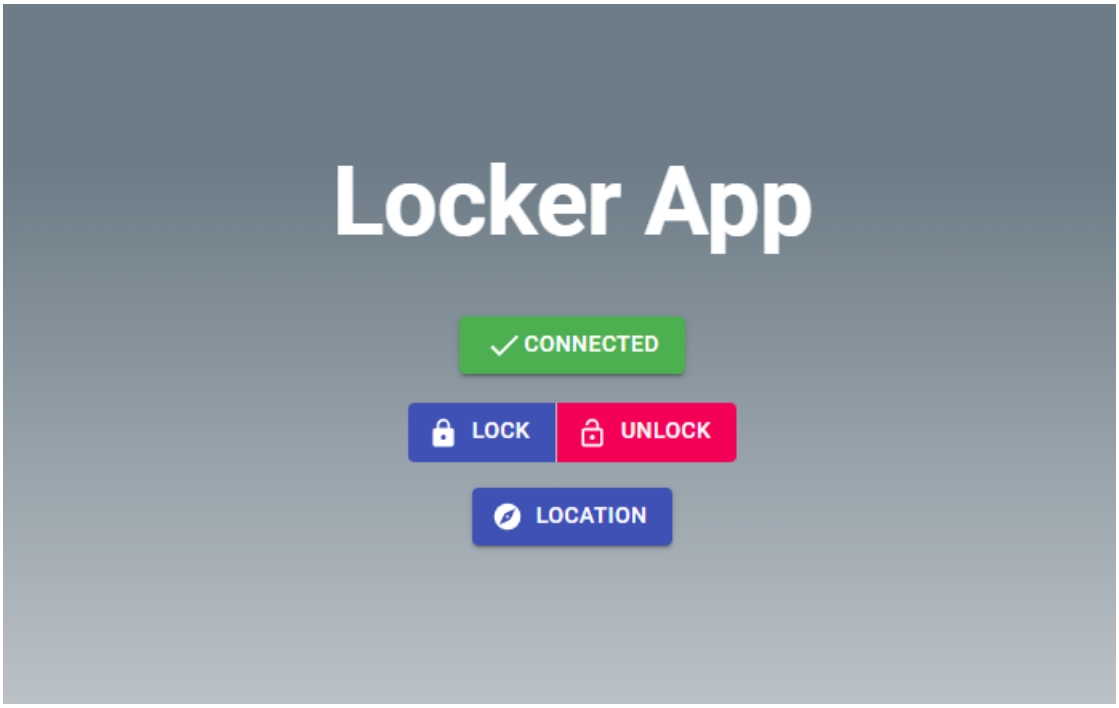


Figura 4.19: Dispositivo conectado com sucesso

botão "*Location*", que ao ser selecionado levará o utilizador a uma nova interface, Figura 4.20. Este novo ambiente possui um mapa, onde são possíveis ações como arrastar e ampliar ou reduzir, e um botão de retorno no canto superior esquerdo. No caso da localização não ser válida (devido a problemas de cobertura) surge no ecrã uma caixa de diálogo que informa o utilizador dessa mesma situação Figura 4.21, e, por defeito, a posição mostrada no mapa é Londres.

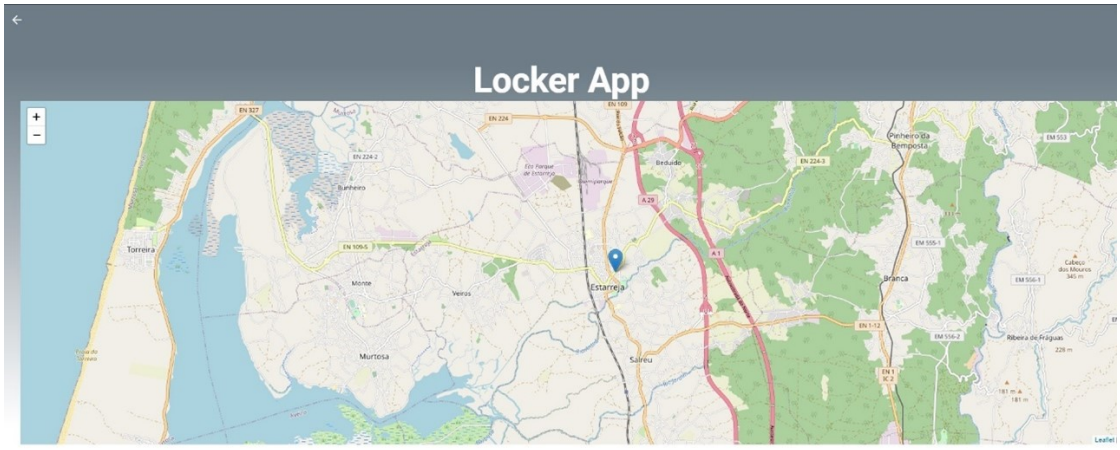


Figura 4.20: Interface de localização

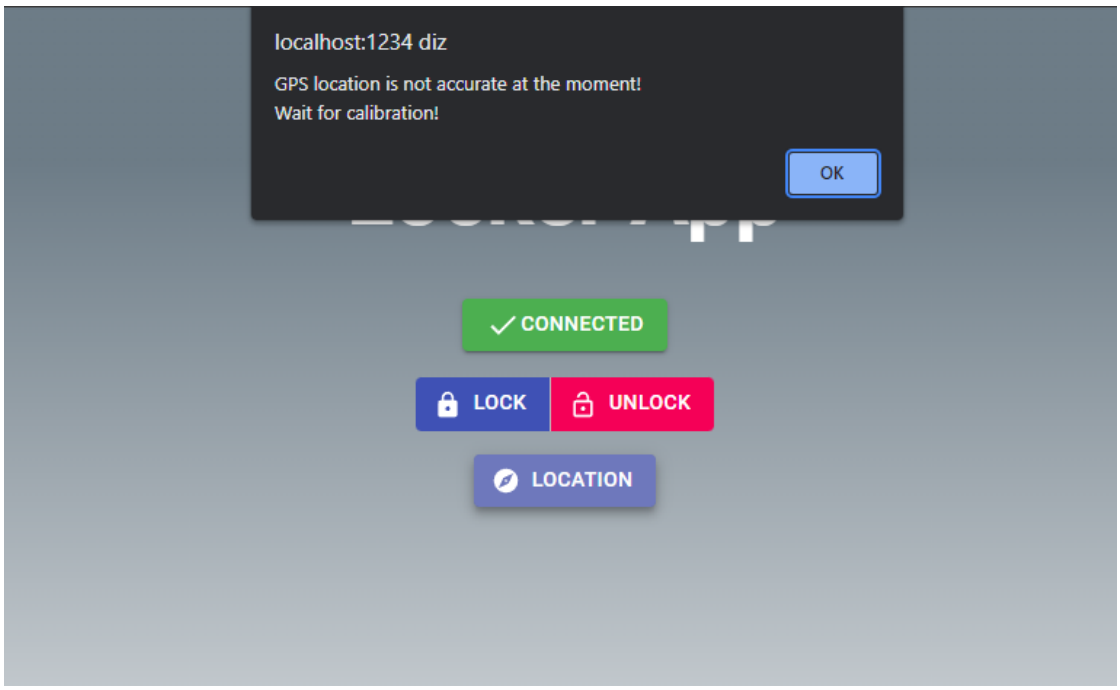


Figura 4.21: Mensagem de aviso de imprecisão das coordenadas recebidas

Capítulo 5

Projeto Mecânico

Neste capítulo apresenta-se o projeto mecânico do fecho automático do cadeado. Dimensionou-se um sistema com parafuso sem fim determinando-se os parâmetros característicos deste e da respetiva coroa 5.1.

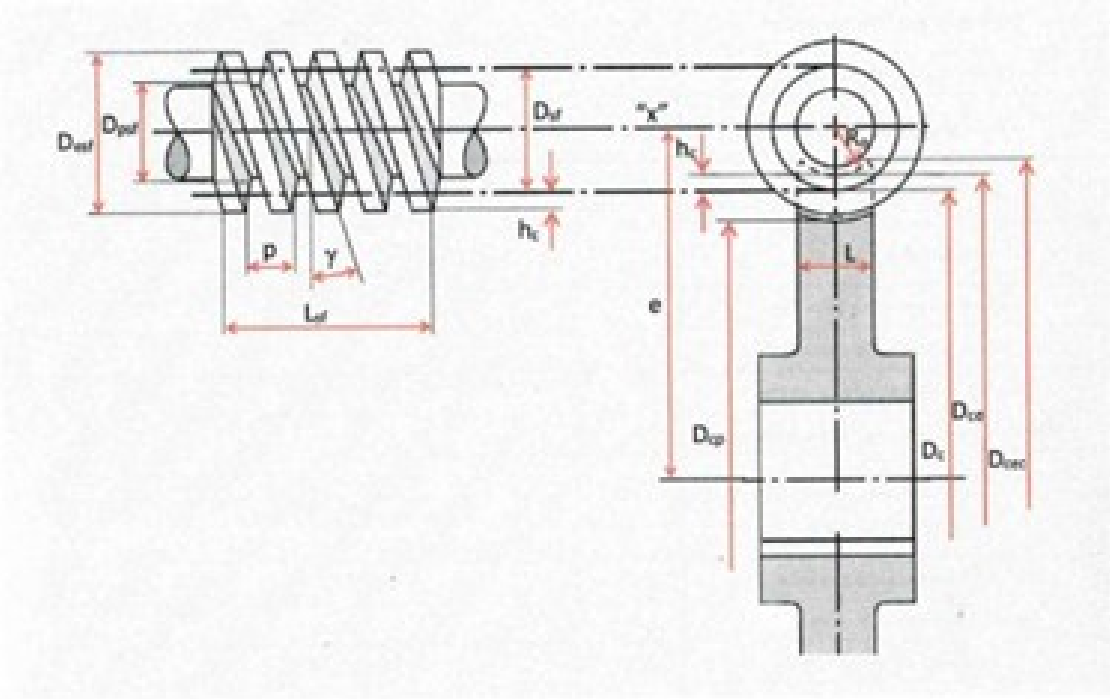


Figura 5.1: Engrenagem roda de coroa parafuso sem-fim [25].

5.1 Dimensionamento do sistema

O dimensionamento do sistema com parafuso sem fim passou por um cálculo iterativo com a necessidade de se arbitrar o diâmetro exterior da coroa,D_{ce}, o número de entradas do sem-fim, i, e o módulo real, m, como explicado no esquema da Figura 5.2.

Apenas se apresenta detalhadamente o dimensionamento para a iteração considerada

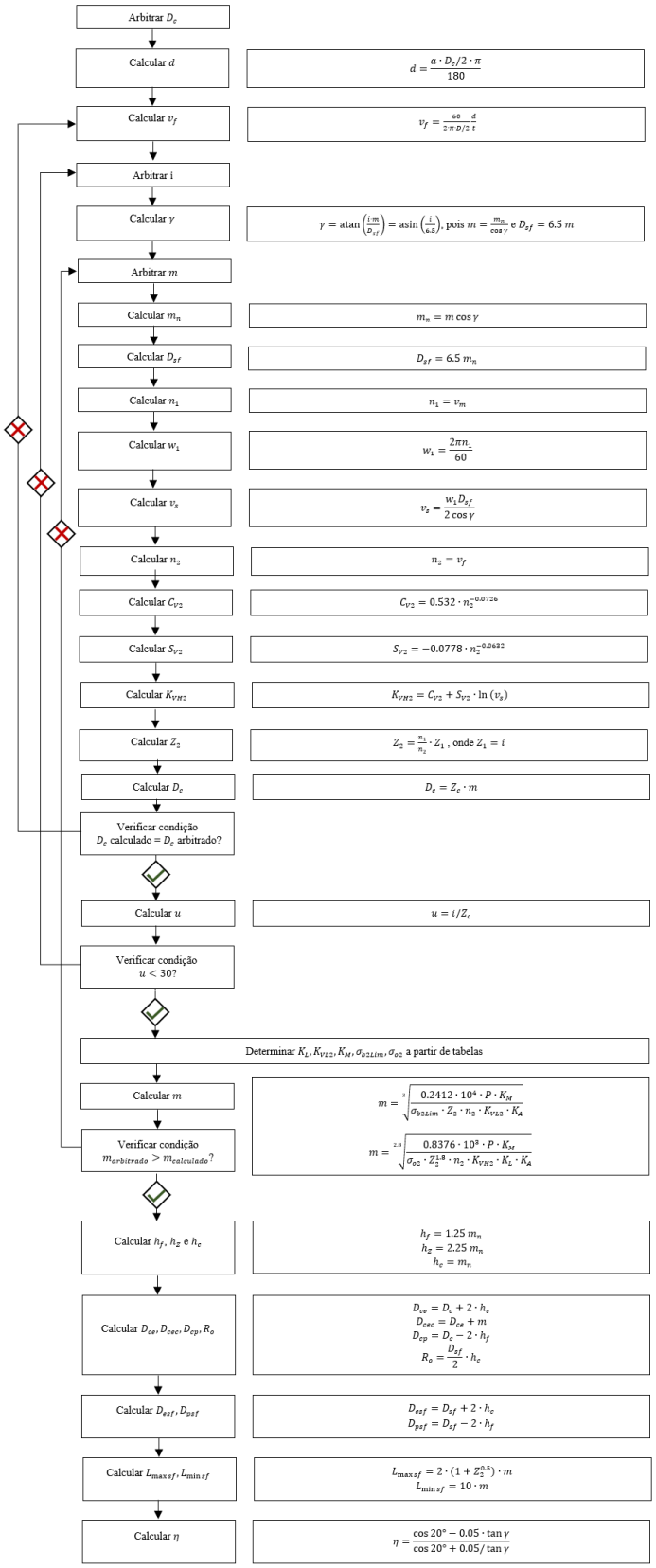


Figura 5.2: Estratégia de dimensionamento

para o projeto, onde o valor arbitrado para o número de entradas do sem-fim foi 3, o diâmetro da coroa foi 132.8 mm e módulo real foi 1.75

Tendo em consideração a potência e rotações por minuto do mini motor utilizado e um arco de 120°com um comprimento de 132.8 mm, calculou-se o comprimento do arco correspondente obtendo-se um valor de 118.21 mm. Este arco corresponde à distância que a parte funcional do cadeado terá que percorrer no seu fecho e abertura. Assumindo um tempo de fecho de 2.2 s calculou-se a velocidade da coroa através da equação 5.1.

$$v_f = \frac{d}{t} \tag{5.1}$$

$$v_f = \frac{118.21 \cdot 10^{-3}}{2.2} = 0.054 \quad m/s$$

Tendo em conta que o raio exterior da roda é 69.8 converteu-se a velocidade para rotações por minuto pela equação 5.2.

$$rpm = \frac{v \cdot 60}{2 \cdot \pi \cdot r} \tag{5.2}$$

$$rpm = \frac{0.054 \cdot 60}{\pi \cdot 132.8} = 7.89 \quad rpm$$

O ângulo de hélice, γ, pode ser calculado a partir da equação 5.3, onde D_{sf} é o diâmetro primitivo do sem-fim.

$$\gamma = \arctan\left(\frac{i \cdot m}{D_{sf}}\right) \tag{5.3}$$

O módulo real pode ser descrito pela equação 5.4, onde m_n representa o módulo nominal e pode-se aproximar o diâmetro primitivo do sem-fim pela equação 5.5.

$$m = \frac{m_n}{\cos\gamma} \tag{5.4}$$

$$D_{sf} = 6.5 \cdot m_n \tag{5.5}$$

Rescrevendo a equação 5.3 considerando as equações anteriores para o módulo real e para o diâmetro do sem-fim, obtém-se a equação 5.6 para o ângulo de hélice.

$$\gamma = \arcsin\left(\frac{i}{6.5}\right) \tag{5.6}$$

Substituindo-se na equação anterior, o valor arbitrado de $i = 3$, obteve-se um ângulo de hélice de 0.48 rad.

$$\gamma = \arcsin\left(\frac{i \cdot m_n}{6.5 \cdot m_n}\right) = \arcsin\left(\frac{3}{6.5}\right) = 0.48 \quad rad$$

Arbitrou-se um valor de 1.75 para o módulo real e calculou-se o módulo nominal e o diâmetro do sem-fim pelas equações 5.4 e 5.5, respetivamente. Obtendo-se um m_n de 1.55 mm e um D_{sf} de 10.09 mm.

$$m_n = m \cdot \cos\gamma = 1.75 \cdot \cos 0.48 = 1.55 \quad mm$$

$$D_{sf} = 6.5 \cdot m_n = 6.5 \cdot 1.55 = 10.09 \quad mm$$

A velocidade de rotação do fuso, n_1 é igual à velocidade de rotação do motor,v_m, logo aplicando a equação 5.7, tem-se que a velocidade de rotação do fuso é igual a 200 rpm e aplicando a equação 5.8 que converte a velocidade de rotação do fuso de rotações por minuto para radianos por segundo, obteve-se um valor de 20.94 rad/s para w_1.

$$n_1 = v_m \tag{5.7}$$

$$n_1 = 200 \quad rpm$$

$$w_1 = \frac{2 \cdot \pi \cdot n_1}{60} \tag{5.8}$$

$$w_1 = \frac{2 \cdot \pi \cdot 200}{60} = 20.94 \quad rad/s$$

A velocidade de escorregamento, V_s, é descrita pela equação 5.9. Substituindo os parâmetros da equação pelos valores determinados anteriormente, tem-se $V_s = 0.12\ m/s$

$$V_s = \frac{\omega_1 \cdot D_{sf}}{2 \cdot \cos\gamma} \tag{5.9}$$

$$V_s = \frac{20.94 \cdot 10.09 \cdot 10^{-3}}{2 \cdot \cos 0.48} = 0.12 \quad m/s$$

A velocidade de rotação da coroa, n_2, é igual à velocidade de fecho, v_f, logo aplicando a equação 5.10, tem-se que a velocidade de rotação da coroa é igual a 0.054 m/s, ou seja, 7.89 rpm.

$$n_2 = v_f \tag{5.10}$$

$$n_2 = 0.054 \ \ m/s = 7.89 \ \ rpm$$

Para $n_2 < 10\ rpm$, os parâmetros C_{V2} e S_{V2} são descritos pelas equações 5.11 e 5.12, respetivamente.

$$C_{V2} = 0.532 \cdot n_2^{-0.0726} \tag{5.11}$$

$$S_{V2} = -0.0778 \cdot n_2^{-0.0632} \tag{5.12}$$

Substituindo nas equações anteriores a velocidade de rotação da coroa pelo valor calculado anteriormente, obtém-se um C_{V2} de 0.46 e um S_{V2} de 0.04.

$$C_{V2} = 0.532 \cdot 7.89^{-0.0726} = 0.46 \quad rpm$$

$$S_{V2} = -0.0778 \cdot 7.89^{-0.0632} = 0.04 \quad rpm$$

O cálculo do fator de velocidade de escorregamento, K_{VH2}, foi efetuado recorrendo à equação 5.13. O valor obtido foi de 0.38.

$$K_{VH2} = C_{V2} + S_{V2} \cdot \ln V_s \tag{5.13}$$

$$K_{VH2} = 0.46 + 0.04 \cdot \ln 0.12 = 0.38$$

O número de dentes da coroa Z_2 pode ser calculado pela equação 5.14, onde Z_1 representa o número de entradas do fuso, ou seja, o valor de i=3, tendo-se determinado que a coroa possui 76 dentes.

$$Z_2 = \frac{n_1}{n_2} \cdot Z_1 \tag{5.14}$$

$$Z_2 = \frac{200}{7.89} \cdot 3 \approx 76$$

Aplicando a equação 5.15, calculou-se o diâmetro da coroa, D_c, onde Z_c, que é igual a Z_2, corresponde ao número de dentes da mesma, concluindo-se que a coroa possui 133 mm de diâmetro. Como o diâmetro calculado é aproximadamente igual ao diâmetro arbitrado inicialmente, a condição é satisfeita e pode continuar-se o procedimento de cálculo.

$$D_c = Z_c \cdot m \tag{5.15}$$

$$D_c = Z_c \cdot m = 77 \cdot 1.75 = 133 \quad mm$$

A razão de transmissão u é um quociente entre o número de entradas do fuso e o número de dentes da coroa, como se encontra representado na equação 5.16.

$$u = \frac{i}{Z_c} \tag{5.16}$$

$$u = \frac{i}{Z_c} = \frac{3}{76} = 25.3$$

Comparando o valor obtido para a razão de transmissão com o valor máximo permitido ($u < 30$), para um i=3 ou 4, verifica-se que a condição é satisfeita e o valor arbitrado para i é aceite.

Para verificar se o valor arbitrado para o módulo real da engrenagem, m é aceite, é necessário verificar o critério de dimensionamento à fadiga, equação 5.17 e o critério de verificação à pressão de Hertz, equação 5.18. P representa a potência a transmitir pela engrenagem, K_M o fator de efeito dinâmico, σ_{b2Lim} tensão limite de fadiga à flexão, Ω_{02} fator de contacto, K_{VL2} fator de velocidade, K_L fator de duração e K_A fator de precisão de montagem.

$$m \geq \sqrt[3]{\frac{0.2412 \cdot 10^4 \cdot P \cdot K_M}{\sigma_{b2lim} \cdot Z_2 \cdot n_2 \cdot K_{VL2} \cdot K_A}} \tag{5.17}$$

$$m \geq \sqrt[2.8]{\frac{0.8376 \cdot 10^3 \cdot P \cdot K_M}{\Omega_{02} \cdot Z_2^{1.8} \cdot n_2 \cdot K_{VH2} \cdot K_L \cdot K_A}} \tag{5.18}$$

K_M	σ_{b2Lim}	Ω_{02}	K_{VL2}	K_L	K_A
1	50	2.85	0.7	2.2	1

Tabela 5.1: Parâmetros utilizados no cálculo do módulo real, m

Os parâmetros K_M, σ_{b2Lim}, Ω_{02}, K_{VL2}, K_L e K_A foram retirados de tabelas do livro [25] que se encontram em no Anexo A.3 e apresentam-se na Tabela 5.1.

Substituindo nas equações 5.17 e 5.18 pelos valores anteriores, verifica-se que o módulo real tem de ser superior a 0.56.

$$m \geq \sqrt[3]{\frac{0.2412 \cdot 10^4 \cdot P \cdot K_M}{\sigma_{b2lim} \cdot Z_2 \cdot n_2 \cdot K_{VL2} \cdot K_A}} <=> m \geq \sqrt[3]{\frac{0.2412 \cdot 10^4 \cdot 1.5 \cdot 1}{50 \cdot 76 \cdot 7.89 \cdot 0.7 \cdot 1}} <=> m \geq 0.56$$

$$m \geq \sqrt[2.8]{\frac{0.8376 \cdot 10^3 \cdot P \cdot K_M}{\Omega_{02} \cdot Z_2^{1.8} \cdot n_2 \cdot K_{VH2} \cdot K_L \cdot K_A}} <=> m \geq \sqrt[2.8]{\frac{0.8376 \cdot 10^3 \cdot 1.5 \cdot 1}{2.85 \cdot 76^{1.8} \cdot 7.89 \cdot 0.38 \cdot 2.2 \cdot 1}}$$

$$<=> m \geq 0.28$$

O módulo real arbitrado foi de 1.75, sendo superior a 0.28 e 0.56, cumprindo, assim, os dois critérios anteriores e não sendo necessário fazer uma nova iteração.

A altura do pé do dente, h_f, a altura total do dente, h_z, e a altura da cabeça do dente, h_c, foram calculadas a partir das equações 5.19, 5.20 e 5.21, respetivamente.

$$h_f = 1.25 \cdot m_n \tag{5.19}$$

$$h_z = 2.25 \cdot m_n \tag{5.20}$$

$$h_c = m_n \tag{5.21}$$

$$h_f = 1.25 \cdot m_n = 1.25 \cdot 1.55 = 1.94 \quad mm$$

$$h_z = 2.25 \cdot m_n = 2.25 \cdot 1.55 = 3.49 \quad mm$$

$$h_c = m_n = 1.55 \quad mm$$

O diâmetro da cabeça da coroa, D_{ce}, o diâmetro externo da coroa, D_{cec} , o diâmetro interno da coroa, D_{cp} e o raio de engrenamento da coroa foram determinados pelas equações 5.22,5.23,5.25 respetivamente.

$$D_{ce} = D_c + 2 \cdot h_c \tag{5.22}$$

$$D_{cec} = D_{ce} + m \tag{5.23}$$

$$D_{cp} = D_c - 2 \cdot h_f \tag{5.24}$$

$$R_o = \frac{D_{sf}}{2} - m_n \tag{5.25}$$

$$D_{ce} = D_c + 2 \cdot h_c = 133 + 2 \cdot 1.55 = 136.10 \quad mm$$

$$D_{cec} = D_{ce} + m = 136.10 + 1.75 = 137.90 \quad mm$$

$$D_{cp} = D_c - 2 \cdot h_f = 133 - 2 \cdot 1.94 = 129.12 \quad mm$$

$$R_o = \frac{D_{sf}}{2} - m_n = \frac{10.09}{2} - 1.55 = 3.49 \quad mm$$

O diâmetro externo do sem-fim, D_{esf}, e o diâmetro interno do sem-fim, D_{psf} são descritos pelas equações 5.26 e 5.27, respetivamente.

$$D_{esf} = D_{sf} + 2 \cdot h_c \tag{5.26}$$

$$D_{psf} = D_{sf} - 2 \cdot h_f \tag{5.27}$$

$$D_{esf} = D_{sf} + 2 \cdot h_c = 10 + 2 \cdot 1.55 = 13.20 \quad mm$$

$$D_{psf} = D_{sf} - 2 \cdot h_f = 10 - 2 \cdot 1.94 = 6.21 \quad mm$$

Determinou-se o comprimento máximo e mínimo do sem-fim,L_{sf} e $L_{sf(min)}$ a partir das equações 5.28 e 5.29, respetivamente.

$$L_{sf} < 2 \cdot (1 + Z_c^{0.5}) \cdot m \tag{5.28}$$

$$L_{sf(min)} > 10 \cdot m \tag{5.29}$$

$$L_{sf} < 2 \cdot (1 + Z_c^{0.5}) \cdot m <=> L_{sf} < 2 \cdot (1 + 76^{0.5}) \cdot 1.75 <=> L_{sf} < 34.01 \quad mm$$

$$L_{sf(min)} > 10 \cdot m <=> L_{sf(min)} > 10 \cdot 1.75 <=> L_{sf(min)} > 17.5 \quad mm$$

Na Tabela 5.2 apresentam-se os principais parâmetros de dimensionamento para a coroa e para o sem-fim e na figura 5.3 um esquema em 3D do sistema do cadeado.

Parâmetros de dimensionamento da coroa			
D_c	D_{ce}	D_{cp}	Z_c
133 mm	136 mm	129 mm	76
Parâmetros de dimensionamento do sem-fim			
D_{sf}	D_{esf}	D_{psf}	Z_1
10 mm	13 mm	6 mm	3

Tabela 5.2: Parâmetros de dimensionamento para a coroa e para o sem-fim

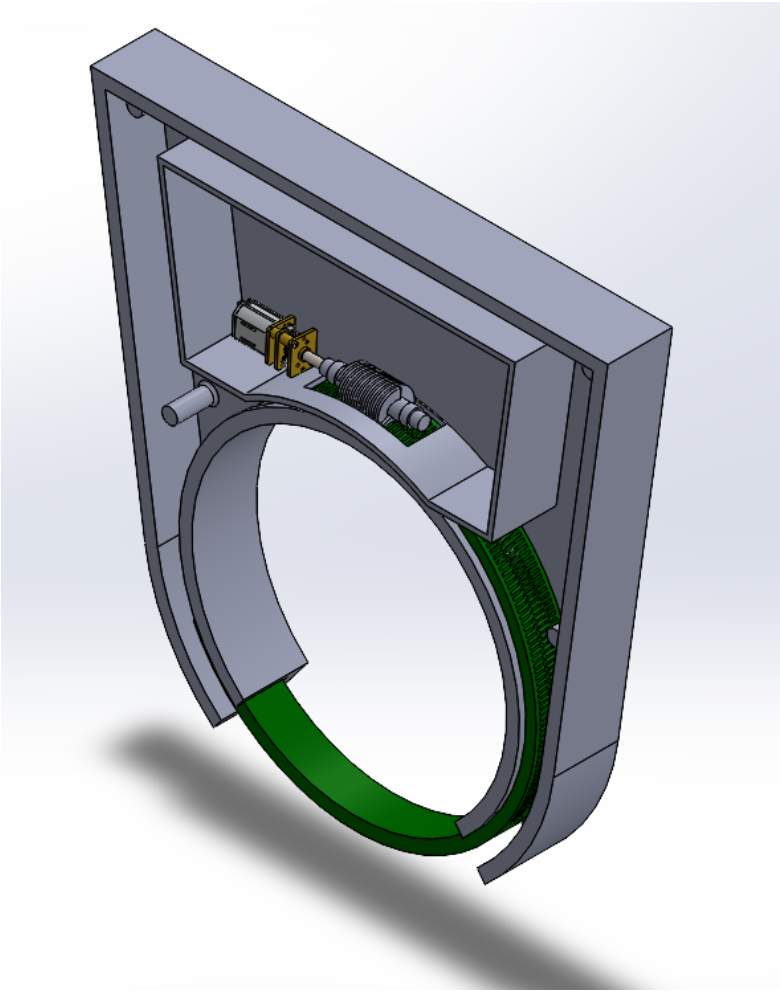


Figura 5.3: Esquema 3D do cadeado.

Capítulo 6

Conclusões e Trabalhos Futuros

6.1 Conclusões

O mercado das bicicletas tradicionais e elétricas tem vindo a aumentar nos últimos anos e, por isso, existe uma motivação para o desenvolvimento de novas formas de bloqueio das bicicletas, como os cadeados automáticos, que além de serem mecanismos eficazes antirroubo, possuem outras características atrativas para os utilizadores. Os principias cadeados automáticos disponíveis no mercado, possuem sistemas de bloqueio e desbloqueio do cadeado remotamente e no próprio cadeado. Existem sistemas que necessitam de um apoio externo para bloquearem a bicicleta e outros são montados no quadro bloqueando uma das rodas. A maioria dos cadeados apresentam um sistema antifurto com alarme sonoro e alguns ainda notificam o utilizador na tentativa de roubo e da localização da bicicleta. A comunicação entre o cadeado e o utilizador é efetuada principalmente por Bluetooth limitando o alcance da comunicação. Os principais extras presentes nos cadeados são a resistência à água, permitir a localização da bicicleta, notificações em caso de tentativa de furto ou acidente e permitem a partilha do veículo com terceiros.

Identificaram-se as necessidades do utilizador e recorrendo ao diagrama de Mudge ordenou-se estas necessidades por ordem decrescente de importância para o cliente. As três principais necessidades do utilizador são a eficiência de bloqueio da bicicleta, possuir GPS e possuir alarme. O diagrama QFD permitiu identificar as características críticas para o cliente e especificar a correlação entre as necessidades do cliente e os requisitos de projeto, tendo-se concluído que a ordem decrescente de atuação é possuir aplicação móvel, Bluetooth, mecanismo de bloqueio, materiais, GPS, alarme, sistema de bloqueio e desbloqueio com código, arquitetura funcional e moderna e tempo de fecho/abertura.

O conceito do cadeado desenvolvido é de fácil montagem, com elevada compatibilidade entre bicicletas e sem necessidade de um objeto externo para a bloquear, com um mecanismo de bloqueio do tipo coroa-parafuso sem-fim. A comunicação entre o cadeado e o utilizador é feita por Bluetooth e GSM/GPRS para permitir um alcance global. O desbloqueio da bicicleta pode ser remoto, desbloqueando o cadeado com a aproximação do utilizador ou manual, no caso de falta de bateria. A aplicação que se pretendia desenvolver permite o bloqueio e desbloqueio da bicicleta possibilitando a partilha da mesma com terceiros e efetua a monitorização de estatísticas como a distância percorrida, velocidade média e calorias queimadas e consegue localizar a bicicleta por GPS.

Infelizmente não foi possível programar o Esp-32 e a aplicação de modo a bloquear e

desbloquear o cadeado à medida que nos afastamos ou aproximamos dele, mas em geral acredita-se que houve uma boa resposta, em termos de desenvolvimento e melhoria, nos requisitos do sistema em que eram mais importantes tendo em conta as necessidades do cliente.

O dimensionamento mecânico do sistema com parafuso sem-fim permitiu a determinação dos parâmetros característicos da coroa e do parafuso. A coroa possui um diâmetro primitivo de 133 mm, com um diâmetro externo de 136 mm e um diâmetro interno de 129 mm, com 76 dentes. O parafuso sem-fim tem um diâmetro primitivo de 10 mm, diâmetro externo de 13 mm e diâmetro interno de 6 mm. O comprimento máximo do sem-fim é de 34 mm e o comprimento mínimo é de 18 mm.

6.2 Trabalhos futuros

Futuramente, seria importante implementar a comunicação `GSM` de forma a conseguir desbloquear o cadeado sem limitações de alcance, permitindo a partilha do veículo, e para notificar o utilizador em caso de furto que poderia ser detetado com um acelerómetro ou através de uma grande variação das coordenadas `GPS`. De modo a aumentar a segurança do cadeado deveria ser acrescentado um alarme sonoro e é indispensável a criação de um desbloqueio manual, através de `NFC` ou de um código pessoal.

Ao nível da aplicação, uma adição interessante seria a criação de um menu de estatísticas do utilizador, que poderia ser conseguido através das mensagens `NMEA` do módulo `GPS`. Acredito também que em breve será possível obter os valores da intensidade do sinal *Bluetooth* por meio, por exemplo, do *Google Chrome* e, uma vez implementada essa função o cadeado poderia ser bloqueado e desbloqueado conforme o afastamento ou aproximação do utilizador, respetivamente.

Por fim, quando todos os componentes eletrónicos estivessem definidos, seria fundamental a criação de um cadeado com as menores dimensões possíveis e com materiais que garantissem a segurança do veículo a que este está acoplado, mas que não comprometessem o seu preço, uma vez que, esse é um dos grandes trunfos contra a concorrência na medida em que os cadeados existentes no mercado possuem um elevado valor de preço de venda ao público.

Numa última fase seria importante investir tempo numa melhoria de design tendo em vista um aumento da segurança e uma diminuição do preço e também numa revisão do projeto mecânico de modo a diminuir os tempos de fecho e de abertura.

Bibliografia

[1] Inventory of U.S. Greenhouse Gas Emissions and Sinks | US EPA [Internet]. [cited 2021 Oct 27]. Available from: https://www.epa.gov/ghgemissions/inventory-us-greenhouse-gas-emissions-and-sinks

[2] CO2 emissions – Global Energy Review 2021 – Analysis - IEA [Internet]. [cited 2021 Oct 27]. Available from: https://www.iea.org/reports/global-energy-review-2021/co2-emissions

[3] Cycling, the better mode of transport [Internet]. [cited 2021 Oct 27]. Available from: https://www.unep.org/news-and-stories/story/cycling-better-mode-transport

[4] Manufacturers can't keep up with the exploding bicycle demand - CBS News [Internet]. [cited 2021 Oct 27]. Available from: https://www.cbsnews.com/news/bike-shortage-nationwide-coronavirus-pandemic/

[5] Oke O, Bhalla K, Love DC, Siddiqui S. Tracking global bicycle ownership patterns. Journal of Transport & Health. 2015 Dec 1;2(4):490–501.

[6] Nokē Hardware | U-Lock [Internet]. [cited 2020 Feb 14]. Available from: https://noke.com/u-lock

[7] Bisecu | Bisecu Smart Bike Lock [Internet]. [cited 2021 Oct 27]. Available from: https://pt.bisecu.com/bisecu-3

[8] Sentinel: your bike's digital bodyguard. | Indiegogo [Internet]. [cited 2021 Oct 27]. Available from: https://www.indiegogo.com/projects/sentinel-your-bike-s-digital-bodyguard#/

[9] LINKA | The Original Smart Bike Lock [Internet]. [cited 2021 Oct 27]. Available from: https://www.linkalock.com/

[10] Patrick H. Lattis Ellipse Smart Bike Lock review: Equal parts smart and frustrating - CNET [Internet]. [cited 2021 Oct 27]. Available from: https://www.cnet.com/reviews/lattis-ellipse-smart-bike-lock-review/

[11] LOCK8 - the World's First Smart Bike Lock by LOCK8 — Kickstarter [Internet]. [cited 2021 Oct 27]. Available from: https://www.kickstarter.com/projects/lock8/lock8-the-worlds-first-smart-bike-lock/description

[12] Bitlock: The world's first smart keyless bike lock [Internet]. [cited 2021 Oct 27]. Available from: https://bitlock.co/

[13] I LOCK IT - smart bike lock – I LOCK IT Shop [Internet]. [cited 2021 Oct 27]. Available from: https://ilockit.bike/en/

[14] Kadalock adds a Bluetooth bike lock to your bottle cage [Internet]. [cited 2021 Oct 27]. Available from: https://newatlas.com/kadalock-bluetooth-bike-lock/36768/

[15] KADALOCK Bicycle Bottle Cage Lock | Taiwantrade.com [Internet]. [cited 2021 Oct 27]. Available from: https://www.taiwantrade.com/product/kadalock-bicycle-bottle-cage-lock-753013.html#

[16] Deeper Lock | The Smartest Toughest Bike Security Ever (Canceled) by Arvydas — Kickstarter [Internet]. [cited 2021 Oct 27]. Available from: https://www.kickstarter.com/projects/156959352/deeper-lock-the-smartest-toughest-bike-security-ev/

[17] Henrique Schuster C, Jonathan Schuster J, Silva de Oliveira A. Application of the Mudge diagram and QFD using the hierarchization of the requirements for a flying car as an example. Revista Gestão da Produção, Operações e Sistemas. 2015 Mar 5;10(1):197–214.

[18] Bisdikian C. An overview of the Bluetooth wireless technology. IEEE Communications Magazine. 2001 Dec;39(12):86–94.

[19] Gomez C, Oller J, Paradells J. Overview and Evaluation of Bluetooth Low Energy: An Emerging Low-Power Wireless Technology. Sensors 2012, Vol 12, Pages 11734-11753 [Internet]. 2012 Aug 29 [cited 2021 Oct 27];12(9):11734–53. Available from: https://www.mdpi.com/1424-8220/12/9/11734/html

[20] GSM - Infopédia [Internet]. [cited 2021 Oct 27]. Available from: https://www.infopedia.pt/$gsm?intlink=true

[21] GPRS - Infopédia [Internet]. [cited 2021 Oct 27]. Available from: https://www.infopedia.pt/$gprs

[22] Lutkevich B. What is a Microcontroller and How Does it Work? [Internet]. [cited 2021 Oct 27]. Available from: https://internetofthingsagenda.techtarget.com/definition/microcontroller

[23] Glossary | Web Fundamentals | Google Developers [Internet]. [cited 2021 Oct 27]. Available from: https://developers.google.com/web/fundamentals/glossary?hl=en

[24] Flores, Paulo & Gomes, J. & Dourado, Nuno & Marques, Filipe. (2017). Engrenagens de Parafuso Sem-Fim.

[25] Completo A, Melo F. Introdução Ao Projeto Mecânico. Engebook - Conteudos de Engenharia e Gestão; 2017. 101–112.

[26] Gupta N. Inside Bluetooth Low Energy [Internet]. Second Edition. Artech House; 2016 [cited 2021 Oct 27]. 1–3.

[27] US10549801B2 - Bicycle locking device and locking method - Google Patents [Internet]. [cited 2021 Oct 27]. Available from: https://patents.google.com/patent/US10549801B2

[28] ESP32¬WROOM¬32 Datasheet. 2021 [cited 2021 Oct 27]; Available from: https://www.espressif.com/en/support/download/documents.

[29] NEO6MV2 Datasheet | PDF [Internet]. [cited 2021 Oct 27]. Available from: https://pt.scribd.com/document/378783749/NEO6MV2-Datasheet

[30] L9110 Datasheet | ASIC - Datasheetspdf.com [Internet]. [cited 2021 Oct 27]. Available from: https://datasheetspdf.com/datasheet/L9110.html

[31] What is Arduino? | Arduino [Internet]. [cited 2021 Oct 27]. Available from: https://www.arduino.cc/en/Guide/Introduction

[32] Visual Studio Code Frequently Asked Questions [Internet]. [cited 2021 Oct 27]. Available from: https://code.visualstudio.com/docs/supporting/faq

Capítulo A

Anexos

A.1 Programação ESP-32

```
/* Based on Neil Kolban example for IDF:
   https://github.com/nkolban/esp32-snippets/blob/master/cpp_utils/tests/BLE%20Tests/SampleServer.cpp
   Ported to Arduino ESP32 by Evandro Copercini
   updates by chegewara
*/
#include <BLEDevice.h>
#include <BLEUtils.h>
#include <BLEServer.h>
#include <BLE2902.h>

// GPS ____________________________________________________________________________
#include <TinyGPS++.h> // Library über http://arduiniana.org/libraries/tinygpsplus/ downloaden und installieren
#include <HardwareSerial.h> // sollte bereits mit Arduino IDE installiert sein
#include "EEPROM.h" // sollte bereits mit Arduino IDE installiert sein

#define EEPROM_SIZE 128
TinyGPSPlus gps;
HardwareSerial SerialGPS(1);

struct GpsDataState_t {
  double originLat = 0;
  double originLon = 0;
  double originAlt = 0;
  double distMax = 0;
  double dist = 0;
  double altMax = -999999;
  double altMin = 999999;
  double spdMax = 0;
  double prevDist = 0;
};

String FinalMessage;

GpsDataState_t gpsState = {};

#define TASK_SERIAL_RATE 1000 // ms
uint32_t nextSerialTaskTs = 0;
uint32_t nextOledTaskTs = 0;

//________________________________________________________________________________

BLEServer* pServer = NULL;
BLECharacteristic* pCharacteristic = NULL;
bool deviceConnected = false;
bool oldDeviceConnected = false;
uint32_t value = 0;
//esp_bd_addr_t* address;
uint8_t* address;

// See the following for generating UUIDs:
// https://www.uuidgenerator.net/

#define SERVICE_UUID        "4fafc201-1fb5-459e-8fcc-c5c9c331914b"
#define CHARACTERISTIC_UUID "beb5483e-36e1-4688-b7f5-ea07361b26a8"

#define LED_PIN 2
#define Forward 13
```

```
#define Forward 13
#define Reverse 12
#define ClosedSensor 32
#define OpenedSensor 33

char remoteAddress[18];
class MyServerCallbacks: public BLEServerCallbacks {
    void onConnect(BLEServer* pServer, esp_ble_gatts_cb_param_t *param) {
      deviceConnected = true;
      BLEDevice::startAdvertising();

    address = param->connect.remote_bda;

      char remoteAddress[18];

    sprintf(
      remoteAddress,
      "%.2X:%.2X:%.2X:%.2X:%.2X:%.2X",
      param->connect.remote_bda[0],
      param->connect.remote_bda[1],
      param->connect.remote_bda[2],
      param->connect.remote_bda[3],
      param->connect.remote_bda[4],
      param->connect.remote_bda[5]
    );
      Serial.println(remoteAddress);
    };

    void onDisconnect(BLEServer* pServer) {
      deviceConnected = false;
    }
};

class MyCallbacks: public BLECharacteristicCallbacks {
    void onWrite(BLECharacteristic *pCharacteristic) {
      std::string value = pCharacteristic->getValue();

      if (value.length() > 0) {
        Serial.println("*********");
        Serial.print("New value: ");
        for (int i = 0; i < value.length(); i++)
          Serial.print(value[i]);

        Serial.println();
        Serial.println("*********");
        if (value== "ON"){
          while (touchRead(ClosedSensor) > 5) {
          digitalWrite (LED_PIN, HIGH);
          digitalWrite (Forward, HIGH);
          digitalWrite (Reverse, LOW);
          }

          digitalWrite (LED_PIN, LOW);
          digitalWrite (Forward, LOW);
          digitalWrite (Reverse, LOW);

        }
```

```
        if (value== "OFF"){
          while (touchRead(OpenedSensor) > 5) {
          digitalWrite (LED_PIN, LOW);
          digitalWrite (Forward, LOW);
          digitalWrite (Reverse, HIGH);
          }

          digitalWrite (LED_PIN, LOW);
          digitalWrite (Forward, LOW);
          digitalWrite (Reverse, LOW);

        }
      }
    }
};
//
static void my_gap_event_handler(esp_gap_ble_cb_event_t  event, esp_ble_gap_cb_param_t* param)
{
    switch (event) {
    case ESP_GAP_BLE_READ_RSSI_COMPLETE_EVT: {
      Serial.println("ESP_GAP_BLE_READ_RSSI_COMPLETE_EVT");
        //m_semaphoreRssiCmplEvt.give((uint32_t) param->read_rssi_cmpl.rssi);
        break;
    }
    }
}

void setup() {
  Serial.begin(115200);

  // GPS _____________________________________________________________
  SerialGPS.begin(9600, SERIAL_8N1, 16, 17);
  while (!EEPROM.begin(EEPROM_SIZE)) {
    true;
  }
  long readValue;
  EEPROM_readAnything(0, readValue);
  gpsState.originLat = (double)readValue / 1000000;

  EEPROM_readAnything(4, readValue);
  gpsState.originLon = (double)readValue / 1000000;

  EEPROM_readAnything(8, readValue);
  gpsState.originAlt = (double)readValue / 1000000;
  // _________________________________________________________________

  //Serial.println("Starting BLE work!");
  pinMode(LED_PIN, OUTPUT);
  pinMode(Forward, OUTPUT);
  pinMode(Reverse, OUTPUT);
  pinMode(ClosedSensor, INPUT);
  pinMode(OpenedSensor, INPUT);
  digitalWrite (LED_PIN, LOW);
  digitalWrite (Forward, LOW);
  digitalWrite (Reverse, LOW);

  BLEDevice::init("ESP32");
```

```
171   BLEServer *pServer = BLEDevice::createServer();
172   pServer->setCallbacks(new MyServerCallbacks());
173
174   BLEService *pService = pServer->createService(SERVICE_UUID);
175   pCharacteristic = pService->createCharacteristic(
176                                          CHARACTERISTIC_UUID,
177                                          BLECharacteristic::PROPERTY_READ |
178                                          BLECharacteristic::PROPERTY_WRITE |
179                                          BLECharacteristic::PROPERTY_NOTIFY
180
181                                        );
182
183   pCharacteristic->addDescriptor(new BLE2902());
184
185   pCharacteristic->setValue("Hello World says Neil");
186   pCharacteristic->setCallbacks(new MyCallbacks());
187   pService->start();
188   // *pAdvertising = pServer->getAdvertising();  // this still is working for backward compatibility
189   BLEAdvertising *pAdvertising = BLEDevice::getAdvertising();
190   pAdvertising->addServiceUUID(SERVICE_UUID);
191   pAdvertising->setScanResponse(true);
192   pAdvertising->setMinPreferred(0x06);  // functions that help with iPhone connections issue
193   pAdvertising->setMinPreferred(0x12);
194   BLEDevice::startAdvertising();
195
196  //  esp_ble_gatts_register_callback(gatts_event_handler);
197   BLEDevice::setCustomGapHandler(my_gap_event_handler);
198   //Serial.println("Characteristic defined! Now you can read it in your phone!");
199 }
200
201 // GPS _____________________________________________________________
202 template <class T> int EEPROM_writeAnything(int ee, const T& value)
203 {
204   const byte* p = (const byte*)(const void*)&value;
205   int i;
206   for (i = 0; i < sizeof(value); i++)
207     EEPROM.write(ee++, *p++);
208   return i;
209 }
210
211 template <class T> int EEPROM_readAnything(int ee, T& value)
212 {
213   byte* p = (byte*)(void*)&value;
214   int i;
215   for (i = 0; i < sizeof(value); i++)
216     *p++ = EEPROM.read(ee++);
217   return i;
218 }
219 // ________________________________________________________________
220
221 void loop() {
222  // GPS _____________________________________________________________
223  static int p0 = 0;
224
225
226   // GPS Koordinaten von Modul lesen
227   gpsState.originLat = gps.location.lat();
```

```
228     gpsState.originLon = gps.location.lng();
229     gpsState.originAlt = gps.altitude.meters();
230
231     // Aktuelle Position in nichtflüchtigen ESP32-Speicher schreiben
232     long writeValue;
233     writeValue = gpsState.originLat * 1000000;
234     EEPROM_writeAnything(0, writeValue);
235     writeValue = gpsState.originLon * 1000000;
236     EEPROM_writeAnything(4, writeValue);
237     writeValue = gpsState.originAlt * 1000000;
238     EEPROM_writeAnything(8, writeValue);
239     EEPROM.commit(); // erst mit commit() werden die Daten geschrieben
240
241     gpsState.distMax = 0;
242     gpsState.altMax = -999999;
243     gpsState.spdMax = 0;
244     gpsState.altMin = 999999;
245
246   if (SerialGPS.available()) {
247       SerialGPS.setTimeout(450);
248       String Message= SerialGPS.readString();
249
250       String GPRMC= Message.substring(Message.indexOf("$GPRMC"),Message.length());
251       GPRMC= GPRMC.substring(0,GPRMC.indexOf('\n'));
252       String GPGGA= Message.substring(Message.indexOf("$GPGGA"),Message.length());
253       GPGGA= GPGGA.substring(0,GPGGA.indexOf('\n'));
254
255       FinalMessage = GPGGA + "\r\n" + GPRMC;
256       Serial.print(FinalMessage.c_str());
257       //Serial.print(Message.c_str());
258   }
259     // ____________________________________________________________
260
261 if (deviceConnected) {
262     //delay(3000);
263     //Serial.println(esp_ble_gap_read_rssi(address));
264 //  char dat;
265 //  if(SerialGPS.available()){
266 //    dat = SerialGPS.read();
267 //    Serial.print(dat);
268 //  }
269 //  if(Serial.available()){
270 //    dat = Serial.read();
271 //    SerialGPS.print(dat);
272 //  }
273   // if (Serial.available()){
274     delay(50);
275     pCharacteristic->setValue(FinalMessage.c_str());
276     pCharacteristic->notify();
277   //}
278 }
279
280 delay(2000);
281 }
282
```

A.2 Programação da Aplicação

A.2.1 app.js

```
import React from "react";
import { makeStyles } from '@material-ui/core/styles';
import Typography from '@material-ui/core/Typography';
import Container from '@material-ui/core/Container';
import Box from '@material-ui/core/Box';
import Leaflet from "./Leaflet";
import OnOffButton from "./OnOffButton";
import BluetoothButton from "./BluetoothButton";
import LocationButton from "./LocationButton";
import ReturnButton from "./ReturnButton"

const pages = {
    HOME: 0,
    MAP: 1,
};

const useStyles = makeStyles((theme) => ({
    root: {
        background: 'linear-gradient(0deg, #ffffff 30%, #6d7c87 90%)',
        border: 0,
        borderRadius: 3,
        color: 'white',
        height: "100%",
        width: "100%",
        padding: '0 30px',
        marginBottom:0,
        marginTop:-8,
        marginLeft:-30,
        marginRight:-10,
      },
}));

export default function App() {
    const classes = useStyles();
    const [tab, setTab] = React.useState(pages.HOME);

    return (
        <Container maxWidth="xl" height="100%">
            <Box my={4} className={classes.root} align="center" style={{paddingTop: '5%' }} >

                <Typography variant="h2" component="h1">
                <b>Locker App</b>
                </Typography>

                {tab === pages.HOME && <React.Fragment>
                    <BluetoothButton/>
                    <OnOffButton />
                    <LocationButton changeToMap={() => setTab(pages.MAP)} />
                </React.Fragment>}

                {tab === pages.MAP && <React.Fragment>
                    <ReturnButton changeToHome={() => setTab(pages.HOME)} />
                    <Leaflet/>
                </React.Fragment>
                }
            </Box>
        </Container>
    );
}
```

A.2.2 BluetoothButton.js

```
import React from 'react';
import clsx from 'clsx';
import { makeStyles } from '@material-ui/core/styles';
import CircularProgress from '@material-ui/core/CircularProgress';
import { green } from '@material-ui/core/colors';
import Button from '@material-ui/core/Button';
import CheckIcon from '@material-ui/icons/Check';
import BluetoothIcon from '@material-ui/icons/Bluetooth';
import Bluetooth from "./Bluetooth-Service"

const useStyles = makeStyles((theme) => ({
  root: {
    //display: 'flex',
    alignItems: 'center',
    paddingTop: '36px' ,
  },
  wrapper: {
    marginBottom: theme.spacing(2),
    position: 'relative',
  },
  buttonSuccess: {
    backgroundColor: green[500],
    '&:hover': {
      backgroundColor: green[700],
    },
  },
  buttonProgress: {
    color: green[500],
    position: 'absolute',
    top: '50%',
    left: '50%',
    marginTop: -12,
    marginLeft: -12,
  },
}));

export default function CircularIntegration() {
  const classes = useStyles();
  const [success, setSuccess] = React.useState(false);
  const [loading, setLoading] = React.useState(false);
  const timer = React.useRef();

  const buttonClassname = clsx({
    [classes.buttonSuccess]: success,
  });

  React.useEffect(() => {
    return () => {
      clearTimeout(timer.current);
    };
  }, []);


  const handleButtonClick = () => {
    if (!loading) {
      setSuccess(false);
      setLoading(true);
      Bluetooth.connect()
        .then(() => {
          setSuccess(true);
          setLoading(false);
        }).catch(() => {
          // alguma coisa correu mal :(
        })

```

```
      // setSuccess(false);
      // setLoading(true);
      // timer.current = window.setTimeout(() => {
      //   setSuccess(true);
      //   setLoading(false);
      // }, 2000);
    }
  };

  /*
    const handleButtonClick = () => {
      bluetooth();
      if (!loading) {
        setSuccess(false);
        setLoading(true);
        timer.current = window.setTimeout(() => {
          setSuccess(true);
          setLoading(false);
        }, 2000);
      }
    };
  */

  return (
    <div className={classes.root}>
      <div className={classes.wrapper}>
        <Button
          variant="contained"
          color="primary"
          className={buttonClassname}
          disabled={loading}
          onClick={handleButtonClick}
        >
          {success ? <CheckIcon /> : <BluetoothIcon />}
          {success ? 'Connected' : 'Bluetooth'}
        </Button>
        {loading && <CircularProgress size={24} className={classes.buttonProgress} />}
      </div>
    </div>
  );
}
```

A.2.3 Bluetooth-Service.js

```
import { DMStoDD } from "./DMStoDD-Service"
import Store from './store';

export default class Bluetooth {
    static device = null;
    static server = null;
    static service = null;
    static bleCharacteristic = null;
    static notifications = null;


    static async connect() {
        const _ = Bluetooth;

        try {
            _.device = await navigator.bluetooth.requestDevice({
                //acceptAllDevices:true,
                filters: [{ services: ["4fafc201-1fb5-459e-8fcc-c5c9c331914b"] }]
            });
            //print("Step 2: Connect to it")
            _.server = await _.device.gatt.connect();
            //device.watchAdvertisements();
            //device.addEventListener('advertisementreceived', GetRSSI);
            //print("Step 3: Get the Service")
            _.service = await _.server.getPrimaryService("4fafc201-1fb5-459e-8fcc-c5c9c331914b");
            //print("Step 4: get the Characteristic")
            _.bleCharacteristic = await _.service.getCharacteristic("beb5483e-36e1-4688-b7f5-ea07361b26a8");

            //print(!!bleCharacteristic.startNotifications);
            _.notifications = await _.bleCharacteristic.startNotifications();
            //print("Step 5: startNotifications")
            _.bleCharacteristic.addEventListener('characteristicvaluechanged', Bluetooth.handleCharacteristicNotifications);

            return Promise.resolve();

        } catch (error) {
            //print( "error: " + error);
            console.error('Connection failed!', error);
        }

        return Promise.reject();
    }

    static Lock() {
        Bluetooth.sendMessage("ON");
    }

    static Unlock() {
        Bluetooth.sendMessage("OFF");
    }

    static sendMessage(message) {
        let encoder = new TextEncoder('utf-8');
        Bluetooth.bleCharacteristic.writeValue(encoder.encode(message));
    }
```

```
56
57      static handleCharacteristicNotifications(event) {
58          let value = event.target.value;
59          let decoder = new TextDecoder('utf-8');
60          const message = decoder.decode(value)
61          console.log(message,value);
62
63          if (message[0] == "$") {
64              const NMEA = message.split("\n");
65              const [, , , , , , , Satellite, ,Altitude] = NMEA[0].split(",");
66              const [,Time, , coordinates1, dir_lat, coordinates2, dir_lon, Velocity, , Date] = NMEA[1].split(",");
67
68              if (coordinates1!=""||dir_lat!=""||coordinates2!=""||dir_lon!=""){
69              const coordinate = DMStoDD(coordinates1, coordinates2, dir_lat, dir_lon);
70              Store.addCoordinate(coordinate);
71              Store.addTime(Time);
72              Store.addDate(Date);
73              Store.addVelocity(Velocity);
74              Store.addSatellite(Satellite);
75              Store.addAltitude(Altitude);
76              console.log(Time, Date, Velocity, Satellite, Altitude);
77              }
78          } else {
79              //print(Message);
80          }
81      }
82  }
```

A.2.4 DMStoDD-Service.js

```
1   export function DMStoDD(C1, C2, Dir_lat,Dir_lon) {
2
3       var a = C1.split(".");
4       var deg_lat = (parseInt(a[0]) / 100).toString().split(".");
5       var lat = parseInt(deg_lat[0]) + parseFloat(deg_lat[1] + '.' + a[1]) / 60;
6
7       if (Dir_lat == 'S') {
8           lat = -lat;
9       }
10
11      var b = C2.split(".");
12      var deg_lon = (parseInt(b[0]) / 100).toString().split(".");
13      var lon = parseInt(deg_lon[0]) + parseFloat(deg_lon[1] + '.' + b[1]) / 60;
14
15      if (Dir_lon == 'W') {
16          lon = -lon;
17      }
18
19      return {
20          lat,
21          lon
22      }
23
24  }
25
26
```

A.2.5 index.js

```
import React from 'react';
import ReactDOM from 'react-dom';

import App from "./App"
import "leaflet/dist/leaflet.css";

/*
const div = document.querySelector('div')
const print = text => {
  div.innerHTML += `<p>${text}</p>`
  console.log(text)
};*/


function GetRSSI(event){
  var rssi= event.rssi;
  //print(rssi);
}

ReactDOM.render(
  <App />,
  document.querySelector('#root')
  );

```

A.2.6 Leaflet.js

```
import React from 'react';
import { MapContainer, Marker, Popup, TileLayer,MapConsumer } from 'react-leaflet';
import 'leaflet/dist/leaflet.css';
import L from 'leaflet';
import Store from './store';
import { view } from "@risingstack/react-easy-state";

delete L.Icon.Default.prototype._getIconUrl;

L.Icon.Default.mergeOptions({
    iconRetinaUrl: require('leaflet/dist/images/marker-icon-2x.png'),
    iconUrl: require('leaflet/dist/images/marker-icon.png'),
    shadowUrl: require('leaflet/dist/images/marker-shadow.png')
});


export default view(function Leaflet() {

  try{
    const coordinate = Store.getLastCoordinate();
    const parsedCoordinate = [coordinate.lat, coordinate.lon];

    return (
      <MapContainer center={[51.505, -0.09]} zoom={13} scrollWheelZoom style={{ height: 400, width: '100%', padding: '5%' }}>
        <TileLayer
          attribution='&copy; <a href="http://osm.org/copyright">OpenStreetMap</a> contributors'
          url="https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png"
        />
        <Marker position={parsedCoordinate}>
          <Popup>
            A pretty CSS3 popup. <br /> Easily customizable.
          </Popup>
        </Marker>
        <MapConsumer>
          {(map) => {
            console.log(map)
            map.setView(parsedCoordinate);
            return null;
          }}
        </MapConsumer>
      </MapContainer>
    );
   }
  catch(error){
    alert('GPS location is not accurate at the moment! \nWait for calibration!')
```

```
    return (
      <MapContainer center={[51.505, -0.09]} zoom={13} scrollWheelZoom style={{ height: 400, width: '100%', padding: '5%' }}>
        <TileLayer
          attribution='&copy; <a href="http://osm.org/copyright">OpenStreetMap</a> contributors'
          url="https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png"
        />
        <Marker position={[51.505, -0.09]}>
          <Popup>
            A pretty CSS3 popup. <br /> Easily customizable.
        </Popup>
        </Marker>
        <MapConsumer>
          {(map) => {
            console.log(map)
            map.setView([51.505, -0.09]);
            return null;
          }}
        </MapConsumer>
      </MapContainer>
    );
  }
});
```

A.2.7 LocationButton.js

```
import React from 'react';
import Button from '@material-ui/core/Button';
import { makeStyles } from '@material-ui/core/styles';
import ExploreIcon from '@material-ui/icons/Explore';

const useStyles = makeStyles((theme) => ({
  button: {
    margin: theme.spacing(2),
  },
}));


export default function IconLabelButtons(props) {
    const classes = useStyles();
    return (
      <div>
        <Button
          variant="contained"
          color= "primary"
          className={classes.button}
          startIcon={<ExploreIcon/>}
          onClick={props.changeToMap}
        >
          Location
        </Button>
        {/* This Button uses a Font Icon, see the installation instructions in the Icon component docs. */}
      </div>
    );
  }

```

A.2.8 OnOffButton.js

```
import React from 'react';
import ButtonGroup from '@material-ui/core/ButtonGroup';
import Button from '@material-ui/core/Button';
import LockOpenIcon from '@material-ui/icons/LockOpen';
import LockIcon from '@material-ui/icons/Lock';
import {Lock} from "./OnOff-Service"
import {Unlock} from "./OnOff-Service"

export default function DisableElevation() {
    return (
      <ButtonGroup disableElevation variant="contained" color= "default" style={{paddingTop: '2px' }}>
        <Button color= "primary" startIcon={<LockIcon/>} onClick={() => {  Lock() }} >Lock</Button>
        <Button color= "secondary" startIcon={<LockOpenIcon/>} onClick={() => {  Unlock() }} >Unlock</Button>
      </ButtonGroup>
    );
  }

```

A.2.9 OnOff-Service.js

```
import Bluetooth, { bleCharacteristic } from "./Bluetooth-Service"


export function Lock() {
  //print("ON");
  Bluetooth.Lock();

}

export function Unlock() {
  //print("OFF");
  Bluetooth.Unlock();

}

```

A.2.10 ReturnButton.js

```
import React from 'react';
import { makeStyles } from '@material-ui/core/styles';
import IconButton from '@material-ui/core/IconButton';
import ArrowBackIcon from '@material-ui/icons/ArrowBack';
import { grey } from '@material-ui/core/colors';

const useStyles = makeStyles((theme) => ({
    margin: {
        color: grey[50],
        position: 'absolute',
        top: 2,
        left: 2,
    },
}));

export default function ButtonSizes(props) {
    const classes = useStyles();
    return (
        <div>
            <IconButton
            aria-label="delete"
            className={classes.margin}
            size="medium"
            onClick={props.changeToHome}
            >
            <ArrowBackIcon fontSize="inherit" />
            </IconButton>
        </div>
    );
}

```

A.2.11 store.js

```
import { store } from "@risingstack/react-easy-state";

const myStore = store({
    Satellites: [],
    addSatellite(Satellite) {
        this.Satellites.push(Satellite);
    },
    getLastSatellite() {
        return this.Satellites[this.Satellites.length - 1];
    },
    Altitudes: [],
    addAltitude(Altitude) {
        this.Altitudes.push(Altitude);
    },
    getLastAltitude() {
        return this.Altitudes[this.Altitudes.length - 1];
    },
    Dates: [],
    addDate(Date) {
        this.Dates.push(Date);
    },
    getLastDate() {
        return this.Dates[this.Dates.length - 1];
    },
    Times: [],
    addTime(Time) {
        this.Times.push(Time);
    },
    getLastTime() {
        return this.Times[this.Times.length - 1];
    },
    Speeds: [],
    addVelocity(Velocity) {
        this.Speeds.push(Velocity);
    },
    getLastVelocity() {
        return this.Speeds[this.Speeds.length - 1];
    },
    coordinates: [],
    addCoordinate(coordinate) {
        this.coordinates.push(coordinate);
    },
    getLastCoordinate() {
        return this.coordinates[this.coordinates.length - 1];
    }
});

export default myStore;

```

A.2.12 index.html

```
<html>
  <head>
    <!-- Fonts to support Material Design -->
    <link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Roboto:300,400,500,700&display=swap" />
    <!-- Icons to support Material Design -->
    <link rel="stylesheet" href="https://fonts.googleapis.com/icon?family=Material+Icons" />
  </head>
  <body>

    <div id="root"></div>

    <script src="./index.js"></script>
  </body>

  </html>

```

A.3 Dimensionamento Mecânico

Tipo de Bronze	Tensão σ_{b2Lim} (MPa)
Bronze vazado (molde de areia)	50
Bronze vazado em coquilha	60
Bronze fosforoso centrifugado	70
Bronze de alumínio/manganésio centrifugado	75

Figura A.1: Imagem da tabela da tensão limite de fadiga (σ_{b2Lim}), para bronzes da roda [25].

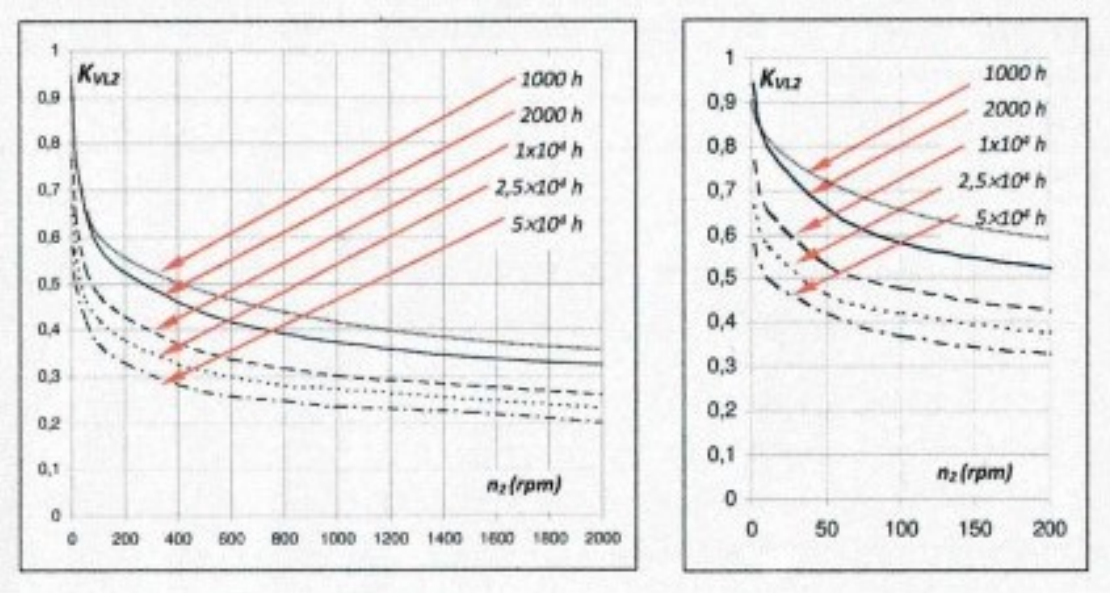


Figura A.2: Imagem com gráfico do fator de velocidade K_{vL2} [25].

Material da roda Bronze fosforoso	Aço de liga (cementação e retificação)	Aço ao carbono temperado (tensão σ_u limite: 850-900MPa)
Vazado em molde de areia	3,62	2,85
Vazado em coquilha	4,20	3,45
Bronze de *Al/Mn* centrifugado	5,75	4,50

Figura A.3: Imagem de tabela com os fatores de contacto, Ω_{02}[25].

Duração e horas (h)	KL
≥ 50000 h	0.8
25000	1,00
10000	1,25
4000	1,38
2000	1.5
1000	1.8
500	2,2

Figura A.4: Imagem de tabela com os fatores de duração, K_L [25].