Universidade da Beira Interior
Departamento de Informática

Nº 1 - 2024: *Internet das Coisas - Pet Feeder*

Elaborado por:

Juscélio Reis

Rene Jerez

Professor Doutor Bruno Silva

15 de abril de 2024

Conteúdo

Lista de Figuras

Lista de Tabelas

Acrónimos

Capítulo

1

Introdução

1.1 Enquadramento

Este documento enquadra-se na cadeira de Internet das Coisas, do primeiro ano de mestrado de engenharia informática.

1.2 Motivação

De acordo com o projecto Mars um em cada três animais de companhia estão em situação sem abrigos. Em Portugal, de acordo com D'Avila (2016) a população de gatos era estimada em 991 000 indivíduos, entretanto não há estimativa da quantidade de gatos errantes. Nota-se o crescimento desta população errante e também as dificuldades dos gatos em encontrarem alimentos confiáveis.

1.3 Objetivos

O objetivo deste trabalho é criar caixas alimentadoras seguras para gatos errantes da zona. Este projeto visa fornecer fontes de alimentação consistentes e seguras para estes animais, que frequentemente enfrentam fome e desnutrição. A implementação dessas caixas garantirá acesso a comida nutritiva em um ambiente seguro, melhorando significativamente suas vidas.

1. Este sistema integra vários sensores na caixa alimentadora, como também deve manter-se ativo em ambiente de rua daí que foi necessário recorrer a energia solar e modulo arduino integrado com Sim Card.

2. Através de uma Representational state transfer application programming interface (REST) application programming interface (API), enviar os dados do ESP8266 para um servidor remoto.

3. Disponibilizar uma aplicação para o telemovel e um dashboard online que permite usadores visualizar os dados em tempo real e dados históricos.

1.4 Organização do Documento

De modo a refletir o trabalho que foi feito, este documento encontra-se estruturado da seguinte forma:

1. O primeiro capítulo – **Introdução** – apresenta o projeto, a motivação para a sua escolha, o enquadramento para o mesmo, os seus objetivos e a respetiva organização do documento;

2. O segundo capítulo – **Estado da Arte** – faz um estudo a implementatações ou investigações já existentes acerca do assunto em questão e procura encontrar uma possível implementação que não tenha sido diretamente aplicada por estes, mas sim complementando-os;

3. O terceiro capítulo – **Engenharia de Software e Arquitetura de Sistema** – detalha os diversos aspetos de engenharia de software relacionados com o presente trabalho. É nesta secção onde são criadas personas e user stories responsáveis pela criação dos requisitos funcionais e não funcionais do sistema. Após isso são mostrados os diagramas de engenharia de software e da arquitetura do sistema. Por último são elaboradas as tecnologias e materiais utilizados para o desenvolvimento do projeto.

Capítulo

2

Estado da Arte

Neste segmento, é feita uma exploração das investigações e estudos no campo, incluindo as metodologias e ferramentas predominantes. O propósito é construir uma visão abrangente do conhecimento atual, possibilitando a detecção de lacunas que poderiam ser exploradas através de novas investigações ou inovações tecnológicas. Aborda-se a análise de três casos ou projetos preexistentes, seguida da introdução de nossa proposta, que visa agregar valor às abordagens já existentes. Uma tabela comparativa é mostrada para facilitar a visualização das diferenças e semelhanças entre nossa proposta e as ideias prévias.

2.1 Soluções Existentes

2.1.1 Primeira Solução - Cat Feeding

Uma solução para alimentar gatos utilizando dispositivos ativados pela Internet das Coisas (IoT). O dispositivo utiliza um microcontrolador Arduino Uno combinado com um sensor TCS3200, RTCDS3231 (relógio em tempo real), célula de carga, HX711 e ESP32CAM para facilitar a alimentação remota e a monitorização das actividades do gato. Também foi utilizada a aplicação Telegram para enviar notificações e imagens ao dono, mantendo-o informado sobre os níveis de comida e as actividades do seu gato.

O sistema foi concebido para aliviar as preocupações dos donos de gatos relativamente aos horários de alimentação dos seus animais de estimação, especialmente quando estão fora de casa. Automatiza o processo de alimentação, fornece actualizações em tempo real e até permite que os donos verifiquem visualmente os seus animais de estimação através de imagens enviadas para

Figura 2.1: Design results of the cat feed

os seus smartphones. A integração destes componentes técnicos cria uma solução abrangente para o tratamento remoto de gatos, realçando as aplicações práticas das tecnologias IoT na vida quotidiana.

2.1.2 Segunda Solução - Automatic Cat Feeding Monitoring

Esta solução apresenta um sistema automático de alimentação de gatos utilizando o microcontrolador Arduino Mega 2560 e o sensor HC-SR04, integrado com capacidades IoT. Destinado a donos de animais de estimação que procuram soluções eficientes e remotas para cuidar de animais de estimação, o sistema garante horários e porções de alimentação adequados para evitar desnutrição ou obesidade em gatos.
O estudo segue a abordagem do ciclo de vida de desenvolvimento do sistema (SDLC), abrangendo as fases de recolha de requisitos, validação, formação e propriedade do sistema. A conceção do sistema inclui também uma bateria recarregável para um funcionamento contínuo durante as falhas de energia, garantindo a fiabilidade. As fases de teste validaram a funcionalidade dos componentes individuais e do sistema integrado, confirmando a sua eficácia na manutenção das rotinas de alimentação programadas e no controlo das porções. Além disso, as soluções foram pensadas para que, no futuro, as melhorias pudessem incluir a utilização de microcontroladores alternativos e a melhoria das características do sistema para aplicações mais vastas.

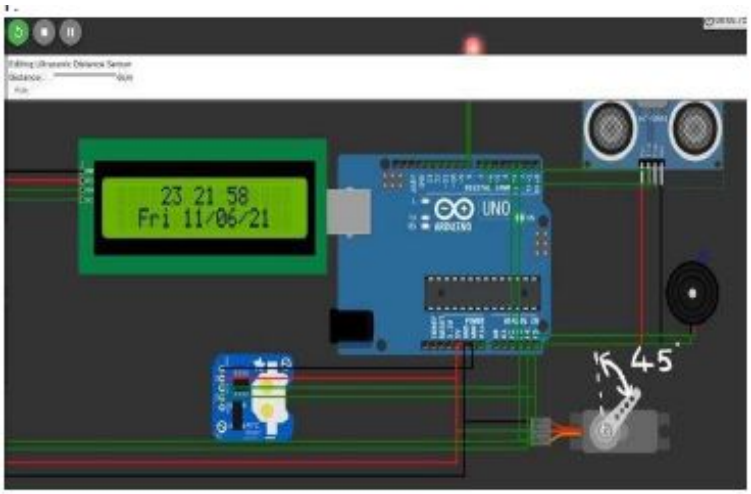

Figura 2.2: Simulated Output when distance is 6 cm from bottom or top

2.1.3 Terceira Solução - Automated Pet Feeder

Esta solução desenvolveu algumas funcionalidades de um alimentador automático de animais de estimação utilizando tecnologias da Internet das Coisas (IoT), concebido para melhorar os cuidados com os animais de estimação, fornecendo horários de alimentação atempados e adequados.

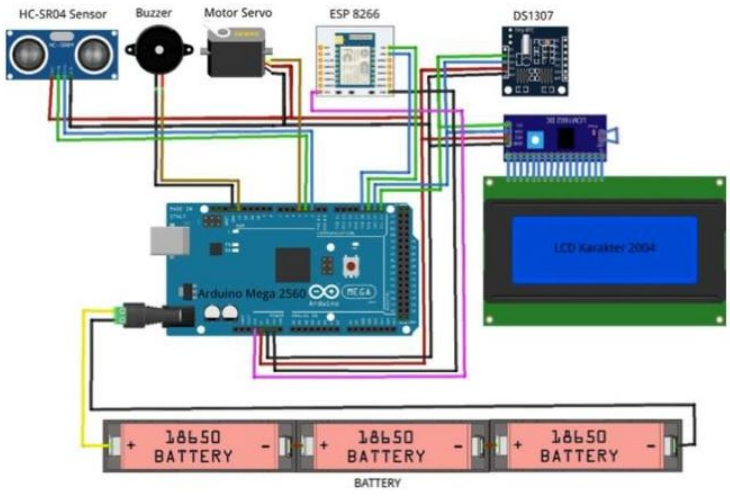

Figura 2.3: Pet Feeder System Circuit

Este sistema tem como objetivo resolver problemas comuns como a des-

nutrição e a obesidade em animais de estimação, especialmente quando os donos estão ausentes. Foram utilizados o Arduino Uno, o módulo RTC, o sensor de distância e o servo-motor SG90.
O alimentador é capaz de ajustar os tempos de alimentação, os intervalos entre as alimentações e a quantidade de ração dispensada. Foi concebido para combater os riscos de sobrealimentação e subalimentação, ajudando assim a controlar o peso dos animais de estimação.
As futuras melhorias poderão incluir técnicas de monitorização mais precisas e capacidades de interação remota alargadas.

2.1.4 Quarta Solução - Dogs Feed Smart System

Neste caso, a solução foi concebida para alimentar e hidratar cães de estimação à distância. Este sistema permite que os donos mantenham horários regulares de alimentação através de um smartphone, garantindo que os animais de estimação são bem tratados, mesmo quando os donos estão fora.
O sistema integra hardware como um Arduino UNO com ESP8266-01 para conetividade WiFi, permitindo a comunicação em tempo real com a Internet.
Inclui um RTC para programação, uma célula de carga para pesar os alimentos e um sensor ultrassónico para monitorizar os níveis de comida e água, garantindo um controlo preciso da alimentação e da hidratação.
Os modos de funcionamento incluem definições manuais e automáticas.

Figura 2.4: Pet Feeder System Circuit

O modo automático funciona com base numa programação definida através

do RTC, automatizando a alimentação a horas específicas. O modo manual, acessível através da aplicação Blynk, o qual permite aos proprietários ajustar as definições em tempo real, o que é útil para quem tem horários variáveis. A aplicação Blynk actualiza os proprietários sobre os níveis de comida e água e permite o controlo manual dos mecanismos de alimentação. Foi observada uma elevada precisão, particularmente com o sensor Load Cell, que garante porções de alimentação correctas para evitar a sobrealimentação.

2.1.5 Quinta Solução - Pet Care System

Este projecto apresenta um sistema integrado de cuidados para animais de companhia que utiliza placas Arduino Uno e módulos Wi-Fi ESP8266 para monitorizar e controlar a alimentação, o abeberamento e a gestão da cama dos animais de estimação. Este sistema permite que os donos de animais de estimação tratem das necessidades básicas dos seus animais de estimação remotamente através de uma aplicação abrangente para smartphone, fornecendo funcionalidades como o acompanhamento do consumo e a monitorização da utilização da caixa de areia para detetar potenciais problemas de saúde.

Figura 2.5: Pet Care System

O sistema integra todos os aspectos essenciais numa única plataforma. A aplicação associada não só apresenta estatísticas detalhadas sobre as actividades do animal de estimação, como também permite o controlo manual dos

subsistemas para ocasiões especialmente quando os proprietários estão frequentemente ocupados ou fora de casa.

2.1.6 Comparativos

Observando todas as soluções é possível fazer um comparativo, consoante as principais caracteristicas.

Componentes	1ª Solução	2ª Solução	3ª Solução	4ª Solução
Microcontroller	Arduino Uno	Arduino Uno	Arduino Uno	Arduino Uno
Comunicação	ESP32CAM	ESP8266 WiFi	ESP8266 WiFi	ESP8266-01
Sensores	RTC, Load Cell, Servo Motor, TCS3200, Camera	RTC, Load Cell, Servo Motor, Distance Sensor	RTC, Load Cell, Servo Motor, Ultrasonic	RTC, Load Cell, Servo Motor, Ultrasonic
Função	Monitorização alimentação	Programação monitorização alimentação	Monitorização alimentação	Gestão abeberamento alimentação
Interface	Telegram App	Blynk App	Blynk App	Blynk App

Um tema importante para o projeto será ter em mente os conceitos de smart cities e smart home.
De acordo com Abdulsattar, Nejood Faisal e outros (2022), Uma cidade inteligente utiliza tecnologias da Internet das Coisas (IoT) para melhorar a eficiência e a eficácia em vários domínios, como a energia, os transportes, a saúde e as infra-estruturas. Estas cidades integram dispositivos electrónicos e tecnologias digitais para facilitar uma vida sustentável, o conforto e a produtividade sem intervenção humana, com o objetivo de transformar os ambientes urbanos a nível global através da integração e gestão de dados sem descontinuidades.
Já para as casas inteligentes amaluddin, Hasmida e outros (2022), comentam que há integração de tecnologias da Internet das Coisas (IoT) para permitir o controlo automático de aparelhos e dispositivos, aumentando a conveniência, o conforto e a eficiência energética. Os utilizadores podem gerir remotamente os seus ambientes domésticos a partir de qualquer lugar através de dispositivos móveis ligados à Internet, tornando as actividades diárias mais simples e mais eficientes.
Assim podemos fazer um comparativo com os aspectos importantes de cada tema:

Aspecto	Cidades Inteligentes	Casas Inteligentes
Foco e Escopo	Melhorar a vida urbana em larga escala; abordar infraestrutura, energia, transporte, saúde e serviços.	Focar em residências individuais; melhorar conveniência, segurança e eficiência energética.
Aplicações Tecnológicas	Utilizam extensivamente IoT para gestão baseada em dados de sistemas urbanos como redes inteligentes e transporte.	Utilizam IoT para automatizar e controlar sistemas domésticos como iluminação, aquecimento e segurança.
Desafios	Gerenciar e proteger grandes quantidades de dados, integrar sistemas diversos, garantir segurança e escalar soluções.	Garantir privacidade do usuário, integrar dispositivos de forma amigável ao usuário e manter conectividade confiável.
Objetivos	Melhorar eficiência, sustentabilidade e qualidade de vida; otimizar recursos e aprimorar conectividade.	Aumentar conforto, conveniência e eficiência energética; proporcionar controle fácil sobre os ambientes domésticos.
Escala e Impacto	Afetam comunidades inteiras e sistemas urbanos.	Focam em espaços de vida individuais.
Complexidade de Integração	Mais complexa devido à diversidade de sistemas e serviços.	Envolve principalmente a integração de dispositivos domésticos.
Gestão de Dados e Segurança	Escopo e implicações mais amplos para a segurança de dados.	Focado na segurança de dados pessoais dentro da casa.
Objetivos Finais	Também focam em sustentabilidade e desenvolvimento econômico.	Mais focados em conveniência pessoal e gestão.

Tabela 2.1: Comparação entre Cidades Inteligentes e Casas Inteligentes

Capítulo

3

Engenharia de Software e Arquitetura de Sistema

3.1 Introdução

Este capítulo explora a aplicação da Engenharia de Software no desenvolvimento do sistema. Na secção 3.2, são introduzidas as personas e as user stories para a definição dos requisitos que caracterizam o sistema, descritos na secção 3.3. A secção 3.4 detalha as várias tecnologias empregadas no desenvolvimento da aplicação. As secções 3.5 e 3.6 ilustram, respectivamente, os diagramas de casos de uso e de atividades para as principais funcionalidades do sistema. A arquitetura do sistema implementada é revelada na secção 3.7. Finalmente, a secção 3.8 apresenta a estrutura da base de dados.

3.2 Personas

De acordo com Pruit e Adlin (2006), uma persona é uma personagem fictícia criada para representar um tipo de utilizador que pode utilizar um site, uma marca ou um produto de forma semelhante. As personas são utilizadas no processo de design centrado no utilizador para ajudar os designers e as partes interessadas a compreender e a simpatizar com os objectivos, experiências, comportamentos e necessidades dos utilizadores finais. Ao criar perfis detalhados de utilizadores imaginários, incluindo os seus dados demográficos, motivações e ambiente, as personas fornecem um modelo tangível que ajuda a orientar as decisões sobre as características do produto, as interacções e o design visual.
As personas são uma ferramenta fundamental no design UX e são cruciais

para garantir que as decisões de design visam necessidades reais e relevantes dos utilizadores. Ajudam a concentrar os esforços num grupo de utilizadores definido, permitindo que todas as decisões sejam tomadas para um público conhecido e não para um público autorreferencial ou especulativo.

3.2.1 Persona 1: O gato sem abrigo

Nome : Felix
Idade: 3 anos
Situação: Sem abrigo
Descrição: Felix é um gato vira-lata que vive nas ruas da Covilhã. Ele passa seus dias procurando por comida, água limpa e um abrigo seguro para proteger-se do clima e de perigos. Felix é cauteloso, mas curioso, explorando novas áreas em busca de recursos. Ele frequentemente depende da bondade de estranhos para alimentação e às vezes encontra refúgio temporário em locais como porões, garagens, e áreas subterrâneas. Felix é esperto, mas a vida nas ruas é dura e imprevisível. Ele precisa de soluções que possam garantir sua sobrevivência diária de maneira mais segura e estável.

3.2.2 Persona 2: O Veterinário Humanitário

Nome : Dra. Laura Mendes
Idade: 28 anos
Profissão: Veterinária
Descrição: Dr. Laura é uma veterinária que trabalha em uma clínica local, mas também cuida de uma colônia de gatos de rua na Avenida Biribau. Ela visita a colônia semanalmente para fornecer comida, água e cuidados médicos básicos. Ela trabalha em colaboração com organizações de resgate locais para tentar encontrar lares permanentes para os gatos mais dóceis, enquanto os mais selvagens são castrados ou esterilizados para controlar a população. Dra. Laura também promove a conscientização sobre a importância do cuidado com os animais de rua, liderando iniciativas educacionais e de arrecadação de fundos.

3.2.3 Persona 3: O Apoiador Solidário

Nome : Rui Teixeira
Idade: 30 anos
Profissão: Informático
Descrição: Rui é um informático bem-sucedido que, embora não tenha pets

em casa devido a alergias, possui uma grande paixão por ajudar animais em necessidade. Ele é um doador ativo e voluntário em várias organizações de resgate de animais, especialmente aquelas focadas em gatos e pets abandonados. Rui usa sua rede profissional e recursos financeiros para apoiar eventos de arrecadação de fundos e campanhas de adoção. Rui frequentemente compra e doa alimentos, medicamentos e suprimentos para abrigos locais. Ele é conhecido na comunidade por sua generosidade e compromisso com o bem-estar animal, e muitas vezes motiva outras pesssoas a se envolverem na causa.

3.3 User Stories

Com base em Cohn (2004) As histórias de usuário são descrições breves e claras de uma funcionalidade vista pelo olhar de quem a deseja, geralmente um usuário ou cliente do sistema. Elas seguem o formato: "Como [tipo de usuário], quero [objetivo] para que [motivo]".
Essenciais em metodologias Agile e Scrum, as histórias de usuário focam nas necessidades dos usuários e na entrega de valor, incentivando uma abordagem flexível no desenvolvimento de software. Elas incluem critérios de aceitação que ajudam a definir e verificar o cumprimento dos requisitos.

- Como Felix, quero identificar fontes confiáveis de água limpa, para que eu possa me manter hidratado e saudável.
- Como Felix, quero encontrar pessoas amigáveis que estejam dispostas a alimentar gatos de rua, para que eu possa ter refeições regulares sem ter que revirar lixeiras.
- Como Felix, quero encontrar locais onde possa receber cuidados veterinários gratuitos ou de baixo custo, para manter minha saúde em dia sem grandes custos.
- Como Felix, quero poder identificar rapidamente rotas de fuga e esconderijos seguros, para que eu possa me esquivar de cães e outros perigos nas ruas.
- Como Dra. Laura Mendes, desejo trabalhar com autoridades locais para garantir que as áreas onde os gatos vivem sejam seguras e livres de ameaças, como envenenamento ou maus-tratos.
- Como Dra. Laura Mendes, desejo implementar um sistema de identificação para os gatos da colônia, como colares ou microchips, para facilitar o monitoramento e identificação de cada animal.

- Como Dra. Laura Mendes, quero criar uma rede de voluntários que possam ajudar nas atividades diárias de cuidado com a colônia de gatos, para garantir que os gatos recebam atenção e cuidado constantes.

- Como Dra. Laura Mendes, quero manter um registro atualizado da saúde e status de cada gato na colônia, para que eu possa monitorar a eficácia dos tratamentos e a evolução da saúde deles ao longo do tempo.

- Como Dra. Laura Mendes, quero ter acesso fácil a suprimentos médicos e de alimentação para cuidar da colônia de gatos, garantindo que eu possa prover as necessidades básicas dos gatos durante minhas visitas semanais.

- Como Rui Teixeira, quero identificar organizações de resgate de animais que necessitam de suporte, para que eu possa direcionar adequadamente meus recursos e esforços de voluntariado.

- Como Rui Teixeira, desejo implementar sistemas de gestão para organizações de resgate que melhorem sua eficiência operacional, usando minhas habilidades em informática para ajudar na organização e no armazenamento de dados.

- Como Rui Teixeira, desejo estabelecer uma rede de contatos com fornecedores de alimentos e medicamentos para animais, para conseguir doações ou descontos significativos que possam beneficiar os abrigos.

- Como Rui Teixeira, quero criar um aplicativo móvel que facilite doações diretas para abrigos e organizações de resgate, permitindo que doadores contribuam de forma rápida e segura.

3.4 Requisitos de Utilizador

Esta seção dos requisitos do usuário explica o que os usuários esperam do nosso sistema. Os requisitos funcionais descrevem as funcionalidades específicas do sistema, enquanto os requisitos não funcionais descrevem questões como desempenho, segurança e usabilidade.

A probabilidade de erros e ajustes durante o processo de desenvolvimento é reduzida com precisão, segundo [1]. O design, o desenvolvimento e o teste do software ficam mais fáceis quando os requisitos do usuário são claros. Isso garante uma experiência de usuário satisfatória e um produto funcional.

3.4.1 Requisitos Funcionais

As funções que o sistema deve ser capaz de executar são especificadas pelos requisitos funcionais. Os requisitos funcionais encontrados para a aplicação em desenvolvimento foram divididos em dois grupos: o Aplicativo e o sistema. Esses grupos incluem:

3.4.1.1 Aplicativo

- RF1: O aplicativo deve ter um portal de login, permitindo a visualização dos dados recolhidos pelo sistema.
- RF2: O aplicativo deve permitir aos usuários cadastrar e configurar perfis para múltiplos animais de estimação.
- RF3: O aplicativo deve fornecer notificações em tempo real sobre o status da alimentação e qualquer alerta de sistema.
- RF4: O aplicativo deve permitir aos usuários programar horários automáticos de alimentação para cada pet cadastrado.
- RF5: O aplicativo deve permitir a visualização do histórico de alimentação dos pets.
- RF6: O aplicativo deve oferecer suporte para a visualização e controle de múltiplas unidades de alimentadores conectadas.
- RF7: O aplicativo deve oferecer uma seção de dicas e cuidados veterinários personalizados, baseados nos dados dos pets.
- RF8: O aplicativo deve permitir que os usuários compartilhem informações do pet e dados de alimentação com veterinários ou cuidadores através da plataforma.

3.4.1.2 Sistema

- RF1-s: O sistema deve transmitir os dados através de REST API, recorrendo ao protocolo hyper text transfer protocol (HTTP) sobre TCP/IP.
- RF2-s: O sistema deverá permitir a introdução de eventos na base de dados.
- RF3-s: O sistema deve armazenar na base de dados um log com eventos que sejam considerados alarmantes.

- RF4-s: O sistema deve validar as credenciais do dispositivo antes de permitir acesso aos dados sensíveis ou à configuração do sistema.
- RF5-s: O sistema deve prover uma interface de administração para a gestão de usuários e permissões de acesso.
- RF6-s: O sistema deverá implementar medidas de segurança robustas para proteger contra acessos não autorizados e ataques cibernéticos.
- RF7-s: O sistema deve registrar todas as operações críticas executadas pelos usuários para fins de auditoria e conformidade.
- RF8-s: O sistema deve permitir a monitorização em tempo real do estado e desempenho dos dispositivos IoT conectados.

3.4.2 Requisitos Não Funcionais

Os requisitos não funcionais descrevem questões como desempenho, segurança e usabilidade.

- RNF1 – Eficiência: O software deve estar implementado com vista à sua eficiência, realizando tarefas unicamente quando solicitado, tais como acessar à base de dados ou enviar avisos.
- RNF2 – Escalabilidade: O sistema deve ser projetado para suportar o aumento no número de dispositivos IoT e usuários sem degradação de desempenho.
- RNF3 – Disponibilidade: O sistema deve garantir uma disponibilidade mínima de 90%, permitindo acesso contínuo aos usuários e dispositivos conectados.
- RNF4 – Segurança: O sistema deve implementar protocolos de segurança de dados atualizados, incluindo criptografia de dados em trânsito e em repouso, e autenticação multifatorial.
- RNF5 – Usabilidade: A interface do usuário, tanto no aplicativo móvel quanto no dashboard web, deve ser intuitiva e fácil de usar, permitindo que os usuários naveguem e operem o sistema sem treinamento prévio extensivo.
- RNF6 – Confiabilidade: O sistema deve ter uma taxa de falhas muito baixa e ser capaz de recuperar-se rapidamente de falhas sem perda de dados.

- RNF7 – Manutenibilidade: O código do sistema deve ser bem documentado e estruturado para facilitar atualizações e manutenção por desenvolvedores, com suporte para diagnóstico e resolução de problemas.
- RNF8 – Portabilidade: O software deve ser projetado para ser facilmente adaptável para diferentes plataformas e sistemas operacionais, especialmente para o aplicativo móvel que deve funcionar tanto em iOS quanto em Android.
- RNF9 – Interoperabilidade: O sistema deve ser capaz de integrar-se de forma eficaz com sistemas externos e APIs, mantendo a compatibilidade com diferentes formatos e padrões de dados.
- RNF10 – Testabilidade: O sistema deve suportar testes automatizados e manuais, permitindo verificações frequentes de sua funcionalidade e desempenho para garantir que todos os componentes estejam operando conforme esperado.

3.5 Tecnologias utilizadas

Este tópico divide os materiais e tecnologias usados na construção do projeto em três partes principais: o alimentador, a infraestrutura do servidor e o aplicativo móvel.

3.5.1 Tecnologias utilizadas no alimentador

No alimentador teremos os seguintes itens:

- Parte Exterior do Alimentador
  - Camera
  - Células de energia solar
- Parte Interior do Alimentador
  - NodeMcu Sim Card
  - Pilhas recarregaveis compativeis com energia solar
  - Sensor de Temperatura e Humidade
  - Mini servo motor
  - Sensor Ultrasônico
  - Camera

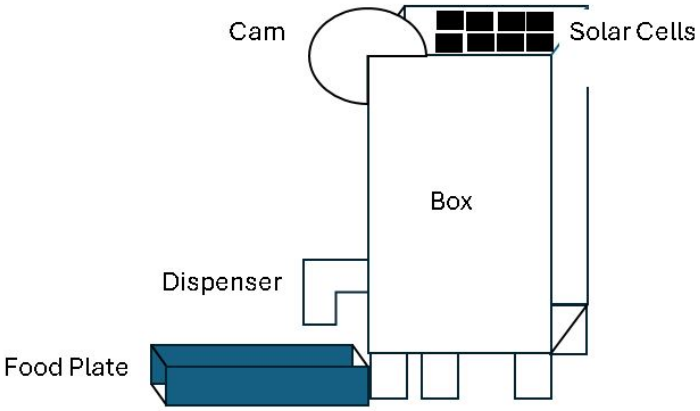

Figura 3.1: Visão Externa da Caixa

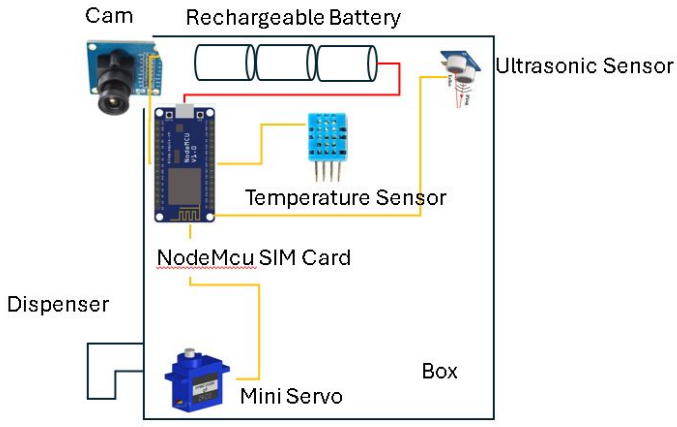

Figura 3.2: Visão Interna da Caixa

3.5.2 Tecnologias utilizadas na infraestrutura do servidor

O servidor funciona com o modelo de plataforma como serviço (PaaS) do Azure App Service e usa o.Net 8 para administrar os dados dos alimentadores armazenados na base de dados e gerenciar os aplicativos móveis que permitem visualizar informações sobre todos os dispositivos. O plano de serviço B1 do Azure App Service gerencia este sistema e oferece recursos como 1,75 GB de memória RAM, 10 GB de armazenamento de aplicações, escala automática e tráfego HTTP/S.

As tecnologias utilizadas para a elaboração da base de dados e do aplicativo foram:

- .Net Core 8.0
- Json
- SQL Azure
- Azure App Service
- Python

3.5.3 Tecnologias utilizadas no aplicativo móvel

O aplicativo móvel do projeto foi desenvolvido usando a tecnologia React Native, uma ferramenta atual e eficiente desenvolvida pelo Facebook. Utilizando esta opção, é possível criar interfaces de usuário atraentes e funcionais que se adaptam ao iOS e ao Android usando um único conjunto de código. A capacidade do React Native de combinar a rapidez do desenvolvimento web com o desempenho das aplicações nativas é um dos seus pontos fortes.

As tecnologias utilizadas para a elaboração do aplicativo móvel foram:

- React Native
- Node
- Android

3.6 Diagramas de Casos de Uso

Em UML (Linguagem de Modelagem Unificada), um caso de uso define uma funcionalidade particular que um sistema oferece em resposta às interações com os usuários ou sistemas externos. O livro [2] enfatiza que os casos de uso

são "uma técnica eficaz para encontrar e documentar requisitos funcionais, ajudando todos os envolvidos a entender de forma clara e sistemática as funcionalidades necessárias do sistema".Os diagramas de casos de uso ajudam os desenvolvedores e analistas a ver a interação entre os atores e o sistema. Isso facilita a compreensão dos requisitos e garante que as necessidades do usuário sejam atendidas adequadamente.

3.6.1 Casos de Uso: Gestão de Alimentação

Atores:

- Apoiador Solidário: O usuário que interage com o aplicativo para gerenciar a alimentação.

Casos de Uso:

- Programar alimentação automática: O Apoiador Solidário configura horários e quantidades para alimentação automática dos pets.
- Ajustar a quantidade de alimento dispensado: O Apoiador Solidário modifica as configurações de quantidade de alimento que cada pet deve receber.
- Visualizar e gerenciar o histórico de alimentação: O Apoiador Solidário acessa o histórico de alimentações, podendo visualizar e gerenciar os registros passados.

O Apoiador Solidário está diretamente ligado a cada um dos casos de uso acima, indicando que ele inicia e interage com essas funcionalidades no aplicativo.

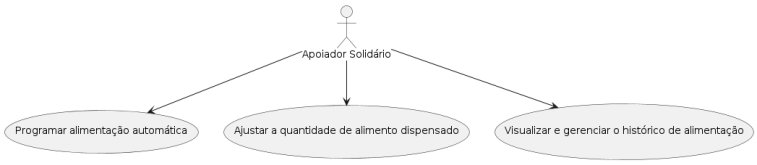

Figura 3.3: Gestão de Alimentação

3.6.2 Casos de Uso: Monitoramento e Alertas

Atores:

- Apoiador Solidário: O usuário que utiliza o aplicativo móvel para receber informações sobre o alimentador.

Casos de Uso:

- Receber notificações sobre o estado do alimentador: O Apoiador Solidário recebe atualizações automáticas sobre o estado atual do alimentador através do aplicativo móvel.
- Consultar alertas de manutenção ou problemas técnicos: O Apoiador Solidário pode acessar informações sobre quaisquer questões técnicas ou de manutenção que necessitem atenção, também através do aplicativo móvel.

O Apoiador Solidário está diretamente ligado a cada um dos casos de uso acima, indicando que ele inicia e interage com essas funcionalidades por meio do aplicativo móvel.

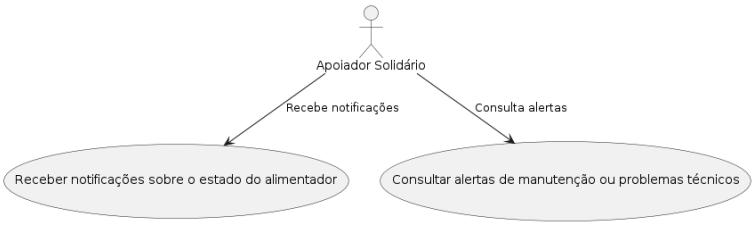

Figura 3.4: Monitoramento e Alertas

3.6.3 Casos de Uso: Gerenciamento de Dispositivos

Atores:

- Apoiador Solidário: Representa o usuário que realiza operações de configuração e gestão de dispositivos no aplicativo.

Casos de Uso:

- Adicionar e configurar novos dispositivos alimentadores: O Apoiador Solidário pode adicionar novos dispositivos ao sistema e configurá-los conforme necessário.
- Integrar dispositivos a diferentes redes ou configurar conexões: O Apoiador Solidário tem a capacidade de integrar dispositivos alimentadores em diferentes redes ou ajustar suas configurações de conexão para garantir a funcionalidade adequada.

Cada caso de uso é diretamente relacionado ao Apoiador Solidário, indicando que ele inicia e controla essas funcionalidades.

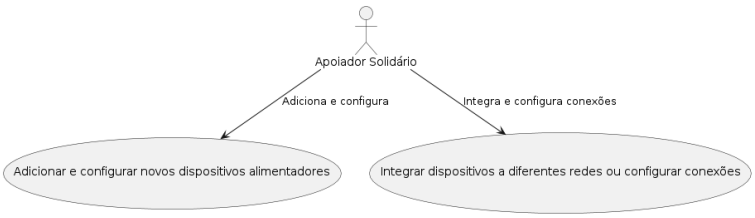

Figura 3.5: Gerenciamento de Dispositivos

3.6.4 Casos de Uso: Interatividade com o Usuário

Atores:

- Apoiador Solidário: Um dos usuários principais que interage com o aplicativo para gerenciar seu perfil e compartilhar dados.

- Veterinário Humanitário: Outro usuário principal que acessa o aplicativo para gerenciar seu perfil, receber e compartilhar informações pertinentes sobre os cuidados dos animais.

Casos de Uso:

- Realizar Login e gestão de perfil de usuário: Ambos, o Apoiador Solidário e o Veterinário Humanitário, podem fazer login no aplicativo e gerenciar suas informações de perfil.

- Compartilhar dados com veterinários ou cuidadores: Eles podem compartilhar informações e dados relevantes sobre os animais com outros profissionais ou cuidadores diretamente através do aplicativo.

- Acessar dicas e cuidados veterinários personalizados: Ambos têm acesso a um banco de dicas e orientações personalizadas sobre o cuidado com os animais, que podem ser consultadas dentro do aplicativo.

Os atores estão conectados a cada um dos casos de uso, refletindo a capacidade de iniciar e interagir com essas funcionalidades.

Figura 3.6: Interatividade com o Usuário

3.6.5 Casos de Uso: Administração do Sistema

Atores:

- Apoiador Solidário: Um usuário que interage com o sistema para gerenciar e monitorar eventos.
- Veterinário Humanitário: Outro usuário que acessa o sistema para acompanhar e responder a eventos alarmantes relacionados aos cuidados com os animais.

Casos de Uso:

- Visualizar logs e eventos considerados alarmantes: Tanto o Apoiador Solidário quanto o Veterinário Humanitário podem acessar uma interface no sistema que lhes permite visualizar registros de atividades e eventos que o sistema identifica como críticos ou alarmantes.

Cada ator está diretamente relacionado ao caso de uso, refletindo a capacidade de ambos visualizar e responder a informações críticas.

Figura 3.7: Administração do Sistema

3.7 Diagrama de Atividade

O diagrama a seguir representa de forma geral dos processos para o alimentador.

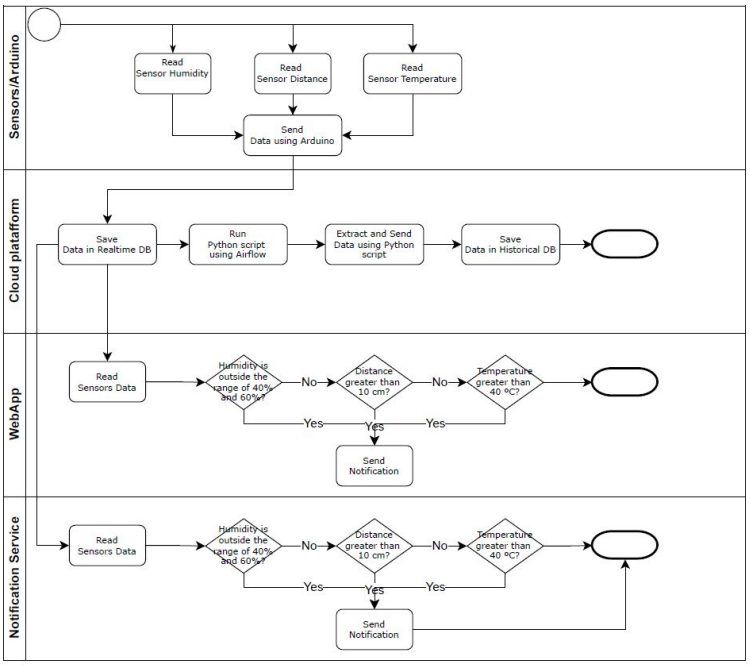

Figura 3.8: Diagrama de Atividade Geral

- Sensores/Arduino: O processo começa com três leituras separadas de sensores ligados a um Arduino:
  - Sensor de humidade: Mede a humidade no ar.
  - Sensor de distância: Mede a proximidade de objectos.
  - Sensor de temperatura: Mede a temperatura ambiente.
- Transmissão de dados: Os dados destes sensores são depois enviados através do Arduino para uma plataforma na nuvem.
- Plataforma de nuvem
  - Guarda os dados dos sensores numa base de dados em tempo real (BD).
  - Executa um script Python utilizando o Airflow para tratar os dados do sensor.

  - Guarda os dados do sensor numa base de dados histórica para armazenamento a longo prazo.

- Aplicação na Nuvem:

  - A aplicação Web lê os dados do sensor a partir da base de dados em tempo real
  - Verifica se a humidade estiver fora do intervalo de 40 - 60.
  - Verifica se a distância do sensor ultrasônico é superior a 10 cm.
  - Verifica se a temperatura for superior a 40°C (Celsius).
  - Se alguma das condições for verdadeira, é enviada uma notificação ao utilizador.

- Serviço de notificação:

  - Este serviço também lê os dados do sensor
  - Efectua as mesmas verificações que a aplicação Web para a humidade, a distância e a temperatura
  - Da mesma forma, se alguma condição for cumprida, envia uma notificação.

3.8 Arquitetura do Sistema

A arquitetura que segue visa demonstrar a relação dos componenres e do tratamento de dados em tempo real e da análise histórica. Na base está a camada de sensores, equipada para a transmissão imediata de dados, que alimenta a base de dados em tempo real para acesso imediato e a base de dados histórica para a manutenção de registos cumulativos. A camada de ligação ao host faz a ponte entre estas bases de dados, assegurando que os dados não só são captados, mas também guardados para referência futura.

Passando da recolha de dados para o envolvimento do utilizador, a Camada de Aplicação oferece uma interface de aplicação Web para monitorização e interação activas, enquanto os Dashboards convertem os dados em análises visuais. O Serviço de Notificação completa este ecossistema, fornecendo alertas atempados aos utilizadores.

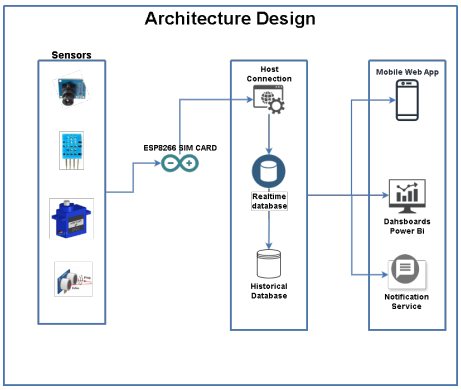

Figura 3.9: Arquitetura do Sistema

Já o diagrama a seguir visa mostrar o fluxo de serviços e como devemos proceder tecnicamente.

Os sensores devem monitorizar diferentes aspectos: Um sensor de presença para detetar quando o alimento esta por terminar, sensores ambientais para medir a temperatura e a humidade, Uma câmara para monitorizar visualmente a área e captar imagens dos gatos. Em seguida os dados dos sensores são geridos da seguinte forma: as leituras de temperatura e humidade são enviadas para uma base de dados, quando o sensor de presença é acionado (indicando que um gato está a comer), este evento é também enviado para a base de dados, a câmara envia imagens para uma aplicação, presumivelmente quando detecta movimento ou presença.

Já o Servidor Web actua como um intermediário que facilita a transferência de dados entre os sensores e as bases de dados, trata os pedidos de API, incluindo POST para receber dados do sensor e GET para recuperar o estado do sensor.

Para armazenar os dados dos sensores devemos considerar dois tipos de base de dados: Base de dados em tempo real (BD) recebe dados actuais dos sensores para utilização imediata ou alertas e a base de dados Histórica armazena dados ao longo do tempo, permitindo a análise de tendências e a manutenção de registos históricos.

Finalmente a Aplicação do utilizador comunica com o servidor Web para enviar e receber dados. Envia dados do sensor em tempo real e recebe actualizações do estado do sensor. A aplicação utiliza estes dados para fornecer uma interface de utilizador para monitorizar e controlar o sistema. O utilizador pode solicitar o estado dos sensores à aplicação, que por sua vez recupera essa informação das bases de dados. O utilizador recebe notificações ou imagens quando o sensor de presença é ativado, sugerindo que um gato está a

comer.

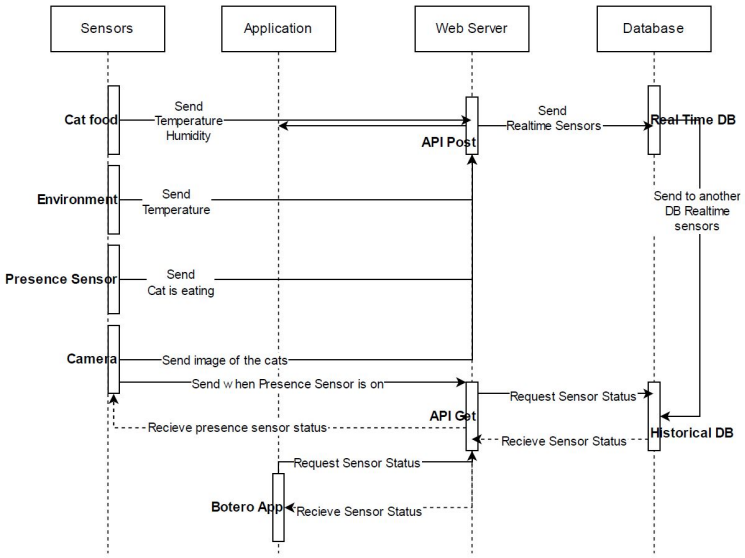

Figura 3.10: Diagrama do fluxo de Serviços Web

3.9 Base de Dados

Nesta solução inicialmente teremos apenas duas tabelas para armazenar todas as informações, dados temporários ou históricos.

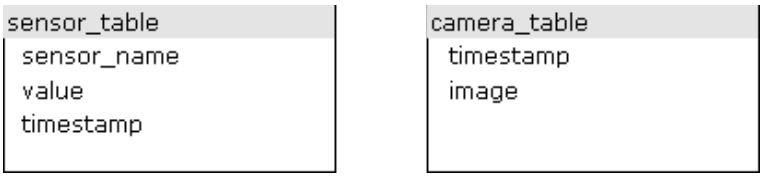

Figura 3.11: Base de Dados Pet Feeder

3.10 Conclusões

O capítulo atual aborda toda a Engenharia de Software usada neste projeto. Apresentamos todos os diagramas da plataforma, bem como os requisitos funcionais e não funcionais. Por fim, a arquitetura do sistema e o esquema da base de dados criado são apresentados.

Capítulo

4

Interoperabilidade

Existem muitos desafios para construir uma arquitetura de sistemas, inclusive quando misturamos em uma mesma solução elementos de hardware e software. Saber como organizar esses elementos é importante para conseguir alcançar o desempenho e a eficiência desejada para o projeto em questão. Além das capacidades operacionais, a arquitetura de um sistema determina a capacidade do sistema de se adaptar a novos requisitos e se integrar com tecnologias atuais e futuras [3]. A interoperabilidade permite que os componentes heterogêneos de vários sistemas e subsistemas trabalhem em conjunto. A comunicação e a troca de dados entre diferentes plataformas tornam possível não apenas uma maior escalabilidade como uma melhora na manutenção da solução.

4.1 Padrões e Protocolos de Comunicação

O protocolo HTTP (Hypertext Transfer Protocol) é amplamente utilizado para comunicações baseadas na web, permitindo a transmissão de dados em sistemas distribuídos de forma fiável e padronizada. Porém, o MQTT (Message Queuing Telemetry Transport) foi especificamente desenhado para ambientes IoT, oferecendo uma comunicação leve e eficiente, ideal para cenários com largura de banda limitada e dispositivos com baixo poder de processamento [4].

Estes protocolos são complementares; enquanto o HTTP facilita interações complexas e a entrega de conteúdo dinâmico, o MQTT destaca-se ao enviar mensagens entre dispositivos de forma rápida e com uso mínimo de recursos. Juntos, permitem a criação de redes de dispositivos IoT altamente eficazes e interoperáveis, assegurando que diferentes tecnologias possam co-

municar e operar conjuntamente de forma eficaz, adaptando-se às necessidades do sistema e às evoluções tecnológicas [5].

4.2 Interoperabilidade de Dispositivos

Um dos maiores obstáculos na implementação de sistemas complexos de Internet das Coisas é a interoperabilidade. Estamos enfrentando uma tarefa tanto técnica quanto burocrática ao permitir que dispositivos e aplicativos de diferentes fabricantes comuniquem eficientemente.

Quando o assunto é IoT, a interoperabilidade de dispositivos é um tópico importante, dado que se trata de uma questão fundamental na concepção da solução tecnológica. Existem diferentes fabricantes de plataformas tecnológicas que podem ou não comunicar entre si, e outras que podem ter alguma forma de cooperação visando a sua interoperabilidade de funcionalidades. A adoção de padrões abertos é uma forma para assegurar esta interoperabilidade em conjunto com o uso de interfaces de programação de aplicações (APIs), conforme destacado por [6]. Segundo esse trabalho a padronização é essencial para facilitar a integração entre dispositivos heterogêneos, permitindo que estes trabalhem de uma forma conjunta e sem gerar atrito na comunicação. Essa estratégia não só melhora a escalabilidade e a gestão dos sistemas IoT, mas também otimiza a eficiência operacional, permitindo que os sistemas se ajustem e evoluam com as mudanças tecnológicas.

Várias iniciativas abordam a questão da falta de padrões "universais". Além disso, os padrões estão rapidamente desatualizados à medida que novas tecnologias entram em vigência, tornando-se um desafio extra para a integração de sistemas legados e os atuais. Os informáticos devem estar constantemente atualizados para se manterem relevante neste ambiente em constante mudança. Abaixo listamos tecnologias com o objetivo de resolver esse problema de integração:

- **MQTT:** Protocolo leve para a comunicação em sistemas onde a largura de banda é limitada. ([7])
- **CoAP:** Protocolo web para dispositivos restritos, ideal para pequenas redes IoT. ([8])
- **HTTP/HTTPS:** Protocolo de comunicação universalmente utilizado, não específico para IoT, mas essencial para interoperabilidade web. ([9])
- **AMQP:** Protocolo orientado a mensagens que suporta interoperabilidade crítica entre sistemas. ([10])

- **DDS:** Padrão para comunicações em tempo real em sistemas distribuídos. ([11])
- **OneM2M:** Padrão global para IoT que visa unificar os dispositivos e serviços sob uma arquitetura comum. ([12])
- **Zigbee:** Protocolo de automação residencial e industrial para comunicação sem fio de baixa potência. ([13])
- **Z-Wave:** Protocolo semelhante ao Zigbee, usado em automação residencial para controle de dispositivos. ([14])
- **Bluetooth Low Energy (BLE):** Versão de baixo consumo de energia do Bluetooth, utilizada para comunicação de curto alcance em dispositivos IoT. ([15])
- **Thread:** Protocolo baseado em IP para automação residencial, otimizado para dispositivos IoT de baixa potência. ([16])

Capítulo

5

Conclusões e Trabalho Futuro

5.1 Conclusões Principais

Esta secção contém a resposta à questão:
Quais foram as conclusões princípais a que o(a) aluno(a) chegou no fim deste trabalho?

5.2 Trabalho Futuro

Esta secção responde a questões como:
O que é que ficou por fazer, e porque?
O que é que seria interessante fazer, mas não foi feito por não ser exatamente o objetivo deste trabalho?
Em que outros casos ou situações ou cenários – que não foram estudados no contexto deste projeto por não ser seu objetivo – é que o trabalho aqui descrito pode ter aplicações interessantes e porque?

Bibliografia

[1] Amel Bennaceur, Thein Tun, Yijun Yu, and Bashar Nuseibeh. *Requirements Engineering*. 01 2018.

[2] Scott W. Ambler. *The elements of UML 2.0 style*. Cambridge University Press, 2010.

[3] Len Bass, Rick Kazman, and Paul Clements. *Software architecture in practice*. Addison-Wesley, 2013.

[4] Roy T. Fielding and Julian F. Reschke. Hypertext transfer protocol (http/1.1): Semantics and content. RFC 7231, 1999.

[5] Andrew Banks and Rahul Gupta. Mqtt version 3.1.1. Oasis standard, OASIS, 2014.

[6] Ammar Rayes and Samer Salam. *Internet of Things From Hype to Reality: The Road to Digitization*. Springer, Cham, Switzerland, 2019.

[7] Mqtt version 3.1.1. `https://mqtt.org/`.

[8] Constrained application protocol (coap). `https://coap.technology/`.

[9] Hypertext transfer protocol – http/1.1. `https://www.ietf.org/rfc/rfc2616.txt`.

[10] Advanced message queuing protocol (amqp). `https://www.amqp.org/`.

[11] Data distribution service (dds) standard. `https://www.omg.org/spec/DDS/`.

[12] onem2m - standards for m2m and the internet of things. `https://www.onem2m.org/`.

[13] Zigbee alliance. `https://www.zigbee.org/`.

[14] Z-wave. `https://www.z-wave.com/`.

[15] Bluetooth technology website. `https://www.bluetooth.com/`.

[16] Thread group. `https://www.threadgroup.org/`.