

**CENTRO ESTADUAL DE EDUCAÇÃO TECNOLÓGICA**

**PAULA SOUZA**

**FACULDADE DE TECNOLOGIA DE LINS PROF. ANTONIO SEABRA**

**CURSO SUPERIOR DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE  
SISTEMAS**

**JOÃO PEDRO CHÉR BENETIS DOS SANTOS**

**JOÃO VICTOR CANDIOTTI ALVES**

**PROTÓTIPO DE APLICAÇÃO WEB PARA CONTROLE GERENCIAL  
DE UM MECANISMO DE SEGURANÇA DE ACESSO BASEADO EM  
UMA CANCELÂNCIA ELETRÔNICA COM ARDUINO**

**LINS/SP**

**2º SEMESTRE/2016**

**CENTRO ESTADUAL DE EDUCAÇÃO TECNOLÓGICA**

**PAULA SOUZA**

**FACULDADE DE TECNOLOGIA DE LINS PROF. ANTONIO SEABRA**

**CURSO SUPERIOR DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE  
SISTEMAS**

**JOÃO PEDRO CHÉR BENETIS DOS SANTOS**

**JOÃO VICTOR CANDIOTTI ALVES**

**PROTÓTIPO DE APLICAÇÃO WEB PARA CONTROLE GERENCIAL  
DE UM MECANISMO DE SEGURANÇA DE ACESSO BASEADO EM  
UMA CANCELHA ELETRÔNICA COM ARDUINO**

Trabalho de Conclusão de Curso apresentado à  
Faculdade de Tecnologia de Lins para obtenção do  
Título de Tecnólogo em Análise e Desenvolvimento  
de Sistemas.

Orientador: Prof. Mestre Alexandre Ponce de

**LINS/SP  
2º SEMESTRE/2016**

S237p Santos, João Pedro Chér Benetis dos  
Protótipo de aplicação web para controle gerencial de um  
mecanismo de segurança de acesso baseado em uma cancela  
eletrônica com Arduino / João Pedro Chér Benetis dos Santos e João  
Vitor Candiotti Alves. – Lins, 2016.  
95 f. : il.

Monografia (Trabalho de Conclusão de Curso de Tecnologia em  
Análise e Desenvolvimento de Sistemas) – Faculdade de Tecnologia de  
Lins Professor Antônio Seabra, 2016.

Orientador: Prof. Me. Alexandre Ponce de Oliveira

1.Protótipo. 2.Cancela. 3.Automação. 4.Arduino. 5.Sistema web.  
I.Alves, João Vitor Candiotti. II.Oliveira, Alexandre Ponce de.  
III.Faculdade de Tecnologia de Lins Prof. Antônio Seabra. IV.Título.

CDD 658.4038

Ficha elaboradora pela Biblioteca da Faculdade de Tecnologia de Lins

**JOÃO PEDRO CHÉR BENETIS DOS SANTOS**

**JOÃO VICTOR CANDIOTTI ALVES**

**PROTÓTIPO DE APLICAÇÃO WEB PARA CONTROLE GERENCIAL  
DE UM MECANISMO DE SEGURANÇA DE ACESSO BASEADO EM  
UMA CANCELÂ ELETRÔNICA COM ARDUINO**

Trabalho de Conclusão de Curso apresentado à Faculdade de Tecnologia de Lins, como parte dos requisitos necessários para obtenção de do título de Tecnólogo em Análise e Desenvolvimento de Sistemas sob orientação do Prof. Me. Alexandre Ponce de Oliveira.

---

Orientador Prof. Me. Alexandre Ponce de Oliveira

---

Prof. Me. Júlio Fernando Lieira

---

Prof. Dr. Mário Herique de Souza Pardo

Dedico este trabalho à toda minha família que me deram apoio, principalmente à minha mãe Rosa e minha vó Vanda por toda a ajuda que me ofereceram proporcionando a conclusão deste curso.

**João Pedro Chér Benetis dos Santos**

Dedico este trabalho primeiramente a Deus que em todos os momentos de dificuldades me exaltou diante aos meus objetivos, agradeço a minha Mãe Ivanilde e Vó Ivanirde por todo o apoio e amparo em meio a esta caminhada e a minha namorada Suellen por todo o sustento durante esta árdua etapa da minha vida.

**João Victor Candiotti Alves**

## **AGRADECIMENTOS**

Agradeço a todos os com quem tive contato nesse curso e nessa faculdade.

Aos professores que nos auxiliaram para a conclusão do curso.

Gostaria de agradecer também aos meus amigos que ficaram até o final desta jornada, tornando esse aprendizado uma experiência para toda a vida e que ajudaram para o desenvolvimento deste trabalho.

Sou grato ao meu orientador Me. Alexandre Ponce de Oliveira pela sua ajuda e paciência, além de proporcionar toda a estrutura para a construção deste projeto.

E sobre tudo, dedico este trabalho à minha família, principalmente minha mãe Rosa e minha vó Vanda, que me deram suporte e pela confiança que depositaram em mim, fornecendo toda uma estrutura para um maior aprendizado para minha vida profissional.

**João Pedro Chér Benetis dos Santos**

## **AGRADECIMENTOS**

Agradeço primeiramente a Deus por me orientar em árduos momentos, por me conceder capacidade e sabedoria para a realização deste trabalho e me motivar todos os dias a seguir em frente com meus objetivos delineados e almejados.

Em especial dedico este trabalho a minha família, Vó Ivanirde Terezinha, Mãe Ivanilde Candiotti e namorada Suellen Montanha que tanto me deram suporte e ajudaram a concluir o trabalho e a faculdade de forma especial e satisfatória, com toda certeza estas pessoas foram cruciais em todas os ciclos da minha vida incluindo este.

Ao meu orientador Me. Alexandre Ponce de Oliveira que me auxiliou e amparou em momentos complexos, tornando tudo mais claro e tangível.

Aos professores que contribuíram com a nossa formação nos tornando um profissional de qualidade, deixo meu grande abraço para todos pois sem eles nenhum conhecimento e experiência existiriam e o propósito do curso se tornaria inválido.

A todos meus amigos que em sala e fora me ajudaram com trabalhos, tarefas e atividades envolvidas durante todo o trajeto de ensino.

Agradeço a todos que de forma direta ou indireta propiciaram para a realização do presente trabalho e me ajudaram a ser um profissional muito melhor.

**João Victor Candiotti Alves**

## **RESUMO**

A tecnologia da informação tem evoluído consideravelmente, hardware mais eficaz, software mais inteligente, rede mais veloz, tecnologias de ponta proporcionam eficiência nos mais diversos aspectos. Alinhado com as necessidades das organizações que necessitam de uma eficaz gerência e controle dentro da sua companhia, o presente trabalho, com o intuito de auxiliar na organização e no gerenciamento de acesso das pessoas e seus veículos, possui o escopo partindo da premissa de solucionar problemas técnicos e ambientais. O software desenvolvido, toda a sua estrutura física e a placa controlada Arduino tem como principal função propor soluções como acesso controlado a determinados ambientes, os registros desses acessos, a identificação automática e a aprovação ou não a ambientes previamente definidos pelos administradores, tudo isso estabelecido por meio de sinais de rádio através de etiquetas RFID, além de efetivar o armazenamento de dados, persistindo as operações no servidor da instituição e localmente na placa controladora que possui um cartão SD com o plano de armazenamento em caso de possíveis falhas por variáveis externas. O sistema e o controle dos dispositivos serão utilizados pelos usuários da instituição, devidamente gerenciados pelo departamento de T.I, os funcionários se apossaram de uma *tag* RFID previamente configurada para verificação de logs, validação de *tag* e disponibilização de funcionamento a diversos dispositivos, ou seja, ambientes da instituição, dispositivos travam e destravam o acesso a ambientes, os mesmos serão projetados especificamente em portas, cancelas eletrônicas dentre outros objetos que após definidos e adicionados mecanismos físicos e o leitor da *TAG* o mesmo pode ser gerenciado via Internet.

Palavras-chave: Protótipo, cancela, automação, arduino, sistema web.

## **ABSTRACT**

Information technology has evolved considerably, more efficient hardware, smarter software, faster network, cutting-edge technologies provide efficiency in the most diverse aspects. Aligned with the needs of organizations that need effective management and control within their company, the present work, with the purpose of assisting in the organization and management of access of people and their vehicles, has the scope starting from the premise of solving problems Technical and environmental aspects. The software developed, all its physical structure and the Arduino controlled board have as main function to propose solutions like controlled access to certain environments, the registries of these accesses, the automatic identification and the approval or not to environments previously defined by the administrators, all this established By means of radio signals through RFID tags, in addition to effecting data storage, the operations in the institution's server and locally in the controller card that has an SD card with the storage plan in case of possible external variables failures. The system and control of the devices will be used by the users of the institution, duly managed by the IT department, the employees take possession of an RFID tag previously configured for verification of logs, validation of tag and availability of operation to several devices, Environments of the institution, devices lock and unlock access to environments, they will be designed specifically on doors, electronic gates among other objects that after defined and added physical mechanisms and the TAG reader can be managed via the Internet.

**Keywords:** Prototype, cancela, automation, arduino, web system.

## LISTA DE ILUSTRAÇÕES

Figura 1.1 - Esboço dos componentes da placa Arduino .....	19
Figura 1.2 – Placa Arduino UNO.....	21
Figura 1.3 - Resumo placa Arduino UNO.....	21
Figura 1.4 – Pinagem placa Arduino UNO (ATmega328) .....	22
Figura 1.5 – Resumo placa Arduino MEGA 2560.....	23
Figura 1.6 – Pinagem placa Arduino MEGA 2560.....	23
Figura 1.7 – Placa Arduino LEONARDO .....	24
Figura 1.8 – Especificação placa Arduino LEONARDO .....	25
Figura 1.9 – Placa Arduino NANO.....	27
Figura 1.10 – Pinagem placa Arduino NANO.....	27
Figura 1.11 – Placa Raspberry Pi.....	28
Figura 2.1 – Resumo Protoboard .....	31
Figura 2.2 – Arduino Ethernet Shield. ....	32
Figura 2.3 – Tela TFT LCD (Front & Back).....	33
Figura 2.4 – Sensor & Etiquetas RFID. ....	34
Figura 2.5 – Exemplos Resistor. ....	35
Figura 2.6 – Exemplos Cabo UTP com Conector RJ45 .....	36
Figura 2.7 – Exemplo de cabo UTP CAT 5e. ....	36
Figura 2.8 - Estrutura Servo motor. ....	37
Figura 2.9 - Pulso Variável de controle posição servo motor. ....	38
Figura 2.10 – Sensor Ultrassônico .....	38
Figura 3.1 - Diagrama de Casos de Uso do Sistema .....	45
Figura 3.2 - Diagrama de Entidade e Relacionamento.....	52
Figura 3.3 - Diagrama de Classes.....	53
Figura 3.4 - Diagrama de MVC Gerenciar Usuário.....	55
Figura 3.5 - Diagrama de MVC Log de Registro .....	56
Figura 3.6 - Diagrama de MVC Login.....	56
Figura 3.7 - Diagrama de MVC Gerenciar Automóvel .....	57
Figura 3.8 - Diagrama de MVC Gerenciar Departamento .....	57
Figura 3.9 - Diagrama de MVC Gerenciar Dispositivo.....	58
Figura 3.10 - Diagrama de MVC Gerenciar Função .....	58
Figura 3.11 - Diagrama de MVC Gerenciar Local .....	59
Figura 3.12 - Diagrama de MVC Gerenciar Tag RFID .....	59
Figura 3.13 - Diagrama de Atividades: Login .....	60
Figura 3.14 - Diagrama de Atividades: Gerenciar Usuário .....	61
Figura 3.15 - Diagrama de Atividades: Gerenciar Automóvel.....	62
Figura 3.16 - Diagrama de Atividades: Gerenciar Etiqueta RFID .....	63
Figura 3.17 - Diagrama de Atividades: Gerenciar Dispositivos .....	64
Figura 3.18 - Diagrama de Atividades: Gerenciar Local.....	65
Figura 3.19 - Diagrama de Atividades: Gerenciar Departamento.....	66
Figura 3.20 - Diagrama de Atividades: Gerenciar Função .....	67
Figura 3.21 - Diagrama de Atividades: Log de Registro Usuário Administrador .....	68
Figura 3.22 - Diagrama de Atividades: Usuário Comum .....	69

Figura 3.23 - Diagrama de Sequência: Gerenciar Usuário.....	70
Figura 3.24 - Diagrama de Sequência: Gerenciar Logs de Registro .....	71
Figura 3.25 - Diagrama de Sequência: Fazer Login.....	71
Figura 3.26 - Diagrama de Sequência: Gerenciar RFID.....	72
Figura 3.27 - Diagrama de Sequência: Gerenciar Dispositivos .....	73
Figura 3.28 - Diagrama de Sequência: Gerenciar Local .....	73
Figura 3.29 - Diagrama de Sequência: Gerenciar Departamento .....	74
Figura 3.32 - Diagrama de Componentes: Gerenciar Usuário .....	76
Figura 3.33 - Diagrama de Componentes: Gerenciar Log de Registro .....	77
Figura 3.34 - Diagrama de Componentes: Gerenciar Automóvel.....	77
Figura 3.35 - Diagrama de Componentes: Gerenciar Etiqueta RFID .....	78
Figura 3.36 - Diagrama de Componentes: Gerenciar Dispositivos.....	78
Figura 3.37 - Diagrama de Componentes: Gerenciar Local .....	79
Figura 3.38 - Diagrama de Componentes: Gerenciar Departamento .....	79
Figura 3.39 - Diagrama de Componentes: Gerenciar Log de Registro .....	80
Figura 3.40 - Diagrama de Componentes: Efetuar Login .....	80
Figura 4.1 – Exemplo de cancela eletrônica com sensor .....	81
Figura 4.2 – Infraestrutura de comunicação entre dispositivo e o sistema.....	82
Figura 4.3 – Protótipo usado no trabalho .....	83
Figura 4.4 – Código do Arduino para comunicação e tratamento dos dados 1 .....	84
Figura 4.5 – Código do Arduino para comunicação e tratamento dos dados 2 .....	85
Figura 4.6 - Interface de Login .....	86
Figura 4.7 - Interface de Log de Registro .....	86
Figura 4.8 - Interface de Gerenciamento de Usuário .....	87
Figura 4.9 - Interface de Gerenciamento de Usuário com caixa de Informações.....	87
Figura 4.10 - Interface de Cadastro de Usuário .....	88
Figura 4.11 - Interface de Cadastro de Automóveis .....	88
Figura 4.12 - Interface de Informações de Cadastro de Dispositivo .....	89
Figura 4.13 - Interface para verificar a tag e liberar o acesso ao dispositivo.....	89
Figura 4.14 - Interface de Informações da Conta de Usuário.....	90

## **LISTA DE ABREVIATURAS E SIGLAS**

API – *Application Programming Interfaces*

AREF - *AnalogReference*

ARM - *Advanced RISC Machine*

CDC – *Connected Device Class*

CMOS - *Complementary metal-oxide semiconductor*

DAO – *Data Access Object*

DC - *Direct Current*

DENATRAN – Departamento Nacional de Trânsito

EEPROM - *Electrically Programmable Read-Only Memory*

HDMI - *High-Definition Multimedia Interface*

IDE - *Integrated Development Environments*

ICSP – *In-Circuit Serial Programming*

IP – *Internet Protocol*

LCD – *Liquid Crystal Display*

LED – *Light Emitting Diode* (Diodo Emissor de Luz)

MER – Modelo Entidade Relacionamento

MISO – *Master In Slave Out*

MOSI – *Master Out Slave In*

MVC – *Model View Controller*

PoE - *Power over Ethernet*

PWM – *Pulse-Width Modulation*

RISC - *Reduced Instruction Set Computer*

RFID – *Radio-Frequency Identification* (Identificação por Radiofrequencia)

SCK – *Serial Clock*

SCL – *Serial Clock Pin*

SD – *Secure Digital Card*

SDA – *Serial Data Line*

SMD – *Surface Mounted Device* (Dispositivo de Montagem Superficial)

SPI – *Serial Peripheral Interface*

SQL - *Structured Query Language*

SRAM - *Static Random Access Memory*

SS – *Slave Select*

TTL - *Transistor-Transistor Logic*

UART - *Universal Asynchronous Receiver/Transmitter*

UDP – *User Datagram Protocol*

UID - *Unique Identification*

UML - *Unified Modeling Language*

USB – *Universal Serial Bus*

UTP – *Unshielded Twisted Par* (Cabo par Trançado)

TCP – *Transmission Control Protocol*

## SUMÁRIO

INTRODUÇÃO .....	16
1 DETALHES, FUNÇÕES E APLICAÇÕES DO ARDUINO .....	18
1.1 PROJETO ARDUINO.....	18
1.2 ARDUINO UNO.....	20
1.3 ARDUINO MEGA 2560 .....	22
1.4 ARDUINO LEONARDO.....	24
1.5 ARDUINO NANO .....	26
1.6 RASPBERRY PI.....	28
1.7 PRETEXTOS PARA UTILIZAR ARDUINO .....	29
2 AUTOMAÇÃO DA CANCELHA E COMPONENTES UTILIZADOS ...	30
2.1 PROTOBOARD.....	30
2.1.1 Faixa de terminais .....	31
2.1.2 Faixas de barramentos .....	31
2.2 ARDUINO ETHERNET SHIELD .....	31
2.3 ARDUINO TFT LCD .....	33
2.4 SENSOR MFRC522 E TAG RFID.....	33
2.4.1 Tipos de etiquetas RFID: .....	34
2.4.1.1 Passiva .....	34
2.4.1.2 Ativa .....	34
2.5 RESISTOR.....	35
2.6 CABO DE REDE .....	36
2.7 SERVO MOTOR .....	37
2.8 SENSOR ULTRASSÔNICO.....	38
3 ESPECIFICAÇÃO, ANÁLISE E ESTRUTURA DO PROJETO .....	39
3.1 Análise e projeto do sistema .....	39
3.2 Análise de negócio .....	39
3.3 Descrição do problema .....	39
3.4 Atores envolvidos .....	40
3.5 Perspectiva do Produto .....	40
3.6 Características .....	41
3.6.1 Funcionalidades do Usuário Administrador: .....	41
3.7 Análise de Requisitos.....	42

3.8	Análise de requisitos Funcionais .....	42
3.9	Análise de Requisitos Não Funcionais .....	42
3.10	Diagrama de caso de uso .....	44
3.10.1	Especificação de Caso de Uso .....	46
3.10.1.1	Caso de Uso: Fazer Login.....	46
3.10.1.2	Caso de Uso: Verificar Log de Registro .....	46
3.10.1.3	Caso de Uso: Gerenciar Usuários.....	47
3.10.1.4	Caso de Uso: Gerenciar Dispositivo .....	47
3.10.1.5	Caso de Uso: Gerenciar Local .....	48
3.10.1.6	Caso de Uso: Gerenciar Função .....	49
3.10.1.7	Caso de Uso: Gerenciar Departamento .....	49
3.10.1.8	Caso de Uso: Gerenciar Automóvel.....	50
3.10.1.9	Caso de Uso: Gerenciar Tag RFID .....	50
3.10.1.10	Caso de Uso: Consultar Log de Registro .....	51
3.11	Análise e Design .....	51
3.12	Modelagem de Banco de dados.....	51
3.13	Diagrama de Classes .....	53
3.14	MODELO VISÃO E CONTROLE E OBJETOS DE ACESSO A DADOS.....	54
3.15	Diagramas de Atividades .....	60
3.15.1	Efetuar Login .....	60
3.15.2	Gerenciar Usuário.....	61
3.15.3	Gerenciar Automóvel .....	62
3.15.4	Gerenciar RFID.....	63
3.15.5	Gerenciar Dispositivos.....	64
3.15.6	Gerenciar Local .....	65
3.15.7	Gerenciar Departamento .....	66
3.15.8	Gerenciar Função .....	67
3.15.9	Exibir Log de Registro .....	68
3.15.10	Visualizar Log de Registros Usuário Comum .....	69
3.16	Diagramas de Sequencia .....	69
3.16.1	Gerenciar Usuário.....	70
3.16.2	Gerenciar Log de Registro .....	71
3.16.3	Fazer Login.....	71
3.16.4	Gerenciar Tag RFID .....	72
3.16.5	Gerenciar Dispositivos .....	72

3.16.6	Gerenciar Local .....	73
3.16.7	Gerenciar Departamento .....	74
3.16.8	Gerenciar Função .....	74
3.16.9	Gerenciar Automóvel .....	75
3.17	Diagramas de componentes .....	76
3.17.1	Gerenciar Usuário.....	76
3.17.2	Gerenciar Log de Registro.....	77
3.17.3	Gerenciar Automóvel .....	77
3.17.4	Gerenciar Etiqueta RFID .....	77
3.17.5	Gerenciar Dispositivos.....	78
3.17.6	Gerenciar Local .....	78
3.17.7	Gerenciar Departamento .....	79
3.17.8	Gerenciar Função .....	79
3.17.9	Fazer Login.....	80
4	IMPLEMENTAÇÃO.....	81
	CONCLUSÃO.....	91
	REFERÊNCIAS BIBLIOGRÁFICAS .....	93

## INTRODUÇÃO

A tecnologia está presente em diversos lugares e áreas e a cada momento surgem inovações que facilitam em diversas áreas da sociedade as tarefas desempenhadas por pessoas. Na computação, a Tecnologia da Informação cresceu de forma com que as pessoas podem criar dispositivos e sistemas cada vez mais uteis a sociedade. Porém, segundo SILVA (2008), toda inovação tem seu ponto positivo e negativo em relação à tecnologia da informação, os benefícios são inúmeros, mas os desafios também. É necessário ter um controle sobre as informações, porém não é sempre possível controlar. (SILVA, 2008)

A tecnologia é também usual na área de segurança e está cada vez mais desenvolvida e requisitada. Diversos dispositivos como cancelas, travas eletrônicas, sensores, etc., podem ser gerenciados por meio de sistemas automatizados de segurança que mantém o controle de informações e acesso a ambientes.

O uso da tecnologia da informação pode gerar problemas, bem como falhas na aplicação de segurança e interferências externas, por exemplo, erros humanos. Uma parte desta problemática está relacionada à organização das pessoas e de seus meios de transportes (automóveis, por exemplo).

De acordo com o DENATRAN (2016), a número de veículos existentes no Brasil até janeiro de 2016 é de 91.191.372 milhões, considerando automóveis, ônibus, motocicletas, caminhões, utilitários e outros, são aproximadamente 50.045.737 somente de automóveis que representam 54,87% de toda a frota de veículos rodando pelo país.

De acordo com essas informações, é de conhecimento que acontecem erros relacionados à segurança e organização do trânsito, onde em alguns casos não se tem controle de acesso ou registro de entradas e saídas de quaisquer veículos, podendo ocorrer congestionamento ou mesmo mortes. (PERKONS, 2007)

Com o intuito de auxiliar na organização e no gerenciamento de acesso das pessoas e suas conduções são utilizados alguns métodos, como por exemplo, a cancela eletrônica. A cancela eletrônica atua sem a interferência constante do ser humano, é a capaz de restringir ou liberar o acesso de veículos a certos locais, como estacionamento, rodovias, entre outros locais.

Este trabalho teve como objetivo desenvolver um sistema web e um protótipo de automação de uma cancela utilizando Arduino que permitem administrar o uso de

uma cancela eletrônica para controlar o acesso de veículos a um determinado estacionamento. Os logs de acesso ao uso da cancela serão armazenados em um banco de dados, ou seja, quem passou pela cancela, quando passou e que horas.

Para atender ao objetivo, foi desenvolvido um protótipo de automação que simula todo o sistema de cancela eletrônica, posto que a mesma faz uso da placa Arduino e de *tags* RFID (*Radio-Frequency Identification*). O protótipo possibilita a identificação automática por meio de sinais de rádio e admitir a recuperação e o armazenamento de dados por meio de dispositivos denominados *tag* RFID.

O trabalho está estruturado da seguinte forma, o primeiro capítulo aborda de forma mais detalhada o uso do Arduino, com demonstração de funções e conceitos que são necessários. O segundo capítulo aborda conceitos relacionados a automação da cancela e seus componentes.

A especificação, análise e projeto do sistema serão abordados no terceiro capítulo, no qual, incluem-se os diagramas de casos de uso, de atividades, de sequência e o MER. A implementação do sistema, interface gráfica, protótipo da automação do sistema de cancela serão apresentadas no quarto capítulo. Por fim, têm-se as conclusões finais e referências utilizadas no desenvolvimento do trabalho.

# 1 DETALHES, FUNÇÕES E APLICAÇÕES DO ARDUINO

Neste capítulo é abordada a plataforma de automação Arduino, a estrutura desta placa micro controlado, seu conceito, suas funcionalidades e sua atuação na área de tecnologia que permite aos seus usuários integra-la em quase todos os tipos de sistemas, adaptando e otimizando sistemas, possibilitando isso a um baixo custo.

## 1.1 PROJETO ARDUINO

Em suma, o Arduino é definido como uma plataforma física de prototipagem, projetado com uma placa micro controladora Atmel AVR, com código aberto baseado em *easy-to-use* incluído *hardware* e *software* livre, objetivando na criação de diversos tipos de projetos interativos e de baixo custo e com ferramentas adaptáveis. Placas Arduino são capazes de ler entradas e transformá-las em uma saída. (ARDUINO, 2016)

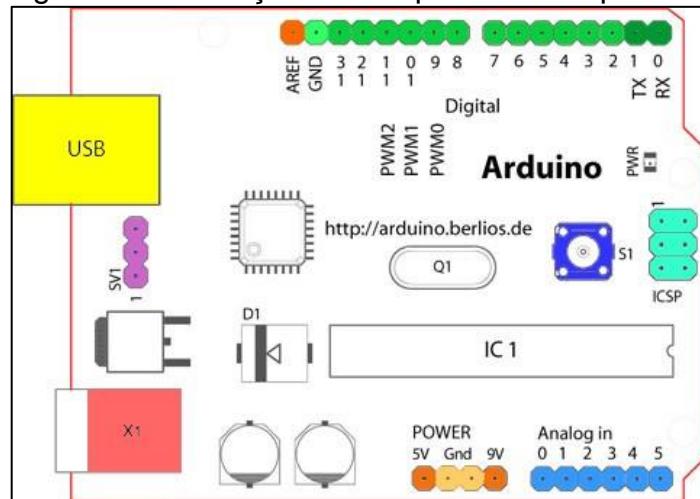
Segundo KUSHNER (2011), o projeto surgiu em Ivrea, na Itália, em 2005, criado por um grupo de estudantes de Design no *Interaction Design Institute Ivrea* (IDII), entre eles o co-fundador Massimo Banzi, e que visando obter uma alternativa ao *hardware* do curso que era caro e complexo para os estudantes iniciaram o projeto.

De acordo com SOUZA (2013), as primeiras placas de Arduino começaram com comunicação serial e poucos componentes, essas placas usavam o padrão RS232 para interface com computador, mas com o grande uso de *Universal Serial Bus* (USB) foi criada a placa Arduino USB, a primeira placa com o nome de Arduino e alimentação de energia diretamente do cabo de comunicação. Posteriormente foi lançada a placa Arduino USB v2.0 que trazia como conversor USB serial o chip FT232BM. Em seguida saíram as placas Arduino Extreme que originou a maioria dos componentes em montagem *Surface Mounted Devices* (SMD), além de trazer os *headers* fêmea, que é um tipo de conector elétrico, popularizando como o “padrão Arduino”. Após toda essa evolução da plataforma, foi lançado a placa Arduino *Nuova Generazione* (NG), trazendo um conversor USB-SERIAL FT232RL que necessita de menos energia de componentes externos em comparação ao FT232BM. Depois do projeto da placa NG, as placas posteriores passaram a utilizar o ATmega168 como micro controlador que dobrava a capacidade de memória para 16KB, substituindo o

antigo micro controlador ATmega8. Em seguida, uma nova versão foi criada Arduino NG REV. C. resolvendo alguns problemas, como a comunicação *Serial Peripheral Interface* (SPI) que causa interferência devido ao *Light Emitting Diode* (LED) ficar no pino 13, além de possuir um resistor de 1k em série. Porém todas essas placas anteriores precisavam de um reset para ativar o *bootloader*, sendo este um pequeno programa que é executado quando você liga o Arduino ou pressiona o botão reset e sua principal função é carregar um programa do seu computador para o Arduino, isso dificultava para fazer o upload do sketch para placa e a fim de resolver este problema foi lançada a placa Arduino Diecimila que tinha um circuito de reset através da comunicação serial, desse modo a placa entrava em modo *bootloader* automaticamente. Em março de 2009 começou a ser fabricada a Arduino Duemilanove com o micro controlador ATmega328 dobrando a memória de 16KB para 32KB. Após o desenvolvimento do Duemilanove, foi lançada a placa Arduino UNO que passou por três revisões e substituiu o conversor USB-SERIAL por um micro controlador ATmega16U2, melhorou a identificação dos pinos.

Na Figura 1.1 é demonstrado um protótipo dos componentes que fazem parte de uma placa Arduino.

Figura 1.1 - Esboço dos componentes da placa Arduino.



Fonte: Arduino, 2016.

- Pino terra digital *ground* (GND) (Verde claro);
- Pino de IN analógica 0 – 5 (Azul claro);
- Pino de referência analógico (Cor laranja);
- Pinos digitais 2 – 13 (Verde);

- Pinos digitais 0 – 1, serial IN/OUT – RX/TX 9 Verde escuro), são pinos usados para entrada e saída digitais (*digitalRead* e *digitalWrite*);
- IN circuito programador de série – ICSP (Azul esverdeado);
- Power – pinos (VIN) e pinos terra, com energia de 5v e 9v (Laranja e pinos terra em Laranja claro);
- Botão Reset – S1 (Azul claro);
- Entrada da fonte de alimentação IN (9 – 12 VDC) – X1 (Rosa);
- USB responsável pela transferência de dados para placa e pela comunicação serial entre a placa e o computador, podendo ser usado na alimentação de energia para a placa;

De acordo com SOUZA, a programação do Arduino é feita através de uma *Integrated Development Environments* (IDE) que possibilita a criação de *sketches* para a placa usando a linguagem *Wiring*, sendo esta linguagem um *framework open-source* que faz a tradução do código escrito para a linguagem C/C++ e depois é enviado para o compilador avr-gcc que trata da transcrição dos comandos para que seja compreendida pelo micro controlador. (SOUZA, 2013)

Por meio de programas que qualquer um pode desenvolver é possível dizer a placa o que fazer através do envio de um conjunto de instruções para o micro controlador da placa. Esses programas desenvolvidos para o Arduino podem ser divididos em três partes: *Structure* (estruturas), *Values* (valores) e *Functions* (funções). (ARDUINO, 2016)

Como existem vários tipos de Arduino no mercado, alguns são mais usados, como por exemplo o Arduino UNO, Arduino Mega, Arduino Leonardo e Arduino nano.

## 1.2 ARDUINO UNO

Atualmente em sua terceira geração é a mais popular entre os usuários desse hardware. Ele possui um micro controlador ATmega328P, 14 pinos I/O (das quais 6 fornecem saída *Pulse-Width Modulation* - PWM), um cristal de quartzo 16Mhz, uma conexão USB, botão reset, um *In-Circuit Serial Programming* (ICSP) header, memória flash de 32KB, velocidade de 16MHz e pesando de 25g. O Arduino UNO vem pré-programado com um *bootloader* que possibilita o envio de novos códigos

sem uso de um hardware externo e sua comunicação é baseada no protocolo original STK500. O Arduino UNO possui 3 revisões. (ARDUINO, 2016)

Na figura 1.2 é demonstrado uma placa Arduino Uno.

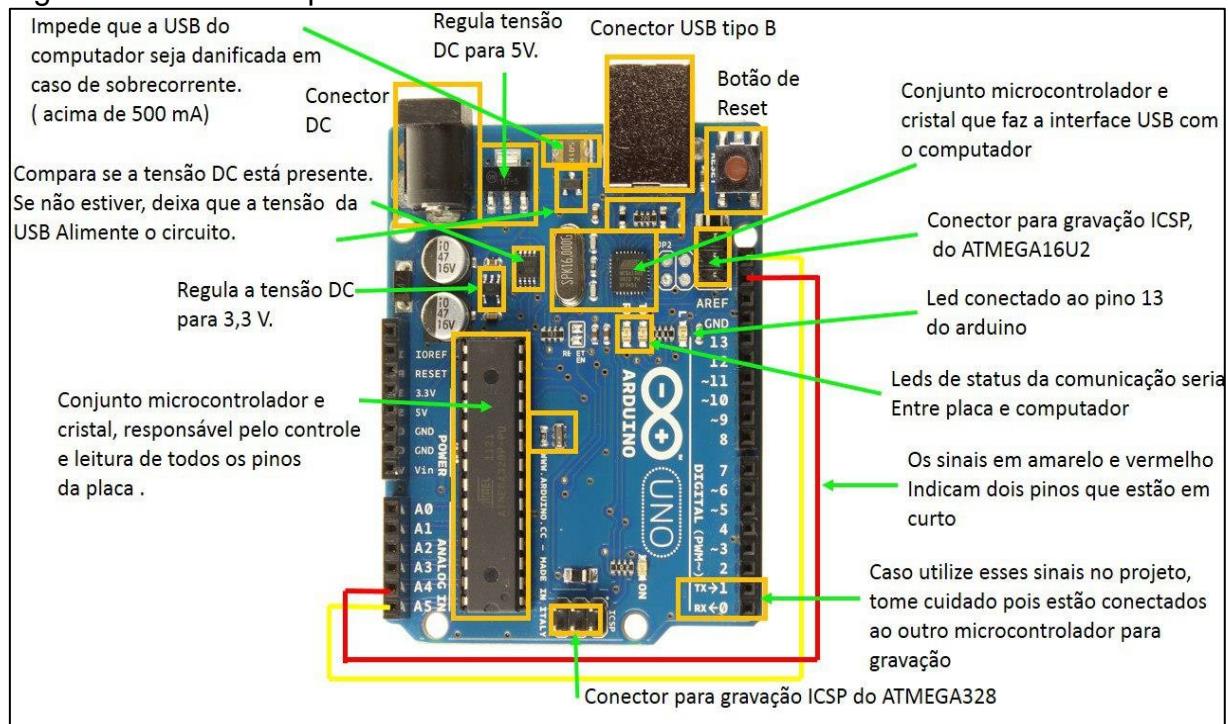
Figura 1.2 – Placa Arduino UNO.



Fonte: Arduino, 2016.

Na figura 1.3 é ilustrada a placa com um resumo de suas configurações.

Figura 1.3 - Resumo placa Arduino UNO

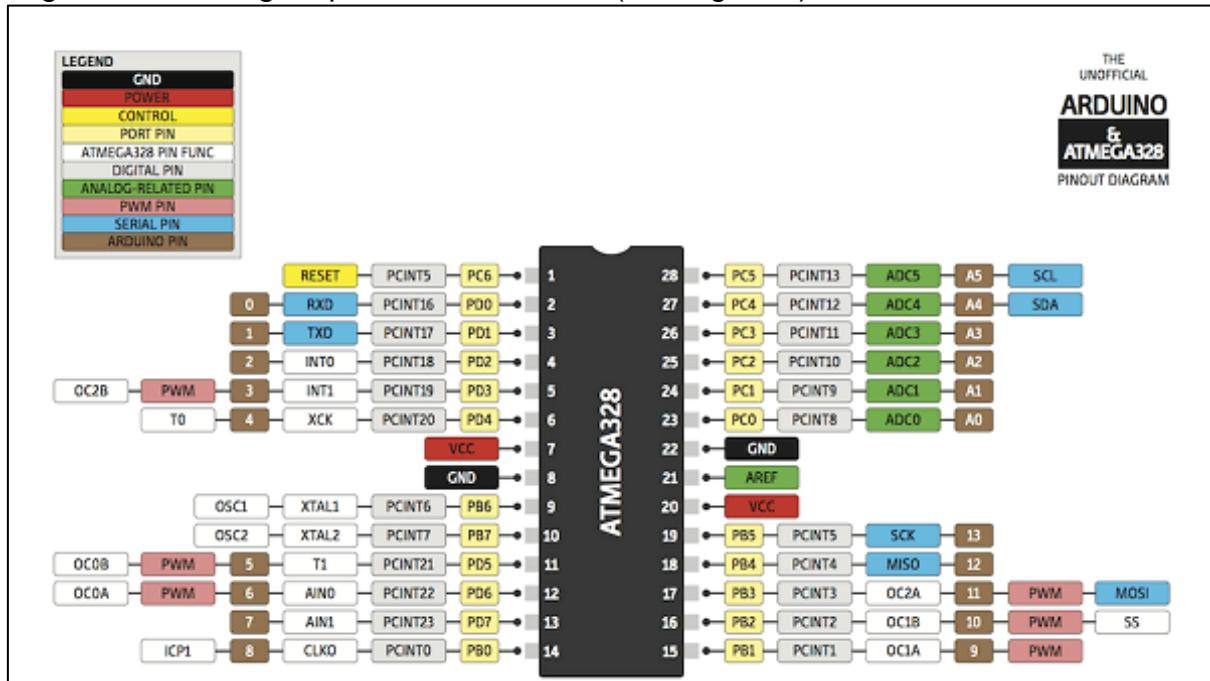


Fonte: Souza, 2013.

O micro controlador ATmega328P é baseado na arquitetura AVR criada pela ATMEL e faz parte dos micros controladores de 8 bits *complementary metal-oxide*

semiconductor (CMOS), além de executar instruções com um ciclo de *clock* com alta performance, alcançando 1 MIPS/MHz (1 Milhão de Instruções por Segundo por Mega Hertz). (ARDUINO, 2016)

Figura 1.4 – Pinagem placa Arduino UNO (ATmega328)



Fonte: Souza, 2013.

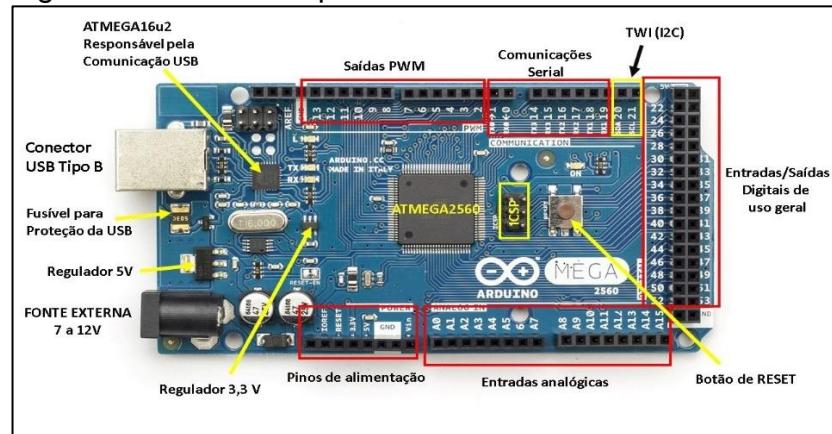
### 1.3 ARDUINO MEGA 2560

O Arduino Mega 2560 é uma atualização para o Arduino Mega, ele possui 2 revisões, além de ser uma das maiores versões existentes devido a sua quantidade de portas, 54 portas digitais de entrada e saída (das quais 15 fornecem uma saída de PWM) e seu micro controlador ATmega2560, sendo este um micro controlador de 8 bits de arquitetura *Reduced Instruction Set Computer* (RISC) avançada, memória flash de 256KB, velocidade de 16MHz e peso de 37g. Essa placa também possui 16 entradas analógicas, 4 *Universal Asynchronous Receiver/Transmitter* (UART) (portas seriais de *hardware*), um cristal oscilador de 16MHz, uma conexão USB, um botão *reset* e *In-Circuit Serial Programming* (ICSP) *header*. (ADUÍNO, 2016)

Essa placa possui uma grande quantidade de porta e memória flash, ideal para expansão de projetos, além de ter um desempenho parecido com a placa Arduino UNO. (SOUZA, 2014)

Na figura 1.5 é exibida a placa e um resumo de suas configurações, mostrando a localização de seus componentes internos.

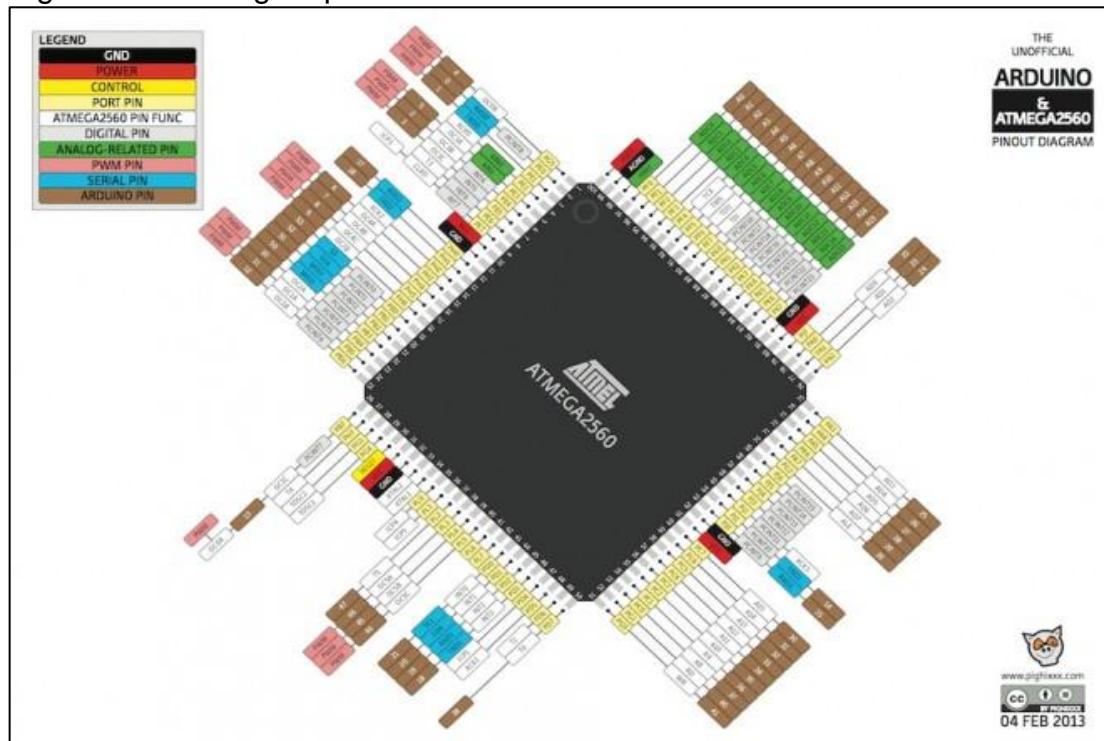
Figura 1.5 – Resumo placa Arduino MEGA 2560.



Fonte: Souza, 2014.

Na figura 1.6 é demonstrada todos os pinos referente a placa, seguindo suas ligações e funções.

Figura 1.6 – Pinagem placa Arduino MEGA 2560



Fonte: Souza, 2014.

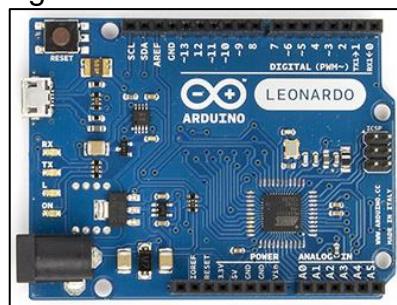
Alguns dos pinos da imagem 1.6 possuem funções especiais como:

- Comunicação Serial: Serial 0 (RX) e 1 (TX); serial 1: 19 (RX) e 18 (TX); serial 2: 17 (RX) e 16 (TX); serial 3: 15 (RX) e 14 (TX). Os pinos 0 e 1 estão conectados aos pinos do ATmega16U2 responsável pela comunicação USB
- Interrupções externas: 2 (*interrupt 0*), 3 (*interrupt 1*), 18 (*interrupt 5*), 19 (*interrupt 4*), 20 (*interrupt 3*), and 21 (*interrupt 2*). Estes pinos podem ser configurados para disparo da interrupção tanto na borda de subida ou descida, ou em níveis lógicos alto ou baixo, conforme a necessidade do projeto. Veja a função *attachInterrupt()* para mais detalhes.
- PWM: os pinos 2 a 13 e 44 a 46 podem ser utilizados como saídas PWM. O sinal PWM possui 8 bits de resolução e é implementado com a função *analogWrite()*.
- Comunicação SPI: Pinos: 50 (MISO), 51 (MOSI), 52 (SCK), 53 (SS). A comunicação SPI pode ser manipulada pela função SPI *library*. Estes pinos estão ligados ao conector ICSP.
- Comunicação I2C: (TWI): pinos 20 (SDA) e 21 (SCL).

#### 1.4 ARDUINO LEONARDO

Uma versão, ilustrada na figura 1.7, que contém um micro controlador Atmel ATmega32u4 e possui 20 pinos digitais de entrada / saída, em que 7 deles podem ser usados como saídas *Pulse-Width Modulation* (PWM), e 12 entradas analógicas, uma conexão USB, um *power jack*, conector *header* ICSP e *clock speed* de 16MHz. Diferentes das outras placas, Leonardo elimina a necessidade de um processador secundário permitindo se conectar com mouse e teclado. (ARDUINO, 2016)

Figura 1.7 – Placa Arduino LEONARDO



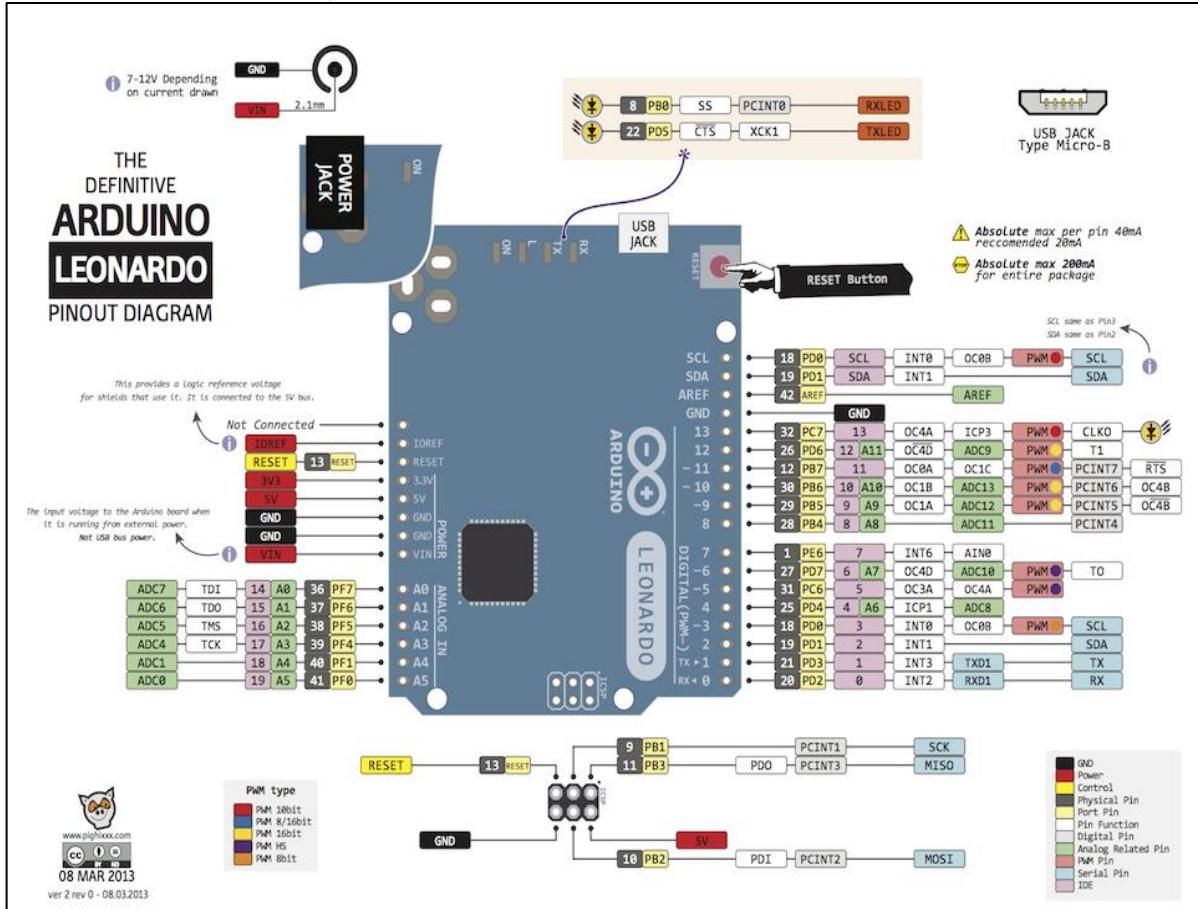
Fonte: Arduino, 2016.

O micro controlador utilizado ATmega32U4 de 8 bits da família AVR com arquitetura RISC avançada que tem encapsulamento *Thin Plastic Quad Flat*

Package (TQFP), também contém 32 KB de memória flash, 2,5 KB de SRAM e 1 KB de *Elettrically Programmable Read-Only Memory* (EEPROM). (SOUZA, 2014)

Na imagem 1.8 é ilustrada as especificações dos pinos da placa Arduino, suas ligações e funcionamento.

Figura 1.8 – Especificação placa Arduino LEONARDO



Fonte: Souza, 2014.

Algumas funções dos pinos mostrados na imagem 1.8 são:

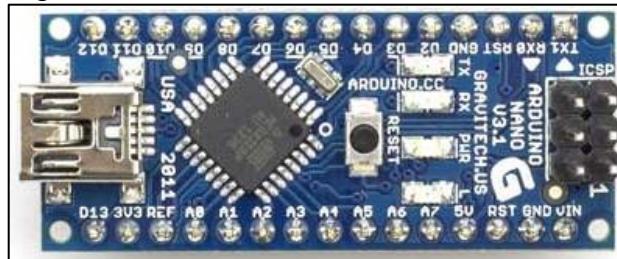
- Comunicação Serial: pinos 0 (RX) e 1 (TX). Esses pinos são usados para enviar e receber dados através de comunicação serial. É interessante notar que para usar esses pinos deve-se usar a classe Serial1, pois a classe Serial refere-se à comunicação USB (CDC).
- Comunicação TWI (I2C): pinos 2 (SDA) e 3 (SCL) permitem a comunicação TWI (I2C da Atmel) usando a biblioteca Wire.

- Interrupção externa: Pinos 3 (interrupt 0), 2 (interrupt 1), 0 (interrupt 2), 1 (interrupt 3) e 7 (interrupt 4). Esses pinos podem ser configurados para disparar uma interrupção quando ocorrer uma mudança no estado, uma borda de descida ou subida, ou um nível baixo. Para mais detalhes verifique a função `attachInterrupt()`.
- Saídas PWM: pinos 3, 5, 6, 9, 10, 11 e 13. Podem ser utilizados como saídas PWM de 8 bits de resolução através da função `analogWrite()`.
- Comunicação SPI: apenas está disponível no conector ICSP e pode ser utilizado com a biblioteca SPI.
- Entradas Analógicas: A interação da interface com o mundo analógico, a Arduino Leonardo possui 12 entradas analógicas. As entradas conhecidas para esse propósito: A0-A5 e mais 6 entradas que estão no lado dos pinos digitais, que são denominadas A6 a A11 e estão respectivamente nos pinos digitais 4, 6, 8, 9, 10 e 12. O conversor analógico/digital (AD) do ATmega32u4 possui resolução de 10 bits e sua referência está ligada internamente a 5V, ou seja, quando a entrada estiver com 5V o valor da conversão analógica digital será 1023. O valor da referência pode ser mudado através do pino `AnalogReference (AREF)`.

## 1.5 ARDUINO NANO

É a menor versão existente e possui um processador baseado no ATmega328 ou ATmega168, essa pequena placa dispõe de 14 pinos I/O digitais, dos quais 6 possibilita saída PWM, 8 pinos analógicos, memória flash 16 KB (ATmega168) ou 32 KB (ATmega328), onde 2 KB são utilizados por *bootloader*. Também possuem de *Static Random Access Memory* (SRAM) 1 KB (ATmega168) ou 2 KB (ATmega328) com um *speed clock* de 16MHz. (ARDUINO, 2016)

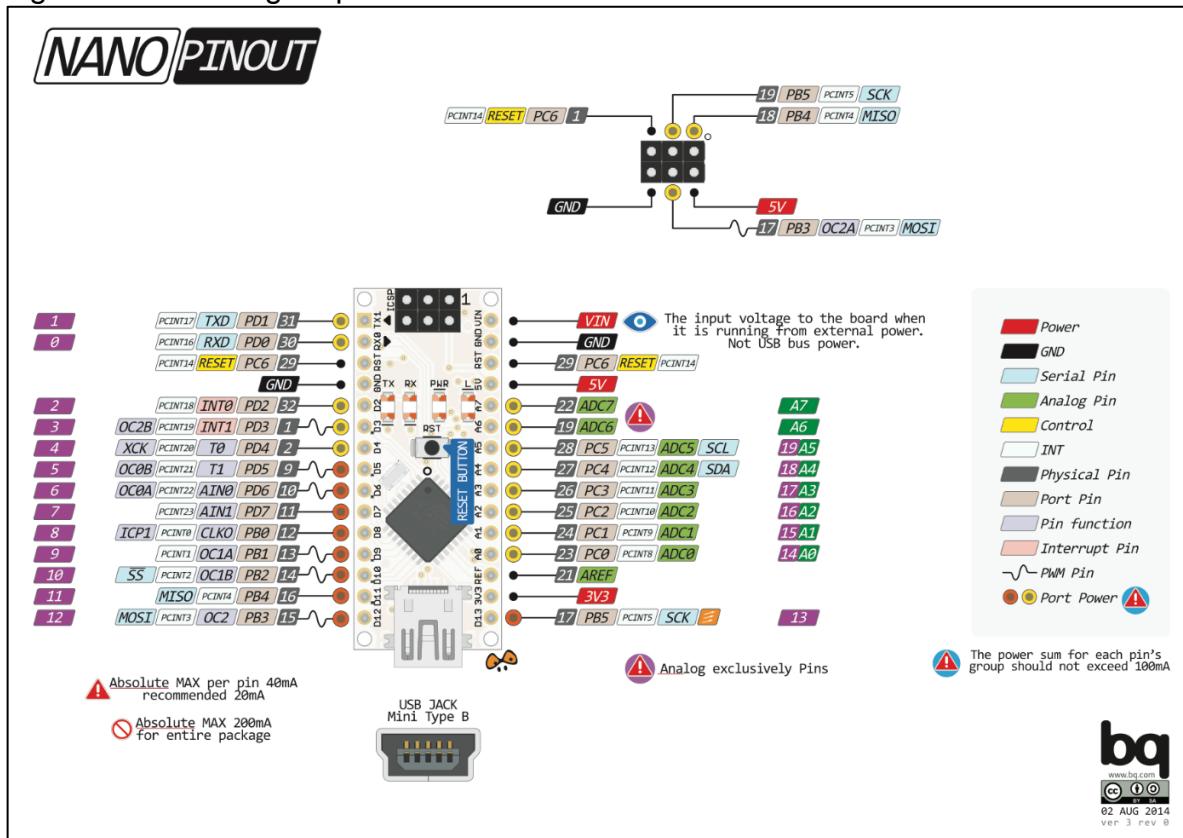
Figura 1.9 – Placa Arduino NANO



Fonte: Arduino, 2016.

Na Figura 1.10 é representado a Pinagem da placa Arduino NANO, referente a figura 1.9, exibindo todos os pinos e suas funções.

Figura 1.10 – Pinagem placa Arduino NANO

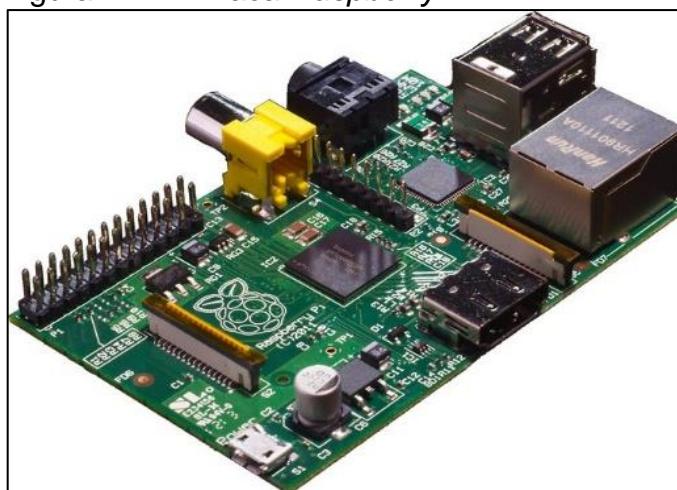


- *Interrupções externas:* 2 e 3: Estes pinos podem ser configurados para disparar uma interrupção por um valor baixo, uma borda de subida ou queda, ou uma mudança de valor. Veja a `attachInterrupt ()` função para obter detalhes.
- *PWM:* 3, 5, 6, 9, 10, e 11: Fornecer saída PWM de 8 bits com a `analogWrite ()` função.
- *SPI:* 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK): Estes pinos suportam comunicação SPI, que, embora fornecido pelo hardware subjacente, não está incluída na linguagem Arduino.
- *LED:* 13: Há um *built-in* LED conectado ao pino digital 13. Quando o pino é de alto valor, o LED está ligado, quando o pino é baixo, ele está fora.

## 1.6 RASPBERRY PI

Um dispositivo no mercado equivalente ao Arduino é o *Raspberry Pi*, exibido na figura 1.11. Esse micro controlador lançado em 2012 criado para fins educacionais, seu tamanho equipara-se a um cartão de crédito, ele apresenta como seus componentes um processador, slots para cartões de memória, interface USB, *High-Definition Multimedia Interface* (HDMI), entrada de energia, barramentos de expansão e seus respectivos controladores. Diferente de um desktop, ele possui processadores construídos a partir dos designs *Advanced RISC Machine* (ARM), que são recomendados para sistemas de controle, máquinas genéricas, unidades que geram menos calor e gastam menos energia. (GARRETT, 2014)

Figura 1.11 – Placa *Raspberry Pi*



Fonte: Garret, 2014.

## 1.7 PRETEXTOS PARA UTILIZAR ARDUINO

De acordo com a própria empresa ARDUINO (2016), a placa é utilizada em diversos projetos, é fácil de usar para os principiantes e também flexível o bastando para usuários avançados, rodando nas principais plataformas como Windows, Mac e Linux. Em meio a outros micros controladores o Arduino oferece uma vantagem aos seus usuários devido a algumas características:

- Barato – as placas são relativamente mais baratas em relação as outra, pode ser montada à mão e seus módulos pré-montado custam pouco.
- Multi-plataforma – sua IDE é executada nos principais sistemas operacionais.
- *Open Source* – é uma ferramenta com código aberto e disponível para a extensão de desenvolvedores experientes, que a modificação tanto de software quanto hardware.

Existem muitos dispositivos baseados no Arduino, como bafômetros, sistemas de automação residencial, sistemas de segurança como leitores de cartão e fechaduras que usam o Arduino com gerenciador. Por exemplo, uma réplica do robô *Mars Rover* criado para explorar a superfície de marte, a sua réplica, criada por Robben Beatty, utiliza um poderoso Arduino Mega e vários motores e servo motores, incluído painéis solares e uma bateria de 7,4 volts e 10000mAh. (ARAÚJO, 2014)

Entres esses projetos está o da automação de uma cancela eletrônica com um sistema para registrar seu uso, que é descrito no próximo capítulo. A automação do projeto tem como base um Arduino UNO e alguns outros componentes ligados a placa que ajudam na comunicação da placa com o sistema e o meio.

## 2 AUTOMAÇÃO DA CANCELA E COMPONENTES UTILIZADOS

Neste capítulo são tratados definições e conceitos a respeito da automação da cancela e as estruturas de componentes que serão utilizados no projeto, a ideia de automação está diretamente ligada a ideia de máquinas que agilizam as tarefas quase sempre sem interferência humana, são técnicas mecânicas ou informatizadas que tem o objetivo de dinamizar e otimizar todos os processos produtivos dos mais diversos setores da economia atual.

O projeto tem como base uma placa de Arduino UNO e ligada a ela estão alguns outros componentes que facilitam no tratamento de dados recebidos e enviados, esses componentes são: *protoboard*, servo motor, sensor de RFID, *tags* RFID, painel de LED, uma *etherne shield* e um cartão *Secure Digital* (SD).

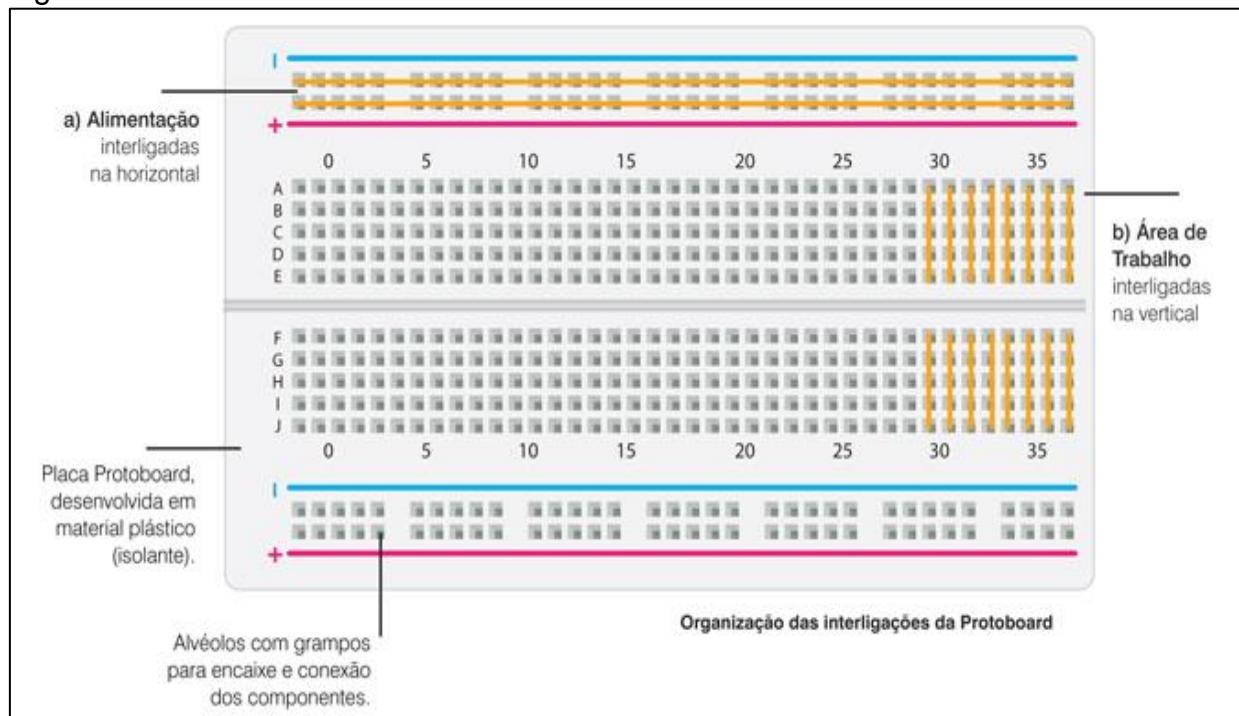
A ligação desses componentes com o Arduino é feita através da placa *protoboard*.

### 2.1 PROTOBOARD

*Protoboard* ou *breadboard* é uma placa de ensaio com base para construção de protótipos de circuitos eletrônicos, essa matriz de contato com orifícios e conexão condutoras para montagem de circuitos eletrônicos, possui uma grande facilidade na montagem de componentes, não sendo necessário soldagem. As placas variam de 800 até 6000 furos, permitindo conexões horizontais e verticais, mas há problemas com mau-contato.

A base dessa placa é feita de plástico com os furos onde são encaixados os componentes. Na parte inferior são instalados contatos metálicos que interligam eletricamente os componentes inseridos na placa, geralmente suportando correntes entre 1A e 3A. Os contatos metálicos estão em diferentes sentidos na matriz. O layout típico de uma placa de ensaio é composto de áreas constituídas em terminais elétricos interligados. (FRONTEIRA TEC, 2013)

Figura 2.1 – Resumo Protoboard



Fonte: Dream Inc., 2016.

### 2.1.1 Faixa de terminais

Faixas de contatos no qual são instalados os componentes eletrônicos. Em suas laterais existem duas trilhas de contatos interligadas verticalmente. No centro da placa há um entalhe central fornecendo um fluxo de ar para permitir um melhor arrefecimento de circuitos integrados e outros componentes. Entre as faixas laterais e o entalhe central existem trilhas de cinco contatos dispostas paralelamente e interligadas horizontalmente.

### 2.1.2 Faixas de barramentos

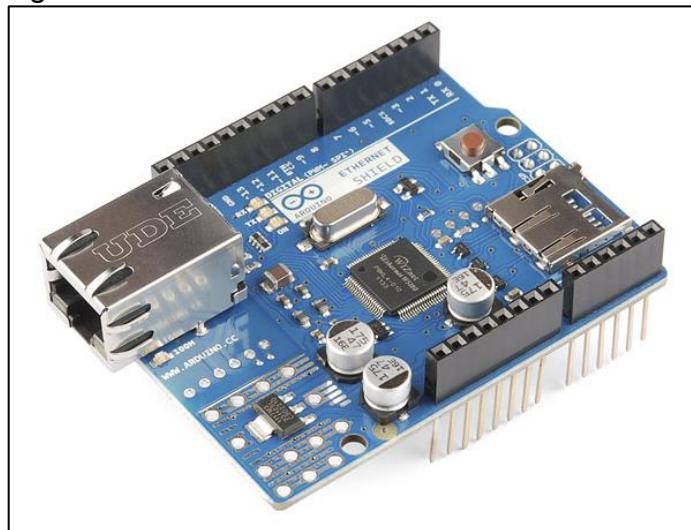
Responsáveis pelo fornecimento de tensão ao circuito, constituídas de duas colunas nas laterais, uma para o condutor negativo ou terra, e outra para o positivo.

## 2.2 ARDUINO ETHERNET SHIELD

O Arduino *Ethernet Shield* é um módulo que conecta o Arduino na rede local de maneira rápida e prática, basta conectar-lo na placa Arduino e na rede a partir de

um cabo com terminal RJ45. Baseado na *Wiznet W5100 chip*, que fornece um *Internet Protocol(IP)* para rede suportando *Transmission Control Protocol (TCP)* e *User Datagram Protocol (UDP)*, suportando até quatro conexões de soquete simultâneos. (ARDUINO, 2016)

Figura 2.2 – Arduino Ethernet Shield.



Fonte: Arduino, 2016.

A *Ethernet* tem um padrão de conexão RJ-45, com transformador de linha integrado e *Power over Ethernet (PoE)* habilitado. Possuindo um *slot* de cartão SD integrado, podendo armazenar arquivos para servir através da rede. Sendo compatível com todas as placas Arduino.

O *Ethernet Shield* também inclui um controlador de reposição, garantindo que o modulo W5100 *Ethernet* está devidamente em *power-up*. Sua comunicação é feita tanto com o W5100 e o cartão SD utilizando o barramento SPI através do *header ICSP* e por meio da biblioteca *Ethernet* se faz a conexão com a internet. (ARDUINO, 2016)

Para usar esse módulo são necessários alguns requisitos e componentes:

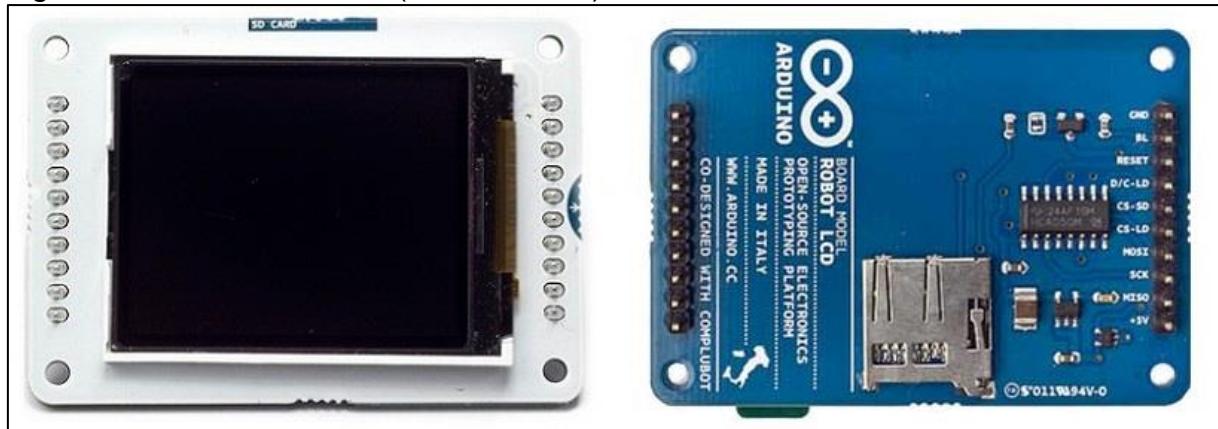
- Requer uma placa Arduino;
- 5V tensão de serviço (fornecido pela placa Arduino);
- Controlador de Ethernet: w5100 com buffer interno de 16K;
- Velocidade de conexão: 10/100Mb;
- Ligação com Arduino na porta SPI.

## 2.3 ARDUINO TFT LCD

A tela TFT Arduino é uma tela LCD retro iluminado com *headers*, onde é possível transmitir textos, imagens e formas para a tela com a biblioteca TFT. Os *headers* da tela são projetados para caber no soquete na parte da frente do Arduino, sendo compatível com qualquer um deles baseados em AVR Arduino (Uno, Leonardo, etc.). (ARDUINO, 2016)

A tela é de 1,77" polegadas com resolução de 160 x 128 pixel, com controlador de interfaces através de SPI quando se utilizar a biblioteca TFT. Além de incluir um slot para cartão SD.

Figura 2.3 – Tela TFT LCD (Front & Back).

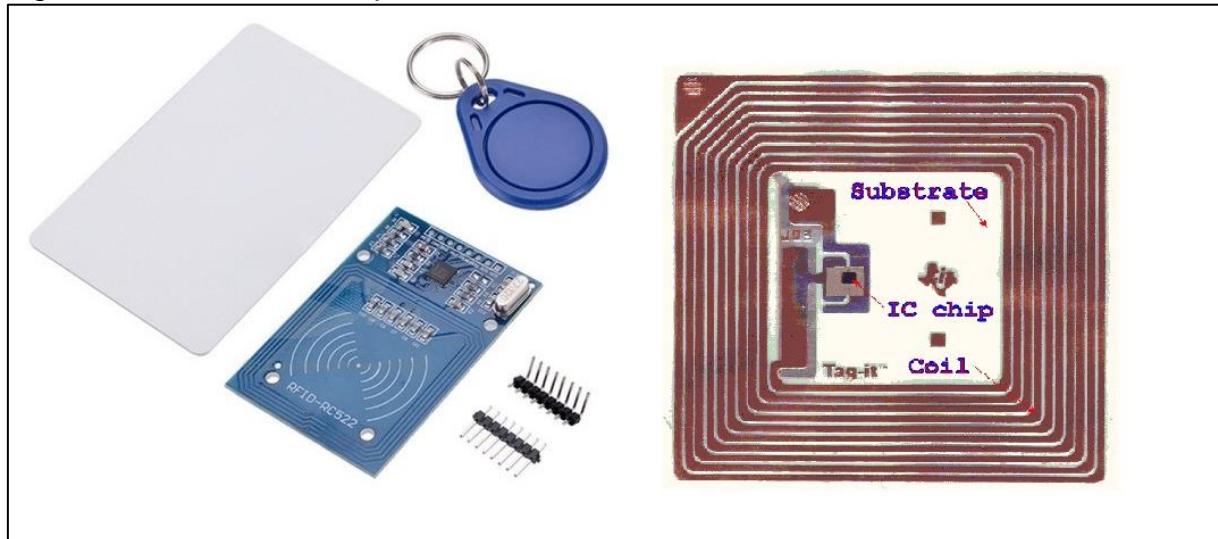


Fonte: Arduino, 2016.

## 2.4 SENSOR MFRC522 E TAG RFID

Segundo THOMSEN, leitores e *tags* RFID geralmente são bastantes utilizados para controle de acesso e identificação de pessoas e equipamentos, basicamente consiste de uma antena, um transceptor, que faz a leitura do sinal e transfere a informação para um dispositivo leitor e uma etiqueta de rádio frequência que deve conter o circuito e a informação a ser transmitida. A antena transmite a informação, emitindo um sinal do circuito integrado para transmitir suas informações para o leitor, que por sua vez converte as ondas de rádio do RFID para informações digitais, após convertidas, elas poderão ser lidas e compreendidas por um computador para então ter seus dados analisados. (THOMSEN, 2014)

Figura 2.4 – Sensor & Etiquetas RFID.



Fonte: Thomsen, 2014.

As etiquetas (ou *tag*) RFID, podem conter vários dados sobre o proprietário do cartão. Como são compostas apenas por um pequeno circuito, as *tags* RFID podem ser embutidas facilmente em vários objetos, nos mais variados tamanhos e formatos. Cada etiqueta do leitor RFID possui sua própria identificação *Unique Identification* (UID).

#### **2.4.1 Tipos de etiquetas RFID:**

##### **2.4.1.1 Passiva**

Estas etiquetas utilizam a rádio frequência do leitor para transmitir o seu sinal e normalmente vem com suas informações gravadas permanente quando são fabricadas, porém algumas delas são "regraváveis".

##### **2.4.1.2 Ativa**

Essas etiquetas são mais caras e possuem bateria própria para transmitir seu sinal sobre uma distância razoável, além de permitir armazenamento em memória RAM capaz de guardar até 32 KB.

O leitor RFID é um modulo RFID RC522 para Arduino capaz de ler *tags* que operam na frequência de 13,56Mhz, suportando cartões tipo Mifare1 S50, Mifare1

S70 *Mifare Ultralight*, *Mifare Pro* e *Mifare DESFire*, utilizando a interface SPI para comunicação com Arduino e usando a biblioteca MFRC552, além de conter 8 pinos que seguem a seguinte sequência de ligação (no caso a alimentação do modulo é de uma tensão de 3.3 volts). (ARDUINO E CIA, 2014)

#### Pinos de ligação do leitor RFID na placa Arduino

- Pino SDA ligado na porta 10 do Arduino.
- Pino SCK ligado na porta 13 do Arduino.
- Pino MOSI ligado na porta 11 do Arduino.
- Pino MISO ligado na porta 12 do Arduino.
- Pino NC Não Conectado.
- Pino GND ligado no pino GND do Arduino.
- Pino RST ligado na porta 9 do Arduino.
- Pino 3.3 ligado no pino 3.3 V do Arduino.

## 2.5 RESISTOR

Resistor é um dispositivo capaz de transformar energia elétrica em energia térmica através do efeito joule, em que joule é uma lei física que expressa a relação entre o calor gerado e a corrente elétrica que percorre um condutor em um determinado tempo, o resistor também funciona como um limitador de corrente elétrica em um circuito, eles oferecem oposição à passagem de corrente elétrica, por meio de seu material, e a medição da resistência de um resistor tem relação com a tensão da corrente que é medida em ohm, uma unidade do Sistema Internacional de Unidades. Os resistores são usados como parte de um circuito elétrico e incorporados dentro de dispositivos microeletrônicos ou semicondutores. (SILVA, 2016)

Figura 2.5 – Exemplos Resistor.

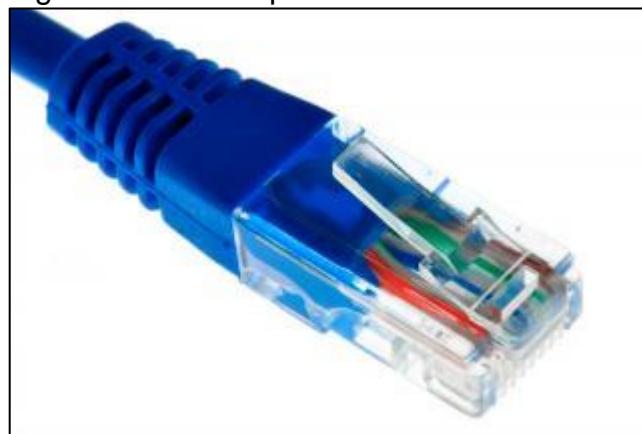


Fonte: Jimbo 2016.

## 2.6 CABO DE REDE

Cabeamento por par trançado é um tipo de cabo que possui pares de fios entrelaçados um ao redor do outro para cancelar as interferências eletromagnéticas. Existem cabos de categoria 1 até categoria 7. O cabo *Unshield Twisted Pair* (UTP) é o mais usado atualmente em redes domésticas e em empresas devido ao fácil manuseio, instalação e permitindo taxas de transmissão de até 100mbps, tendo como conector RJ45 e com a utilização do cabo Categoria 5e “enhanced”, uma versão melhorada da CAT 5, sendo barato e cobrindo distâncias de até 100 metros, entretanto ele não possui blindagem de malha metálica que evitam ainda mais interferências eletromagnéticas. (MORIMOTO, 2008)

Figura 2.6 – Exemplos Cabo UTP com Conector RJ45



Fonte: Stockvault, 2012.

Na figura 2.7 é demonstrado um cabo UTP categoria 5e com os fios entrelaçados

Figura 2.7 – Exemplo de cabo UTP CAT 5e.



Fonte: Morimoto, 2008.

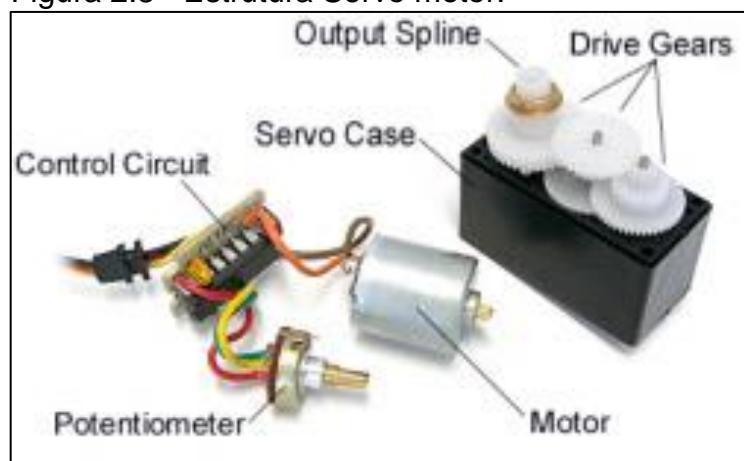
Na figura 2.7, é demonstrado um cabo UTP, onde as placas de rede utilizam o sistema “*balanced pair*” de transmissão, para potencializar o efeito da blindagem eletromagnética, os dois fios dentro de cada par, enviam o sinal juntos, mas com polaridade invertida. O primeiro fio envia um sinal elétrico positivo para um bit “1”, o

segundo fio é usado para enviar uma cópia invertida da transmissão enviada através do primeiro, o que tira proveito das tranças do cabo para criar o campo eletromagnético que protege os sinais contra interferências externas, mesmo nos cabos sem blindagem. Esta técnica de transmissão, os cabos de par trançado são também chamados de "*balanced twisted pair*". (MORIMOTO, 2008)

## 2.7 SERVO MOTOR

Servo motor é um componente eletromecânico, como demonstrado na imagem 2.8, que recebe um sinal de controle, apresentando um movimento proporcional a um comando, que verifica sua atual posição controlando seu movimento e sua velocidade. O circuito servo é construído ao lado do motor e possui um eixo posicionado, que geralmente é equipado com uma engrenagem. O motor é controlado por um sinal elétrico, que determina seu movimento. Sua estrutura contém um pequeno motor *Direct Current* (DC), ou seja, de corrente continua, possui também um potenciômetro e um circuito de controle. Seu motor é ligado por engrenagens para controlá-las. As mudanças de resistência possibilitam o controle do movimento e direção das rotações. (REED, 2016)

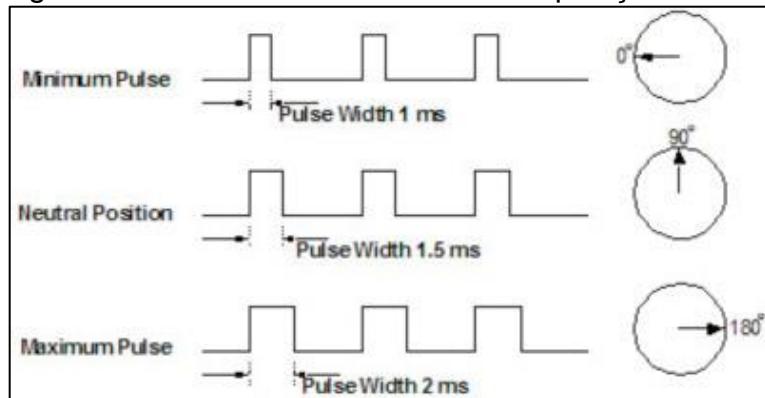
Figura 2.8 - Estrutura Servo motor.



Fonte: Reed, 2016.

O servo motor é controlado por um pulso elétrico de largura variável (PWM). Existe um pulso mínimo, máximo e uma posição neutra, sendo ilustrado na figura 2.9. Normalmente um servo pode virar 90 ° graus em qualquer direção em um total de 180 graus de movimento.

Figura 2.9 - Pulso Variável de controle posição servo motor.



Fonte: Reed, 2016.

## 2.8 SENSOR ULTRASSÔNICO

Este dispositivo, também conhecido como transceptor, utiliza frequência de som para medição de distância e detecção de posição de objetos, sendo que a maioria dos objetos sólidos são capazes de refletir ondas sonoras. O tipo deste sensor difere dos demais, pois ele também pode medir variáveis como curvatura, altura e enchimento, sem precisar entrar em contato com o material. O sensor ultrassônico se assemelha, no modo de operar, a um sonar, porém seu uso é mais comum em ambiente terrestre, tendo o ar como meio de transmissão.

Figura 2.10 – Sensor Ultrassônico



Fonte: Mecânica Industrial, 2016.

O sensor consiste, characteristicamente, de uma unidade de transceptor único, que é capaz de emitir e detectar o som, criando um pulso sonoro que está além da faixa de audição do ouvido humano. O transceptor também faz uso de um temporizador que determina com precisão quanto tempo um pulso ultrassônico leva para “saltar” em um objeto e retornar à unidade.

### **3 ESPECIFICAÇÃO, ANÁLISE E ESTRUTURA DO PROJETO**

#### **3.1 ANÁLISE E PROJETO DO SISTEMA**

Neste capítulo é definida uma análise do projeto do sistema que engloba desde a situação problema a qual nos trouxe a oportunidade de concretizar este projeto até a modelagem do sistema utilizando a Linguagem de Modelagem Unificada baseada em Engenharia de Software que nos permite representar um sistema de forma padronizada com o intuito de facilitar a compreensão de todos, foram desenvolvidos os diagramas de casos de uso, classes, atividades, sequência, componentes e o *Model-View-Controller* (MVC) seguido com a modelagem do banco de dados.

Segundo GONÇALVES (2007), a finalidade da *Unified Modeling Language* (UML) é proporcionar um padrão para a preparação de planos de arquitetura de projetos de sistemas, incluindo aspectos conceituais, como processos de negócios e funções de sistema, além de itens concretos, como as classes escritas em determinada linguagem de programação, esquemas de bancos de dados e componentes de softwares reutilizáveis.

De acordo com MEDEIROS (2004) a linguagem auxilia o desenvolvedor durante a modelagem do sistema. A UML é uma linguagem que utiliza uma notação padrão para especificar, construir, visualizar e documentar sistemas de informação orientados por objetos. Esta linguagem é utilizada para a documentação dos sistemas, e propicia a definição da arquitetura do software representado pelos diagramas que a UML oferece. Esta linguagem deve ser utilizada para definir o funcionamento do sistema desenvolvido em suas mais diversas fases.

#### **3.2 ANÁLISE DE NEGÓCIO**

A finalidade do presente tópico é apresentar e identificar toda a situação da instituição, tendo como principal objetivo investigar e coletar informações sobre os problemas e necessidades visando atingir uma solução habilitada a suprimir os problemas encontrados.

#### **3.3 DESCRIÇÃO DO PROBLEMA**

As organizações possuem dificuldades em relação a administração e controle de acesso dos seus colaboradores e suas devidas conduções, existe uma grande escassez no mercado a respeito de serviços que provêm esse acesso gerenciado a determinados ambientes destituindo assim o acordo de palavras e atribuindo total precisão ao sistema, além de propiciar os registros desses acessos e formas de consultas instantâneas dos mesmos.

A instituição pública Fatec do governo do estado de São Paulo, não possui qualquer tipo de gerenciamento dos seus alunos e nem mesmo de seus funcionários, pensando no fator de gestão de ambientes como estacionamentos, salas de aulas, laboratórios a proposta do nosso trabalho tem o objetivo maior de proporcionar soluções como acessos controlados a determinados ambientes, a identificação automática e a aprovação ou não a esses lugares previamente definidos pelos administradores, registros dessas eventuais admissões a espaços delimitados tudo isso estabelecido por meio de sinais de rádio além de efetivar o armazenamento de dados por meio de dispositivos denominados TAGS RFID como já foi mencionado em capítulos anteriores.

Através destes serviços descritos no parágrafo anterior, pode se concretizar um grande catálogo de vantagens e recursos obtidos com a implantação do sistema de controle e gerenciamento de dispositivos de acesso.

### **3.4 ATORES ENVOLVIDOS**

No sistema estão envolvidos dois atores, dentre eles o ator administrador que tem por sua responsabilidade gerir todos os usuários do sistema, gerenciar os registros de acessos, alterar perfis, relacionar etiquetas RFID aos usuários, relacionar departamentos dentre outras funções e o ator usuário comum que possui exclusivamente a função de consultar os seus registros pessoais de acessos.

### **3.5 PERSPECTIVA DO PRODUTO**

O sistema desenvolvido para o ambiente da Faculdade de Tecnologia de Lins, ao qual assim como diversas organizações, carece de uma solução com a mesma premissa da proposta apresentada. A intenção do serviço é oportunizar ao usuário e também a gerência da instituição uma forma de controlar ambientes a entidades permitidas a adesão, impossibilitando acessos não permitidos e não verificados,

existe também a possibilidade de supervisionar todos os registros de acessos contidos no sistema e com a suscetível implantação do sistema integrado o fluxo de informações, a gestão de ambientes e o controle de usuários serão efetivamente aperfeiçoados.

### **3.6 CARACTERÍSTICAS**

As funcionalidades do sistema são identificadas e apresentadas de acordo com os atores.

#### **3.6.1 Funcionalidades do Usuário Administrador:**

**Gerenciar Usuário:** o administrador do sistema pode cadastrar o usuário/funcionário e manter todas as informações do mesmo inclusive relacionar o usuário ao seu devido departamento, função, automóvel e etiquetas RFID, está funcionalidade o permite cadastrar, alterar e remover o usuário.

**Gerenciar Log de Registro:** o administrador do sistema pode visualizar todas as informações do log de registro.

**Gerenciar Automóvel:** o administrador do sistema pode cadastrar o automóvel e manter todas as informações do mesmo inclusive relacionar o automóvel ao seu proprietário, esta funcionalidade também permite cadastrar, alterar e remover o automóvel.

**Gerenciar RFID:** o administrador do sistema pode cadastrar a etiqueta RFID e manter todas as informações do mesmo possibilitando também, relacionar a etiqueta ao usuário proprietário, esta funcionalidade permite cadastrar e remover a etiqueta RFID.

**Gerenciar Dispositivos:** o administrador do sistema pode cadastrar o dispositivo e manter todas as informações do mesmo, propiciando também a relação do dispositivo com os atores do sistema, esta funcionalidade permite cadastrar, remover e alterar as informações do dispositivo.

**Gerenciar Local:** o administrador do sistema pode cadastrar o local, manter todas as informações do ambiente e relacionar o mesmo ao usuário que uma vez já relacionado pode adentrar aos locais determinados por esta função, está atribuição permite também cadastrar local, alterar e remover o mesmo.

**Gerenciar Departamento:** o administrador do sistema pode cadastrar o departamento, manter todas as informações do mesmo e o relacionar ao seu usuário da instituição, esta funcionalidade também permite cadastrar, alterar e remover o departamento.

**Gerenciar Função:** o administrador do sistema pode cadastrar a função e manter todas as informações do mesmo, relacionando-o com o usuário, está funcionalidade também permite cadastrar, alterar e remover a função dos atores do sistema.

**Fazer Login:** para realizar o processo de *login* o usuário do sistema deve ser verificado pelo sistema através da base de dados, se o mesmo for identificado como administrador pode utilizar efetivamente todas as atividades e funcionalidades do sistema, ou caso seja usuário comum, será limitado ao seu registro pessoal.

### 3.7 ANÁLISE DE REQUISITOS

A presente análise tem como objetivo englobar todas as tarefas da engenharia de requisitos, identificando as necessidades ou requisitos de um cliente através de pesquisas, reuniões e investigações difundindo o mesmo em duas fases, requisitos funcionais que expressam uma função de um sistema ou seu componente e requisitos não funcionais que se define em requisitos relacionados ao uso da aplicação em termos de desempenho, usabilidade dentre vários outros envolvidos, ou seja mostrar detalhadamente as funcionalidades do sistemas expressando o mesmo através de diagramas e suas respectivas descrições.

### 3.8 ANÁLISE DE REQUISITOS FUNCIONAIS

Fundamentando-se nos atores envolvidos administrador e usuário comum as seguintes funcionalidades podem ser identificadas: gerenciar usuário, gerenciar logs de registros, gerenciar automóvel, gerenciar RFID, gerenciar dispositivos, gerenciar local, gerenciar departamento, gerenciar função, efetuar *login* e consultar logs de registros.

### 3.9 ANÁLISE DE REQUISITOS NÃO FUNCIONAIS

**Usabilidade:** A usabilidade do sistema é de grande clareza e simplicidade, a ferramenta *Primefaces* utilizada no projeto proporciona ao sistema uma experiência de grande facilidade na tomada de decisões, se tratando da parte física do projeto o mesmo não é necessário, partindo da premissa que o software informatizara qualquer ação humana, sendo assim registrando logs de acesso, emitindo sinais digitais aos dispositivos que deverão efetuar a operação determinada dependente da escolha que assim é coletada a partir do sinal que virá por meio de cabos físicos e lógica de programação, em questão de acesso ao sistema o mesmo pode ser acessado e gerenciado de qualquer estação.

**Confiabilidade:** O sistema tem 95% de confiabilidade pelo fato de necessitar de energia para ocasionar o funcionamento, em caso de queda de energia o sistema torna-se inoperante e para solucionar este problema deve-se adquirir e implantar um *No-break* capaz de suportar a falta de energia por uma determinada fatia de tempo e nesse meio tempo um gerador de energia deve ser ativado para os dispositivos funcionarem corretamente, se tratando da energia provida aos servidores da instituição isso já é devidamente gerenciado pelo setor de tecnologia da informação.

**Padronização do sistema:** O sistema atende ao padrão Modelo-Visão-Controle (MVC) permitindo a fácil reparação e manutenção do projeto.

**Controle de Acesso:** O sistema utiliza um controle de acesso fundamentado por níveis, propiciando uma melhor gestão. O acesso ao sistema web é realizado a partir da verificação de *login* e senha de usuário, se ele for verificado como um usuário terá acesso simples sem privilégios administrador tendo assim a única possibilidade da leitura de logs de acesso a partir de qualquer funcionário da instituição, caso ele seja verificado como um administrador terá total acesso ao sistema, tanto na parte de verificação de logs de acesso de funcionários quanto também a parte de cadastramento, remoção e alteração dos atores.

**Backup e Restauração:** O sistema possui backups de informações cadastrais e de logs de acesso, são mantidos em um Servidor localizado na instituição e também externo a instituição nas nuvens com o seu devido backup armazenado em outra estação fora do ambiente físico da instituição, os backups são

realizados diariamente e testados semanalmente em caso de destruição da informação não há a perca de dados.

**Tolerante a Falhas:** O sistema foi analisado com as possíveis possibilidades de falhas nas seguintes situações: Indisponibilidade de energia, esta situação comprometeria totalmente o funcionamento dos dispositivos que logicamente devem ser abastecidos com a determinada demanda de energia para exercer funções elétricas e mecânicas. Indisponibilidade de servidor web, caso o servidor web esteja indisponível os dispositivos continuaram verificando o acesso porém os logs do mesmo serão registrados temporariamente em uma memória não volátil que é introduzida no Arduino para manter esses dados que por conta da indisponibilidade do servidor não poderá ser registrado e mantido nos espaços de armazenamento, quando esta situação for reparada deverá existir uma interação homem-computador para recuperar fisicamente estes dados que estarão salvos na memória *flash* da placa controladora Arduino.

**Performance:** A comunicação e performance é mantida e baseada na tratativa e usabilidade de meios físicos como cabo de par trançado que são traçados em uma metragem posteriormente definida, o mesmo deverá ser espetado na placa física do Arduino e no módulo de rede central logo assim interligado a intranet do instituto público, provendo a este a comunicação da estrutura lógica do Arduino com a rede mundial de computadores. O sistema desenvolvido em Java Web trata as suas classes e extensões de maneira organizada e ágil tornando assim o sistema em uma leve aplicação utilizada através do seu *web browser*.

**Necessidades do Sistema:** O sistema requisita um Servidor Web disponível na instituição que tem como objetivo permitir o acesso dos seus usuários e manter o sistema operante vinte e quatro horas em plataforma web.

### 3.10 DIAGRAMA DE CASO DE USO

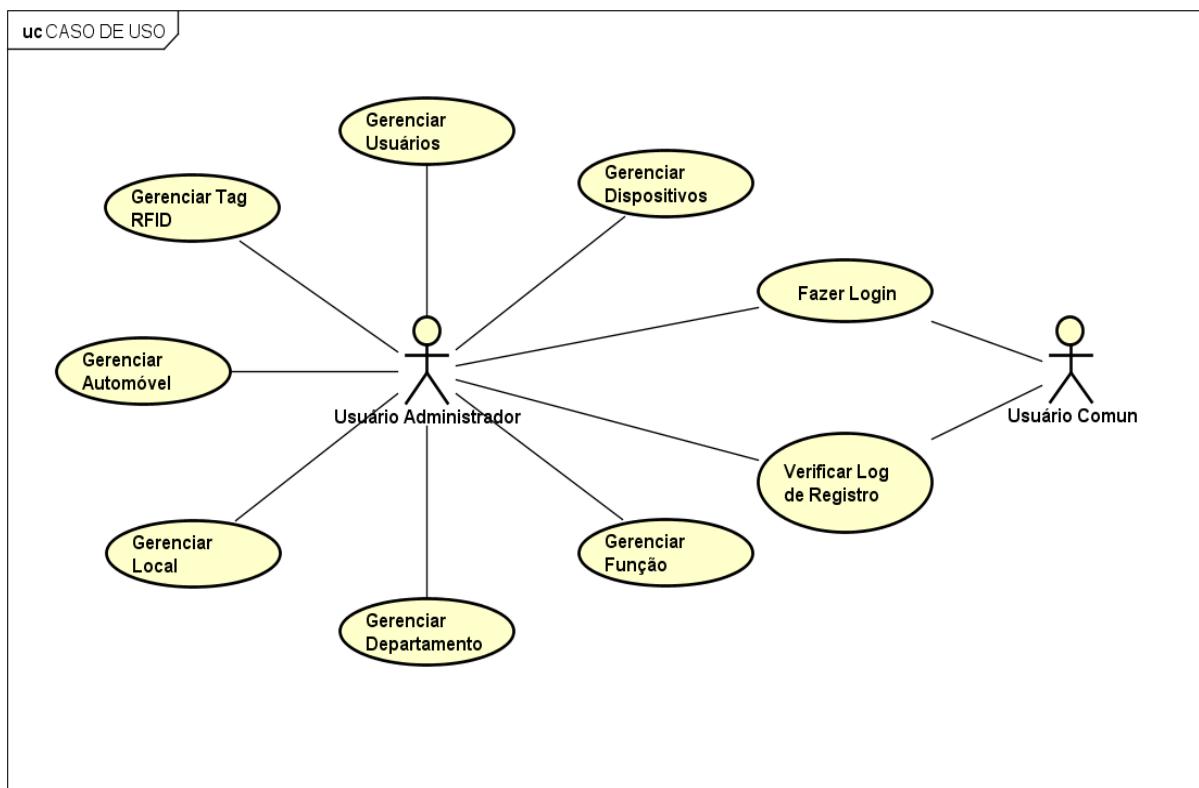
O Diagrama de Casos de Uso tem como objetivo de auxiliar a comunicação entre os analistas e os clientes, o mesmo descreve um cenário que mostra as

funcionalidades do sistema do ponto de vista do usuário, o cliente deve ver no diagrama de Casos de Uso as principais funcionalidades do sistema.

O Diagrama de Caso de Uso é o diagrama mais informal da UML, sendo utilizado principalmente para auxiliar no levantamento e na análise dos requisitos, descreve as relações entre os requisitos, os usuários e os principais componentes, auxilia na compreensão do sistema como um todo, além disso, serve como base para a maioria dos outros diagramas. (GUEDES, 2014).

Segundo MEDEIROS (2004) Caso de Uso é a parte mais importante da construção de software orientado a objetos utilizando a UML. Os Casos de Uso são, talvez, o único instrumento que acompanha um software do seu início até a conclusão. Em todas as iterações que vão ocorrendo na confecção do novo software, o Caso de Uso é a ferramenta de consulta, acerto, discussão, reuniões, alterações em requisitos e alterações em desenho, ele é a análise intrínseca de um negócio, dentro do processo de desenvolvimento de software, sugerido pelo processo iterativo e por outras metodologias que se utilizam da UML.

Figura 3.1 - Diagrama de Casos de Uso do Sistema



Fonte: Elaborado pelos autores, 2016.

### 3.10.1 Especificação de Caso de Uso

#### 3.10.1.1 Caso de Uso: Fazer Login

Este caso de uso representa a funcionalidade responsável pelo *login* do usuário, validando ou não seu acesso ao sistema ao verificar se a entrada de login e senha estão no registro.

#### Fluxo Básico

- fb1: O Usuário *Administrador/Comum* inicia o caso de uso
- fb2: O Usuário *Administrador/Comum* insere seu *login* e senha
- fb3: O sistema verifica se os dados estão cadastrados
- fb4: O sistema exibe a interface de acordo com o tipo de usuário
- fb5: O caso de uso é encerrado

#### Fluxos Alternativos

- fa1: Usuário inválido – No fluxo básico **fb3**, caso o usuário não seja válido, o sistema notifica o usuário que seu acesso não é permitido. Retornar ao fluxo **fb5**.
- fa2: Usuário válido – No fluxo básico **fb3**, caso o usuário seja válido e seja do tipo *administrador*, o sistema exibe a interface de gerencia do sistema com todos os privilégios, retornar ao fluxo **fb4**, caso o usuário seja válido e seja do tipo *comum*, o sistema exibe a interface limitada com informações pessoais do seu usuário, retornar ao fluxo **fb4**.

#### 3.10.1.2 Caso de Uso: Verificar Log de Registro

Este caso de uso representa a funcionalidade que é responsável pela visualização dos Logs de Registros dos atores usuários.

#### Fluxo Básico

- fb1: O Usuário *Administrador/Comum* inicia o caso de uso
- fb2: O sistema exibe todos os registros de acesso
- fb3: O caso de uso é encerrado

## Fluxos Alternativos

fa1: Tipo de Usuário – No fluxo básico **fb2**, caso o usuário seja do tipo *administrador*, o sistema exibe a interface com o registro de os usuários que fizeram uso do sistema com dispositivo, retornar ao fluxo **fb3**, caso o usuário seja do tipo *comum*, o sistema exibe a interface limitada com registros pessoais do seu usuário, retornar ao fluxo **fb3**.

### 3.10.1.3 Caso de Uso: Gerenciar Usuários

Este caso de uso representa a funcionalidade responsável pelo cadastro, alteração de dados e também a remoção do mesmo em relação a todos os usuários cadastrados do sistema, atribuindo a um departamento, função, tipo e *tag*, se necessária.

#### Fluxo Básico

- fb1: O Usuário *Administrador* inicia o caso de uso
- fb2: O sistema exibe todos os usuário cadastrados
- fb3: O Usuário *Administrador* realiza o cadastro de usuário
- fb4: O Usuário *Administrador* realiza a atualização dos dados de usuário
- fb5: O Usuário *Administrador* realiza a remoção dos dados de usuário
- fb6: O caso de uso é encerrado

#### Fluxos Alternativos

- fa1: Efetuar Cadastro – No fluxo básico **fb3**, caso um usuário já seja cadastrado, o sistema cancela o cadastro. Retornar ao fluxo **fb6**.
- fa2: Remover Dados: No fluxo básico **fb5**, caso o usuário já cadastrado tenha um veículo já vinculado, o automóvel é removido também, retornar ao fluxo **fb6**.

### 3.10.1.4 Caso de Uso: Gerenciar Dispositivo

Este caso de uso representa a funcionalidade responsável pelo cadastro de dispositivos, alterar de informações dos dispositivos e efetuar a remoção do mesmo, além de atribui-lo a um local.

## Fluxo Básico

- fb1: O Usuário *Administrador* inicia o caso de uso
- fb2: O sistema exibe todos os dispositivos cadastrados
- fb3: O Usuário *Administrador* realiza o cadastro de dispositivo
- fb4: O Usuário *Administrador* realiza a atualização dos dados de dispositivo
- fb5: O Usuário *Administrador* realiza a remoção dos dados de dispositivo
- fb6: O caso de uso é encerrado

## Fluxos Alternativos

- fa1: Efetuar Cadastro – No fluxo básico **fb3**, caso um dispositivo já seja cadastrado, o sistema cancela o cadastro. Retornar ao fluxo **fb6**.

### 3.10.1.5 Caso de Uso: Gerenciar Local

Este caso de uso representa a funcionalidade responsável pelo cadastro de locais, ambientes, também se vê responsável pela alteração de informações e remoção do mesmo.

## Fluxo Básico

- fb1: O Usuário *Administrador* inicia o caso de uso
- fb2: O sistema exibe todos os locais cadastrados
- fb3: O Usuário *Administrador* realiza o cadastro de local
- fb4: O Usuário *Administrador* realiza a atualização dos dados de local
- fb5: O Usuário *Administrador* realiza a remoção dos dados de local
- fb6: O caso de uso é encerrado

## Fluxos Alternativos

- fa1: Efetuar Cadastro – No fluxo básico **fb3**, caso um local já seja cadastrado, o sistema cancela o cadastro. Retornar ao fluxo **fb6**.
- fa2: Remover Dados – No fluxo básico **fb5**, caso o local já cadastrado tenha uma relação com algum dispositivo atribuído à ela, a remoção é cancelada, retornar ao fluxo **fb6**.

### **3.10.1.6 Caso de Uso: Gerenciar Função**

Este caso de uso representa a funcionalidade responsável por cadastrar a função do usuário dentro da instituição, alterar informações e efetuar a remoção da função.

#### **Fluxo Básico**

- fb1: O Usuário *Administrador* inicia o caso de uso
- fb2: O sistema exibe todas as funções cadastradas
- fb3: O Usuário *Administrador* realiza o cadastro de função
- fb4: O Usuário *Administrador* realiza a atualização dos dados de função
- fb5: O Usuário *Administrador* realiza a remoção dos dados de função
- fb6: O caso de uso é encerrado

#### **Fluxos Alternativos**

- fa1: Efetuar Cadastro – No fluxo básico **fb3**, caso uma função já seja cadastrada, o sistema cancela o cadastro. Retornar ao fluxo **fb6**.
- fa2: Remover Dados – No fluxo básico **fb5**, caso a função já cadastrada tenha uma relação com algum usuário atribuído à ela, a remoção é cancelada, retornar ao fluxo **fb6**.

### **3.10.1.7 Caso de Uso: Gerenciar Departamento**

Este caso de uso representa a funcionalidade responsável por cadastrar o departamento da instituição, alterar informações e remoção de departamentos.

#### **Fluxo Básico**

- fb1: O Usuário *Administrador* inicia o caso de uso
- fb2: O sistema exibirá todos os departamentos cadastrados
- fb3: O Usuário *Administrador* realiza o cadastro de departamento
- fb4: O Usuário *Administrador* realiza a atualização dos dados de departamento
- fb5: O Usuário *Administrador* realiza a remoção dos dados de departamento
- fb6: O caso de uso é encerrado

## Fluxos Alternativos

fa1: Efetuar Cadastro – No fluxo básico **fb3**, caso um dispositivo já seja cadastrado, o sistema cancela o cadastro. Retornar ao fluxo **fb6**.

fa2: Remover Dados – No fluxo básico **fb5**, caso o departamento já cadastrado tenha uma relação com algum usuário atribuído à ele, a remoção é cancelada, retornar ao fluxo **fb6**.

### 3.10.1.8 Caso de Uso: Gerenciar Automóvel

Este caso de uso representa a funcionalidade responsável pelo cadastro de automóvel, também é responsável por relacionar o automóvel a seu respectivo proprietário, responsável também por alterar registros do automóvel e também a remoção do mesmo.

#### Fluxo Básico

fb1: O Usuário *Administrador* inicia o caso de uso

fb2: O sistema exibirá todos os automóveis cadastrados

fb3: O Usuário *Administrador* realiza o cadastro de automóvel

fb4: O Usuário *Administrador* realiza a atualização dos dados de automóvel

fb5: O Usuário *Administrador* realiza a remoção dos dados de automóvel

fb6: O caso de uso é encerrado

#### Fluxos Alternativos

fa1: Efetuar Cadastro – No fluxo básico **fb3**, caso um automóvel já seja cadastrado, o sistema cancela o cadastro. Retornar ao fluxo **fb6**.

fa2: Remover Dados – No fluxo básico **fb5**, caso o automóvel já cadastrado tenha uma relação com algum usuário atribuído à ele, a remoção é cancelada, retornar ao fluxo **fb6**.

### 3.10.1.9 Caso de Uso: Gerenciar Tag RFID

Este caso de uso representa a funcionalidade responsável pelo cadastro de etiquetas RFID, o caso de uso também é responsável somente pela remoção da etiqueta RFID não possibilitando a alteração do mesmo.

## Fluxo Básico

- fb1: O Usuário *Administrador* inicia o caso de uso
- fb2: O sistema exibirá todos as etiquetas cadastradas
- fb3: O Usuário *Administrador* realiza o cadastro de etiqueta
- fb4: O Usuário *Administrador* realiza a remoção dos dados de etiqueta
- fb5: O caso de uso é encerrado

## Fluxos Alternativos

- fa1: Efetuar Cadastro – No fluxo básico **fb3**, caso uma etiqueta já seja cadastrada, o sistema cancela o cadastro. Retornar ao fluxo **fb5**.
- fa2: Remover Dados – No fluxo básico **fb5**, caso a etiqueta já cadastrada tenha uma relação com algum usuário atribuído à ela, a remoção é cancelada, retornar ao fluxo **fb5**.

### 3.10.1.10 Caso de Uso: Consultar Log de Registro

Este caso de uso representa a funcionalidade responsável pela consulta de Logs de registros a ambientes previamente definidos.

## Fluxo Básico

- fb1: O Usuário *Administrador* inicia o caso de uso
- fb2: O sistema exibirá todos os registros do log
- fb3: O caso de uso é encerrado

## 3.11 ANÁLISE E DESIGN

O objetivo deste tópico é designar e orientar a arquitetura do sistema proposto e prover a partir de uma análise mais específica e apurada dos requisitos coletados responder todas as necessidades de implantação por meio de diagramas e suas descrições.

## 3.12 MODELAGEM DE BANCO DE DADOS

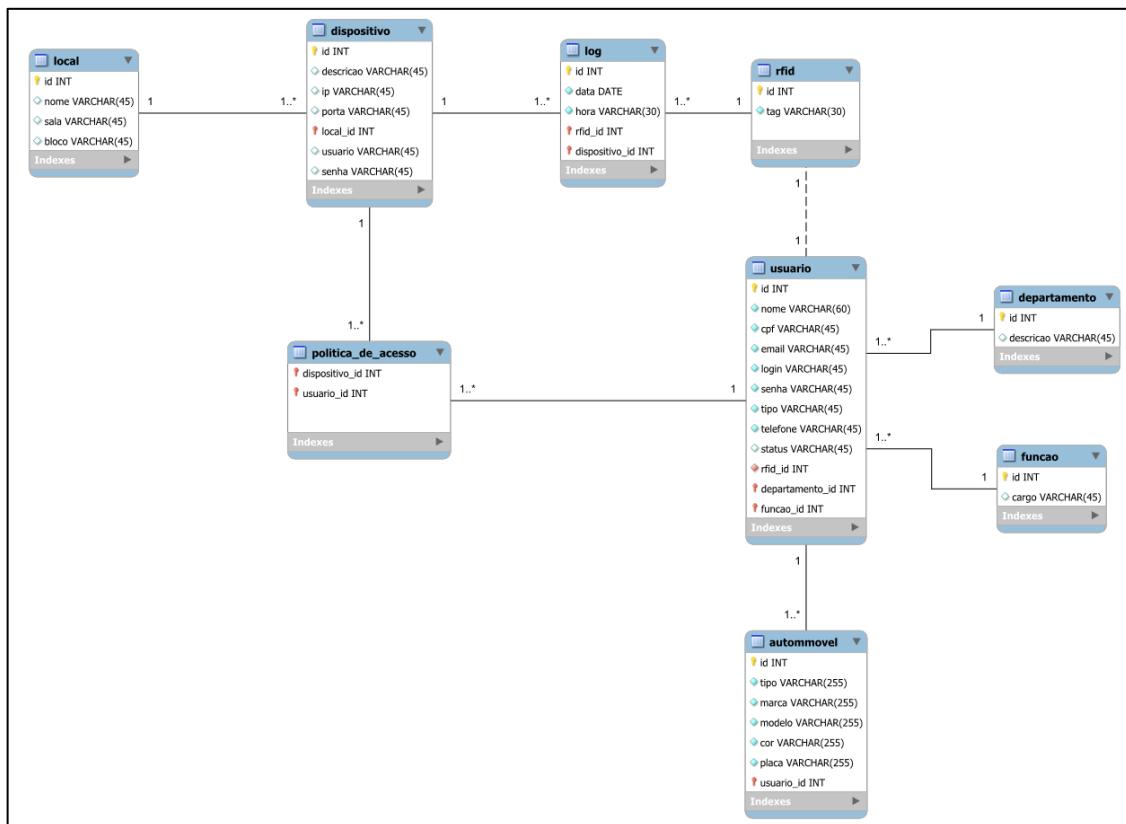
Segundo MEDEIROS (2004) Banco de Dados se define em uma coleção de dados inter-relacionados e armazenados de forma independente das aplicações que

o utilizam. Um banco de dados pode ter várias entidades, cada uma guarda informações de um determinado assunto, as entidades são divididas em linhas ou registros e estes divididos em colunas chamadas de campos. Cada campo trata um lado diferente, cada registro trata uma ocorrência para o assunto que aquela tabela cuida.

O banco de dados e sua tecnologia tem um impacto importante sobre o uso crescente dos computadores. É correto afirmar que os banco de dados desempenham um papel crítico em quase todas as áreas em que os computadores são usados incluindo negócios, comércio eletrônico, engenharia dentre outros. Os sistemas de bancos de dados mais populares de hoje são bancos de dados relacionais quase universalmente os bancos de dados relacionais usam uma linguagem chamada *Structured Query Language (SQL)* – Linguagem de consulta estruturada – para realizar consultas, isto é, solicitar informações que satisfaçam critérios e manipular dados. (DEITEL, 2010).

Ao desenho de diagramas, que mostra o relacionamento entre entidades para obtermos uma informação, chamamos de Modelo de Entidade e Relacionamento ou por sua abreviação MER. MEDEIROS (2004).

Figura 3.2 - Diagrama de Entidade e Relacionamento



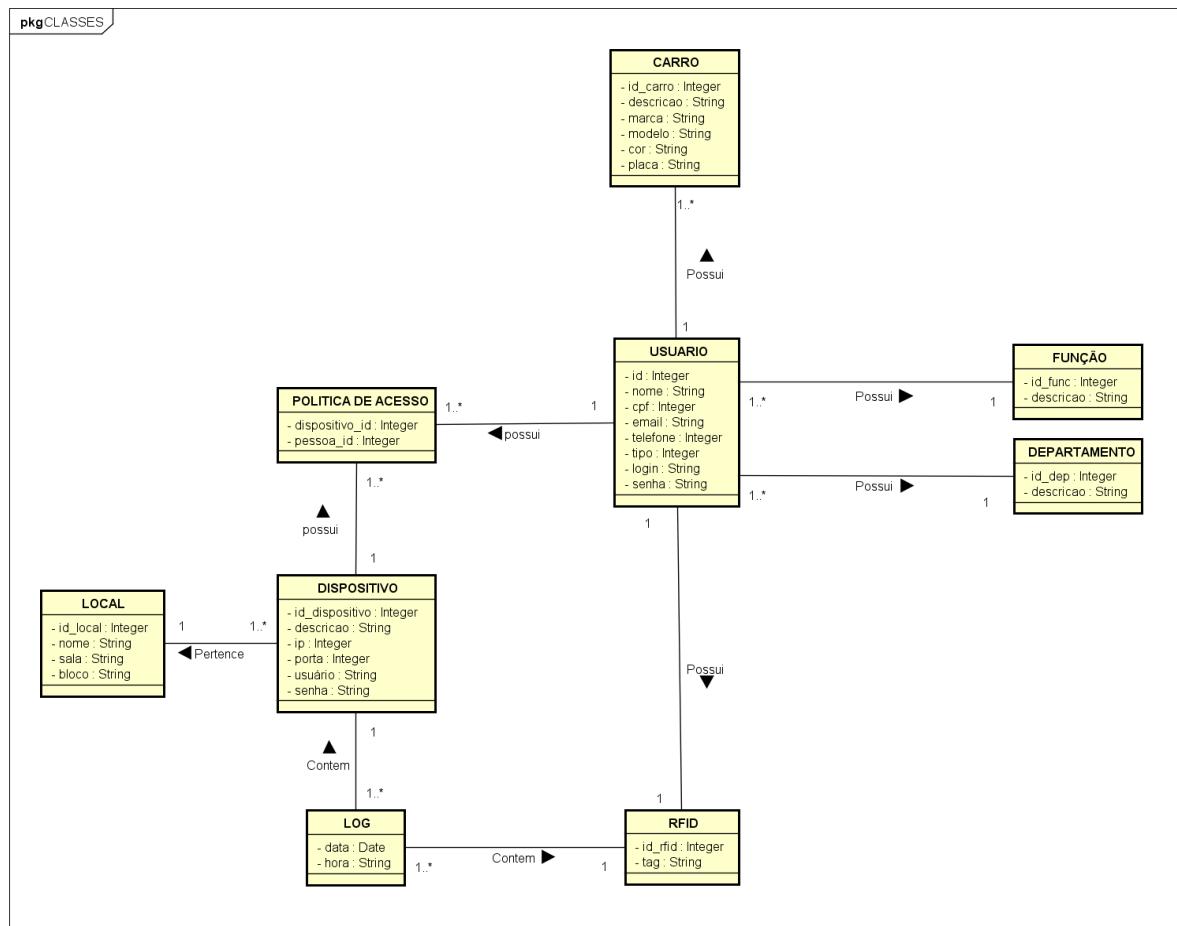
Fonte: Elaborado pelos autores, 2016.

### 3.13 DIAGRAMA DE CLASSE

De acordo com GUEDES (2014), Diagrama de Classes é um dos mais utilizados e importantes da UML, servindo de apoio para a maioria dos outros diagramas. O diagrama de classes define a estrutura das classes utilizadas pelo sistema, determinando os atributos e métodos possuídos por cada classe, além de estabelecer como as classes se relacionam e trocam informações entre si.

Segundo MEDEIROS (2004) Fowler e Scot acertadamente indicam três momentos distintos quando estamos fazendo os diagramas de classes. Esses momentos são: Conceitual, Especificação e Implementação. No panorama conceitual, resolvemos o negócio; na especificação, tratamos as classes já como softwares, indicando tipos de retorno, parâmetros de entrada e saída, e tudo que envolva suas interfaces expostas ao mundo; no panorama de implementação, restamos colocar as classes em pacotes e códigos nestas.

Figura 3.3 - Diagrama de Classes



Fonte: Elaborado pelos autores, 2016.

### 3.14 MODELO VISÃO E CONTROLE E OBJETOS DE ACESSO A DADOS

MVC é uma estratégia para o projeto de interfaces com usuário orientadas a objetos. O padrão desacopla a UI do sistema, dividindo o projeto da UI em três partes separadas, o modelo que representa o sistema, o modo de visualização que exibe o modelo e o controlador que processa as entradas dos usuários especificamente cada parte do MVC tem seu conjunto próprio de responsabilidades exclusivas (SINTES, 2002).

Conforme BAPTISTELLA (2009) por essência o MVC separa em camadas distintas responsabilidades pertinentes a soluções de um software, essas partes se colaboram entre si mesmo estando divididas logicamente. Basicamente, o controlador gerencia o fluxo da aplicação, os dados do modelo somente são transportados à visão e vice-versa por intermédio do controlador.

De acordo com GONÇALVES (2007), MVC é um conceito de desenvolvimento e design que tenta separar uma aplicação em três partes distintas, o mesmo é uma forma de desenvolvimento que ajuda na manutenção do sistema, um padrão muito aceito no desenvolvimento de aplicações Java, principalmente de aplicações escritas para a Web. A separação lógica da aplicação nestas partes assegura que a camada Modelo não sabe nada praticamente do que é exibido, restringindo por representar as partes de componentes do problema que é resolvido pela aplicação. Igualmente, a camada de Apresentação só está relacionada a exibir os dados e não com implementar lógica de negócios que é controlada pela camada Modelo. O Controlador, como um gerenciador de tráfego, dirige as apresentações a serem exibidas e com as devidas mudanças de dados e recuperações vindas da camada Modelo.

Integrado com o *Data Access Object* (DAO), são modernos sistemas que acessam dados utilizam esse padrão de desenvolvimento como base para a manipulação com o banco de dados. GONÇALVES (2007)

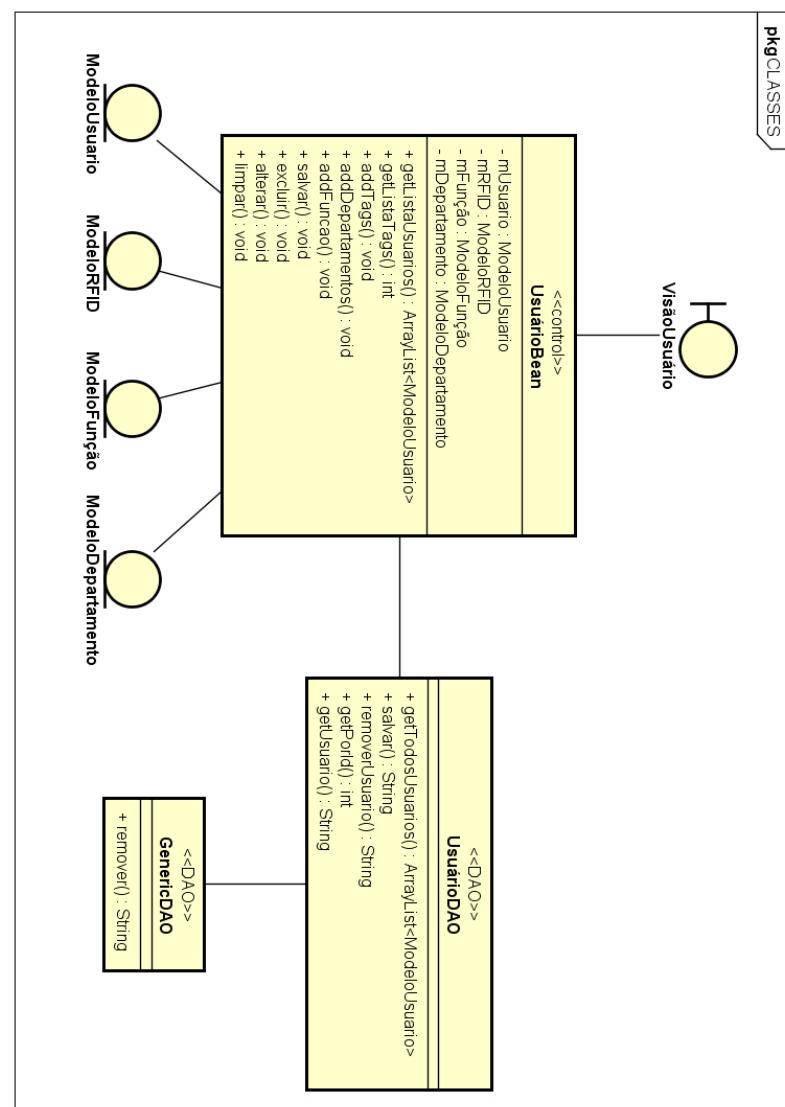
O Padrão DAO é o padrão mais utilizado para acesso de dados contidos em banco de dados, sempre que você precisa acessar um banco de dados que está mantendo seu modelo de objetos é melhor empregar o padrão DAO, o mesmo fornece uma interface independente, no qual você pode usar para persistir objetos de dados. A ideia é colocar todas as funcionalidades encontradas no desenvolvimento de acesso e trabalho com dados em um só local, tornando simples

sua manutenção. O DAO inclui métodos para inserir, selecionar, atualizar e excluir objetos de um banco de dados.

Nas figuras a seguir deste tópico é demonstrado a relação entre o controle, a visão, o modelo e banco referente a cada ação do sistema.

MVC – Usuário: Gerenciamento de operações pelo controle entre a visão do usuário e os modelos relacionados.

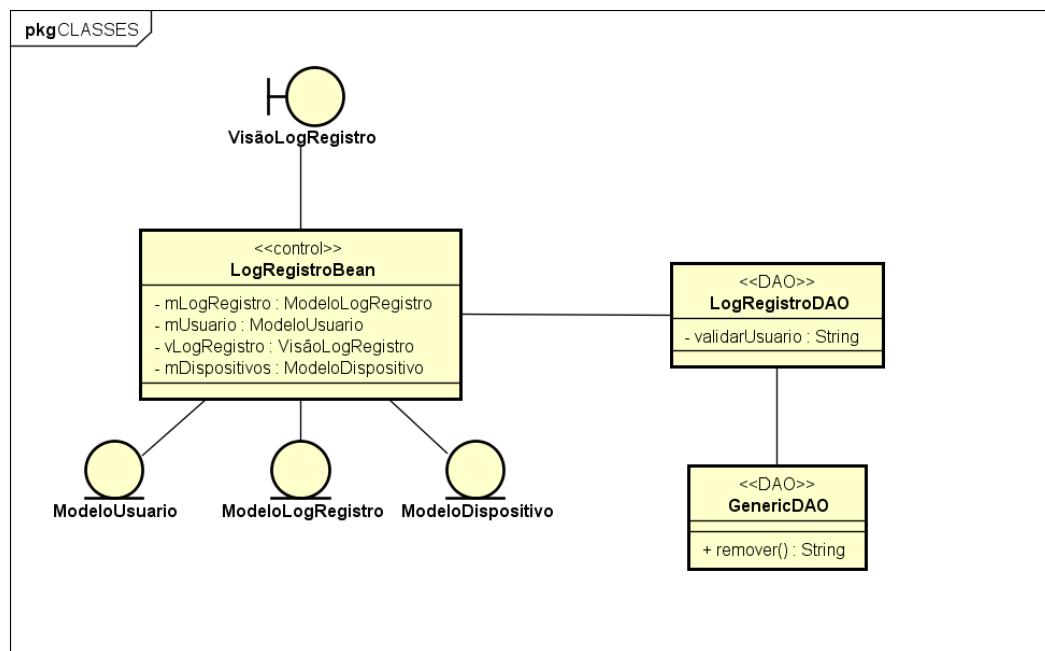
Figura 3.4 - Diagrama de MVC Gerenciar Usuário



Fonte: Elaborado pelos autores, 2016.

MVC – Log de Registro: Gerenciamento de operações pelo controle entre a visão do usuário e os modelos relacionados.

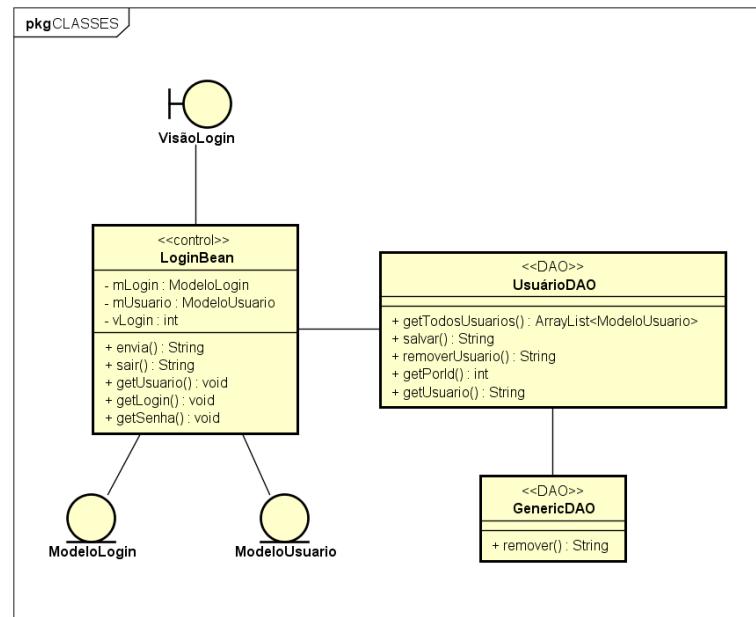
Figura 3.5 - Diagrama de MVC Log de Registro



Fonte: Elaborado pelos autores, 2016.

MVC – *Login*: Gerenciamento de operações pelo controle entre a visão do usuário e os modelos relacionados.

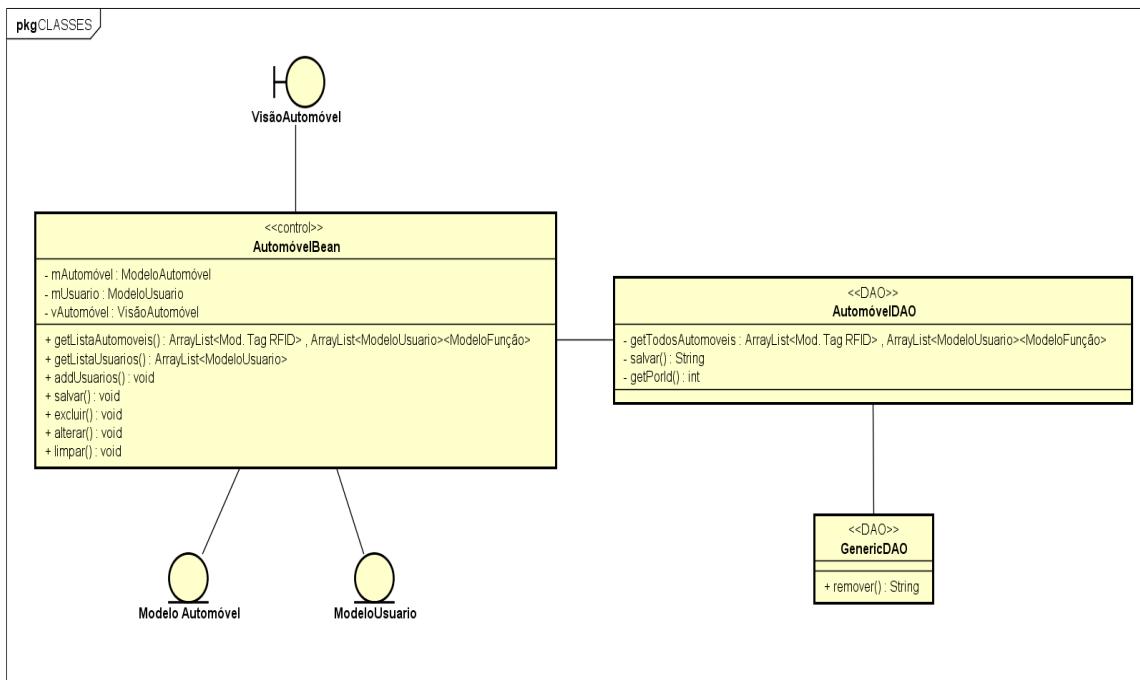
Figura 3.6 - Diagrama de MVC Login



Fonte: Elaborado pelos autores, 2016.

MVC – *Login*: Gerenciamento de operações pelo controle entre a visão do usuário e os modelos relacionados.

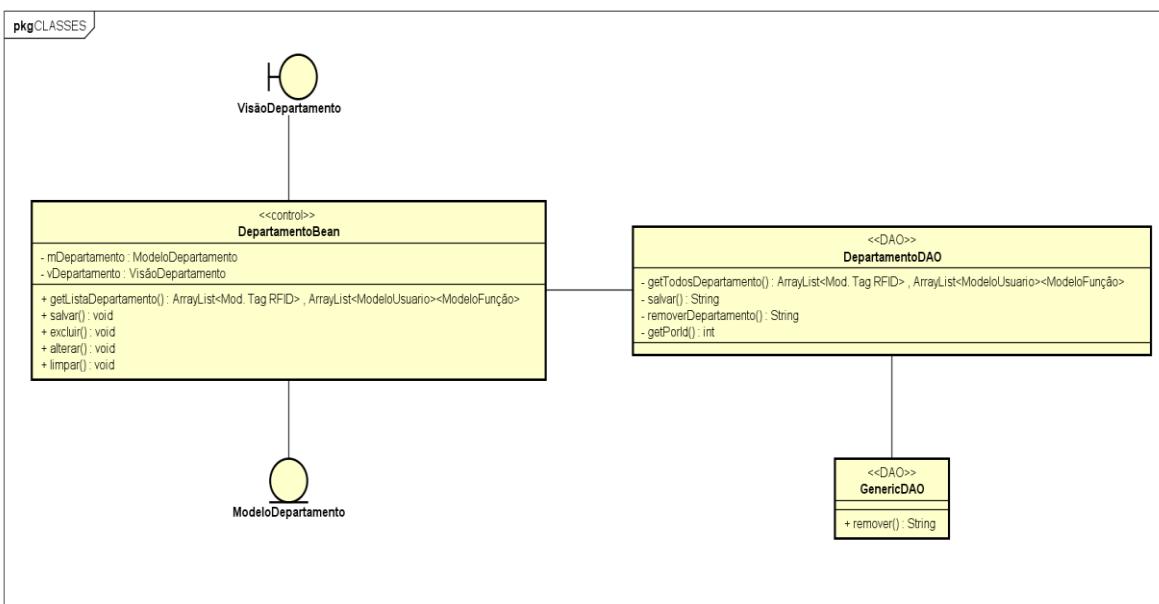
Figura 3.7 - Diagrama de MVC Gerenciar Automóvel



Fonte: Elaborado pelos autores, 2016.

MVC – Departamento: Gerenciamento de operações pelo controle entre a visão do usuário e os modelos relacionados.

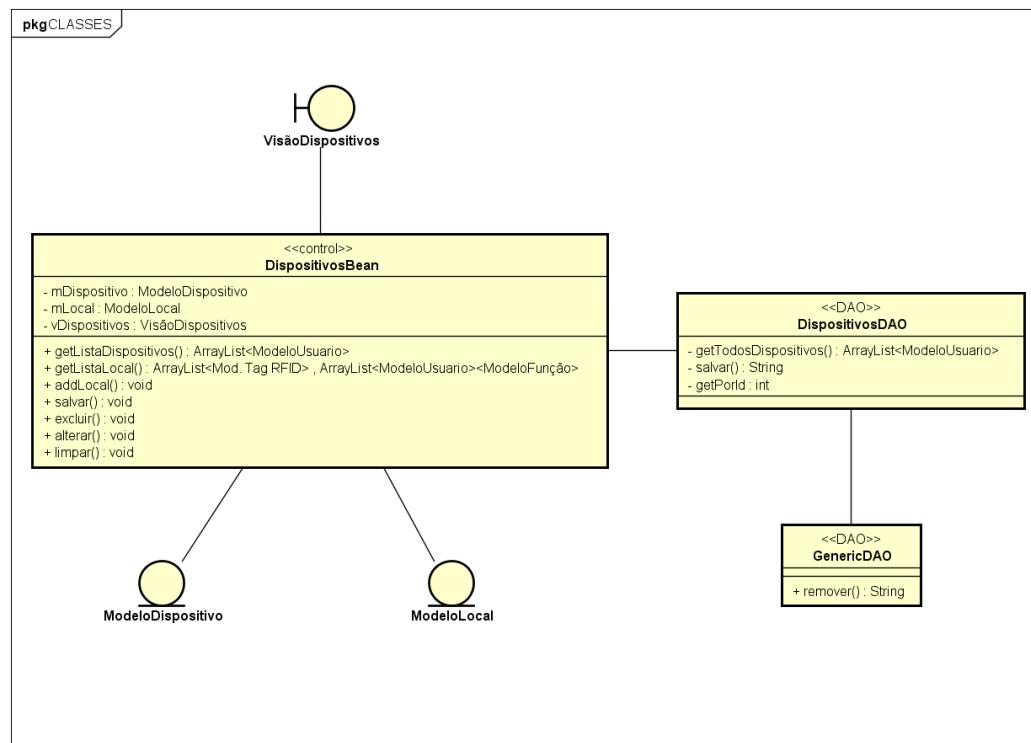
Figura 3.8 - Diagrama de MVC Gerenciar Departamento



Fonte: Elaborado pelos autores, 2016.

MVC – Dispositivo: Gerenciamento de operações pelo controle entre a visão do usuário e os modelos relacionados.

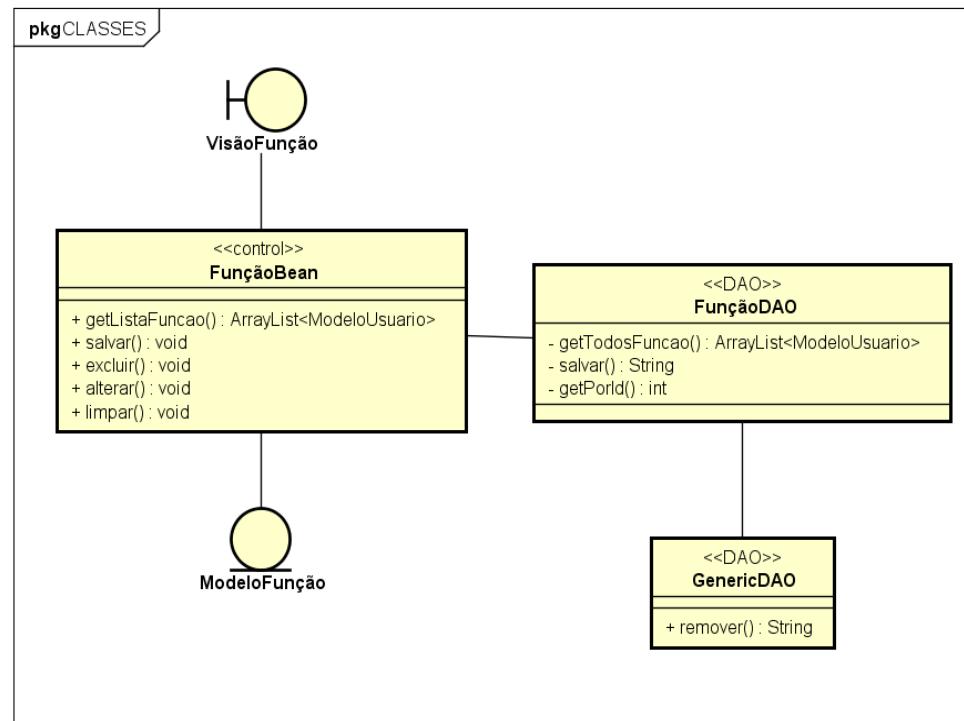
Figura 3.9 - Diagrama de MVC Gerenciar Dispositivo



Fonte: Elaborado pelos autores, 2016.

MVC – Função: Gerenciamento de operações pelo controle entre a visão do usuário e os modelos relacionados.

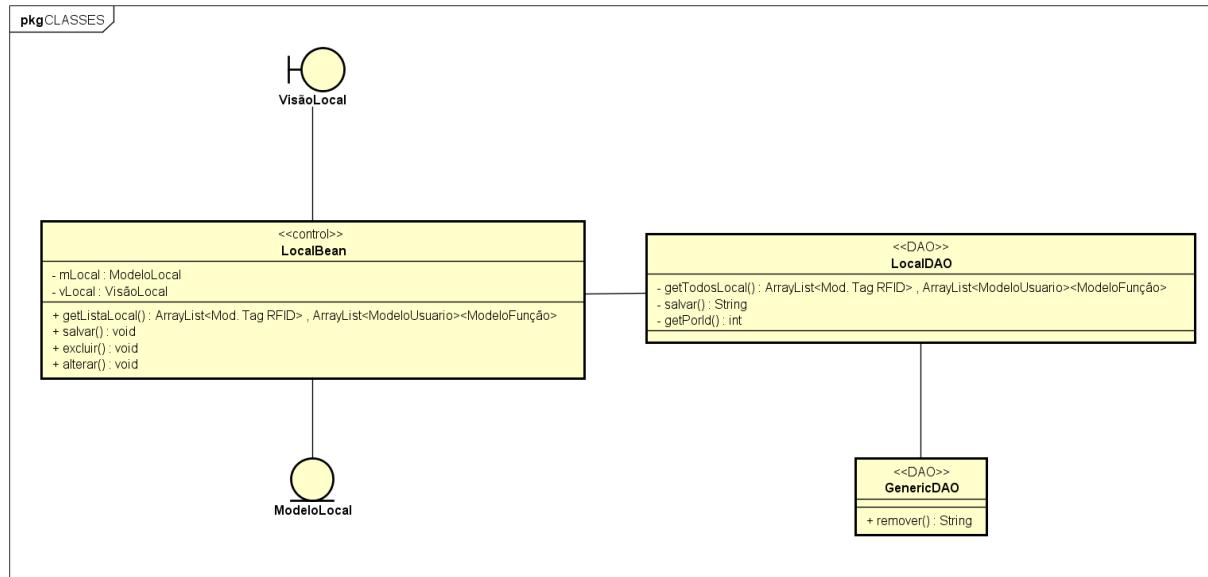
Figura 3.10 - Diagrama de MVC Gerenciar Função



Fonte: Elaborado pelos autores, 2016.

MVC – Local: Gerenciamento de operações pelo controle entre a visão do usuário e os modelos relacionados.

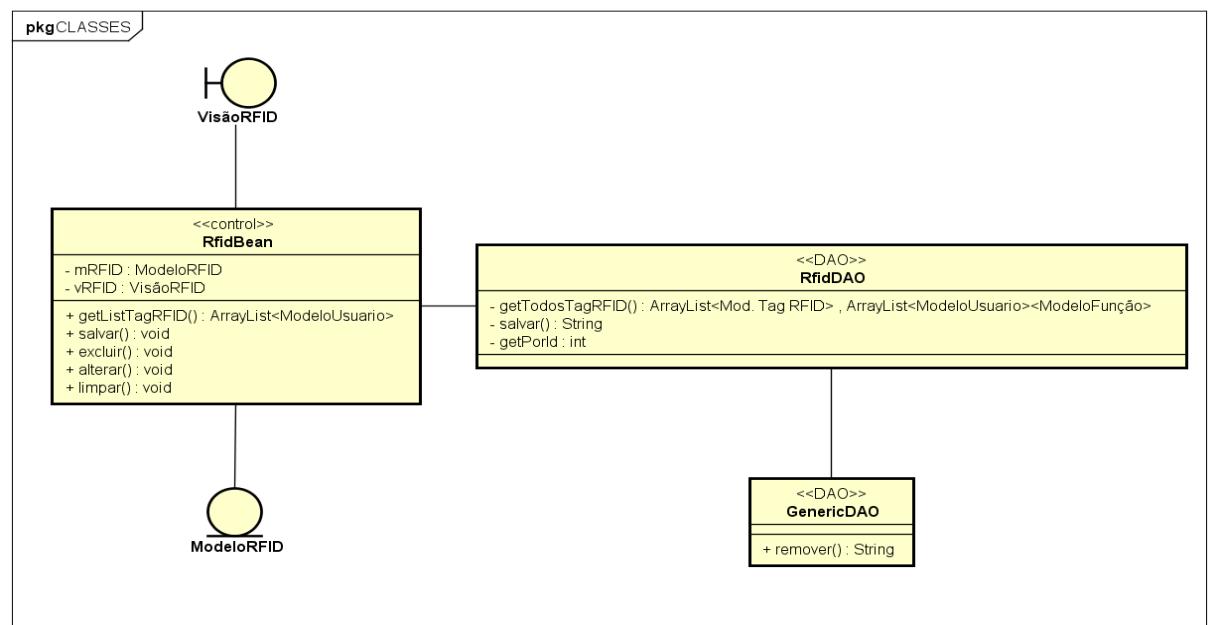
Figura 3.11 - Diagrama de MVC Gerenciar Local



Fonte: Elaborado pelos autores, 2016.

MVC – Tag: Gerenciamento de operações pelo controle entre a visão do usuário e os modelos relacionados.

Figura 3.12 - Diagrama de MVC Gerenciar Tag RFID



Fonte: Elaborado pelos autores, 2016.

### 3.15 DIAGRAMAS DE ATIVIDADES

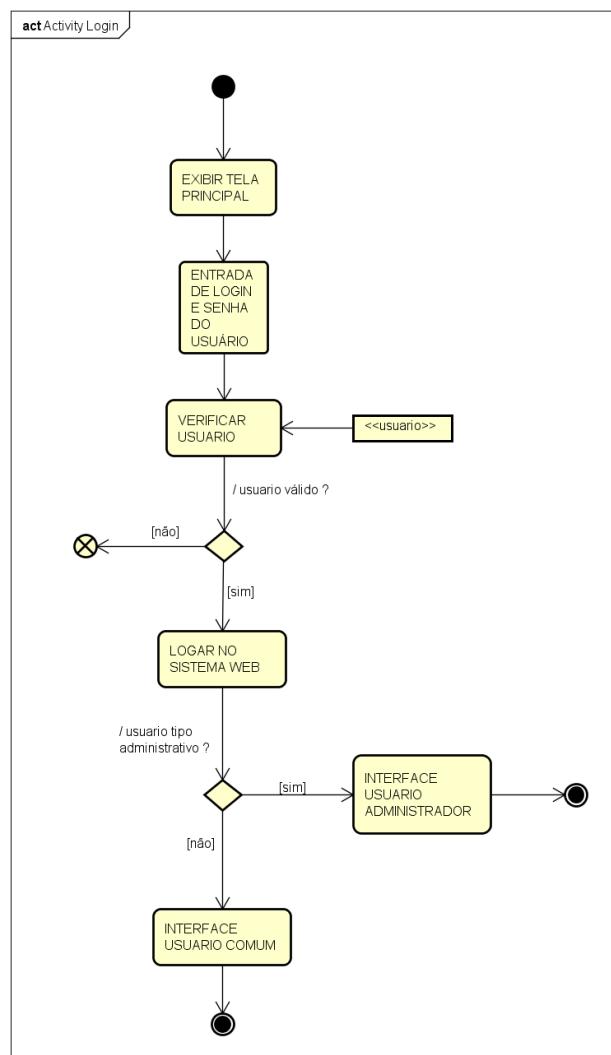
De acordo com Medeiros (2004) o Diagrama de Atividades demonstra o comportamento lógico da atividade e serve para ajudar a criar boas descrições de caso de uso, mostrando uma situação por meio de vários casos de uso ou ajustando dúvidas surgidas no Diagrama de Classes e objetos.

Geralmente é utilizado em cenários complexos, em que o negócio a ser modelado é de difícil compreensão (MEDEIROS, 2004)

#### 3.15.1 Efetuar *Login*

Fluxo de atividade para efetuar e validar o acesso de usuário ao sistema web.

Figura 3.13 - Diagrama de Atividades: *Login*

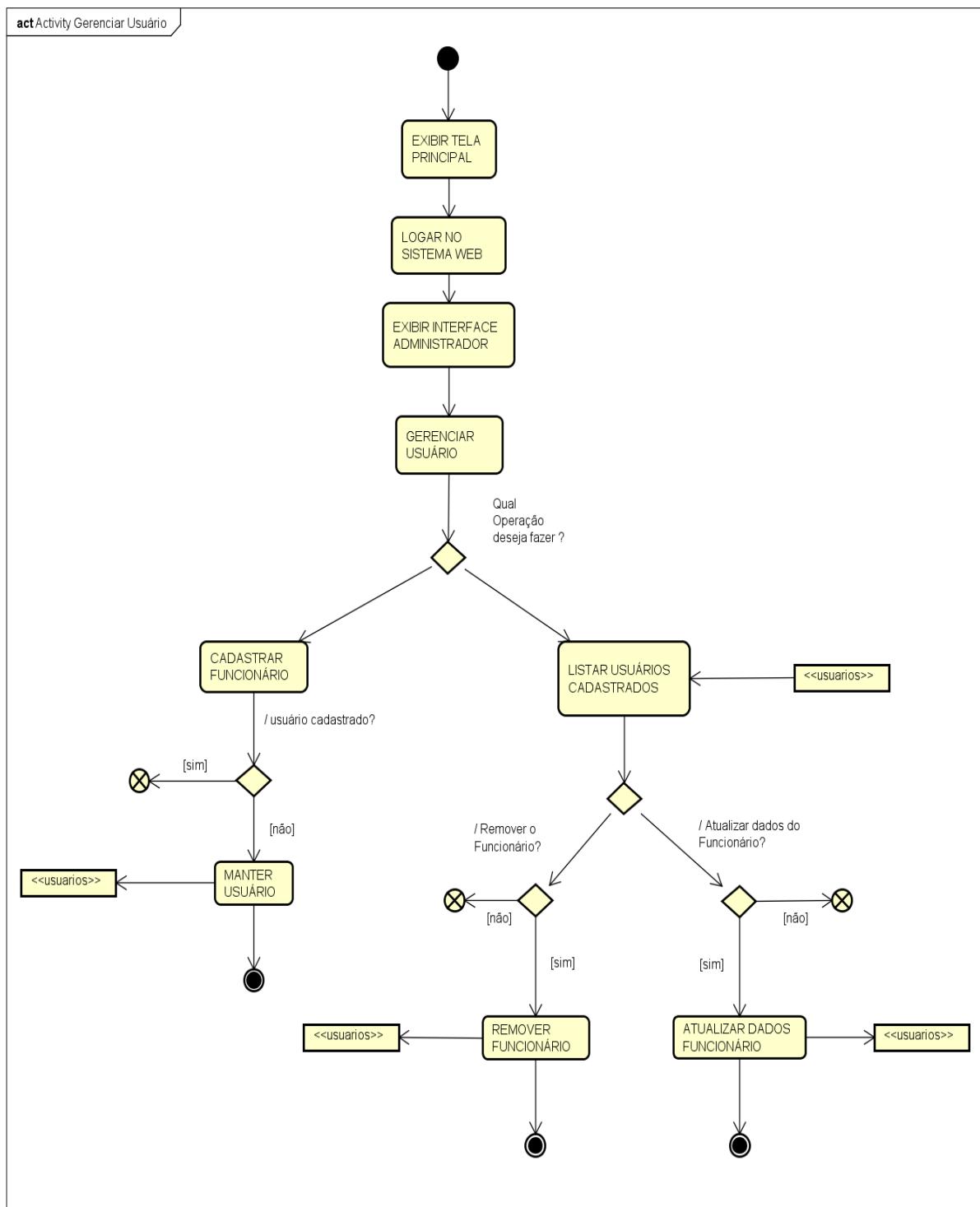


Fonte: Elaborado pelos autores, 2016.

### 3.15.2 Gerenciar Usuário

Fluxo de atividade para manter o usuário, apresentando os dados já inseridos no sistema, cadastrando, alterando e removendo dados.

Figura 3.14 - Diagrama de Atividades: Gerenciar Usuário

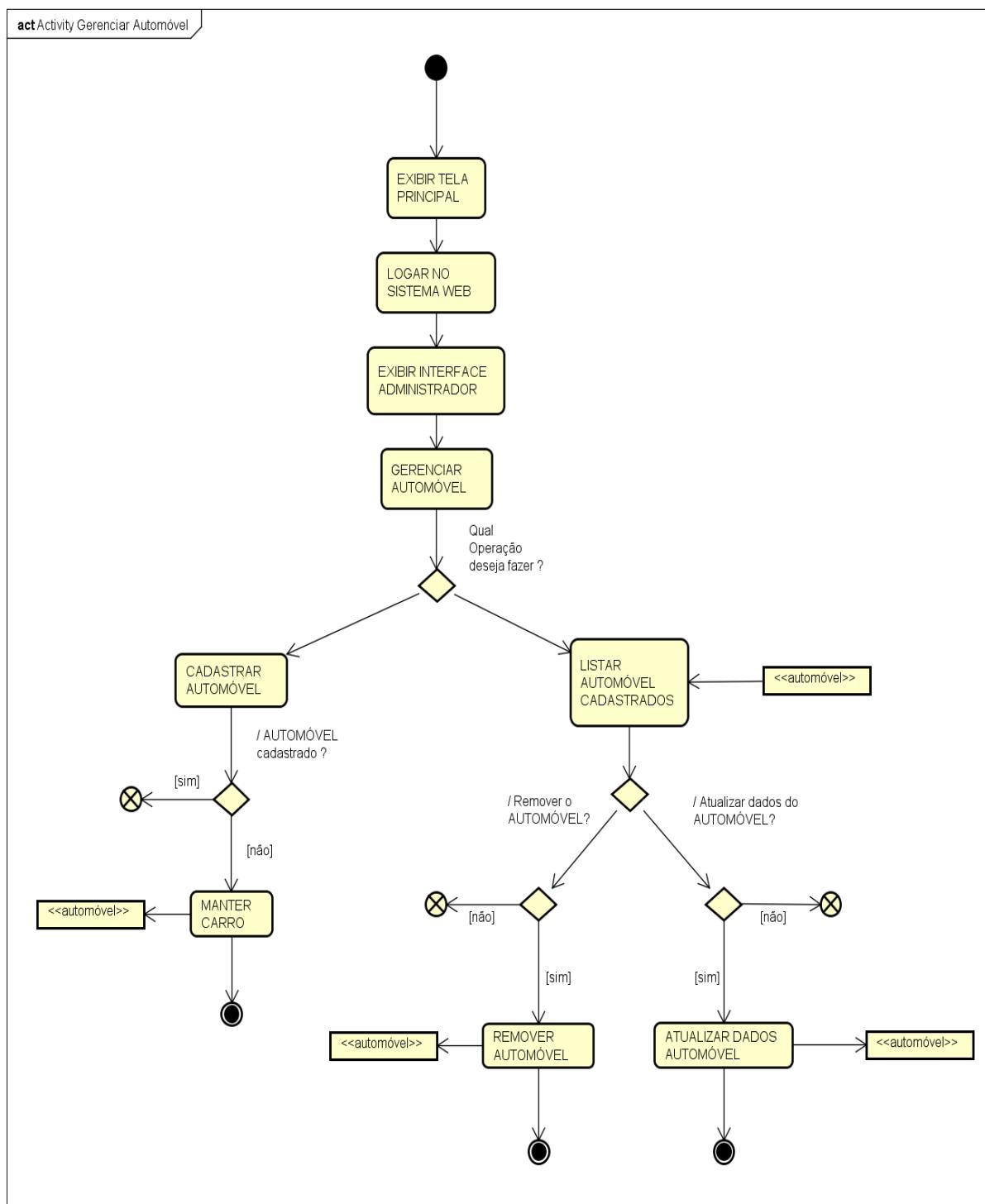


Fonte: Elaborado pelos autores, 2016.

### 3.15.3 Gerenciar Automóvel

Fluxo de atividade para manter o automóvel, apresentando os dados já inseridos no sistema, cadastrando, alterando e removendo dados.

Figura 3.15 - Diagrama de Atividades: Gerenciar Automóvel

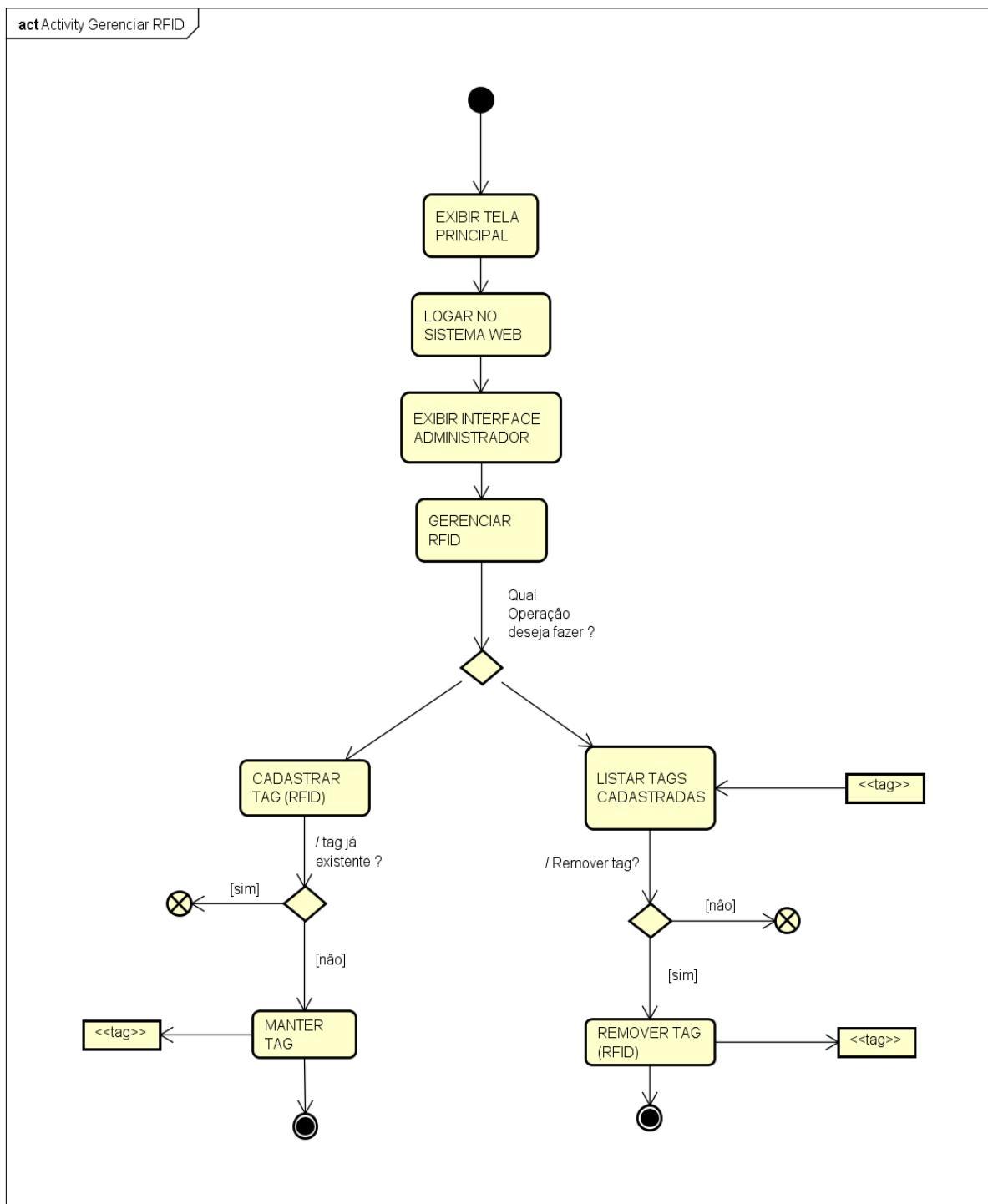


Fonte: Elaborado pelos autores, 2016.

### 3.15.4 Gerenciar RFID

Fluxo de atividade para manter a tag RFID, apresentando os dados já inseridos no sistema, cadastrando e removendo dados.

Figura 3.16 - Diagrama de Atividades: Gerenciar Etiqueta RFID

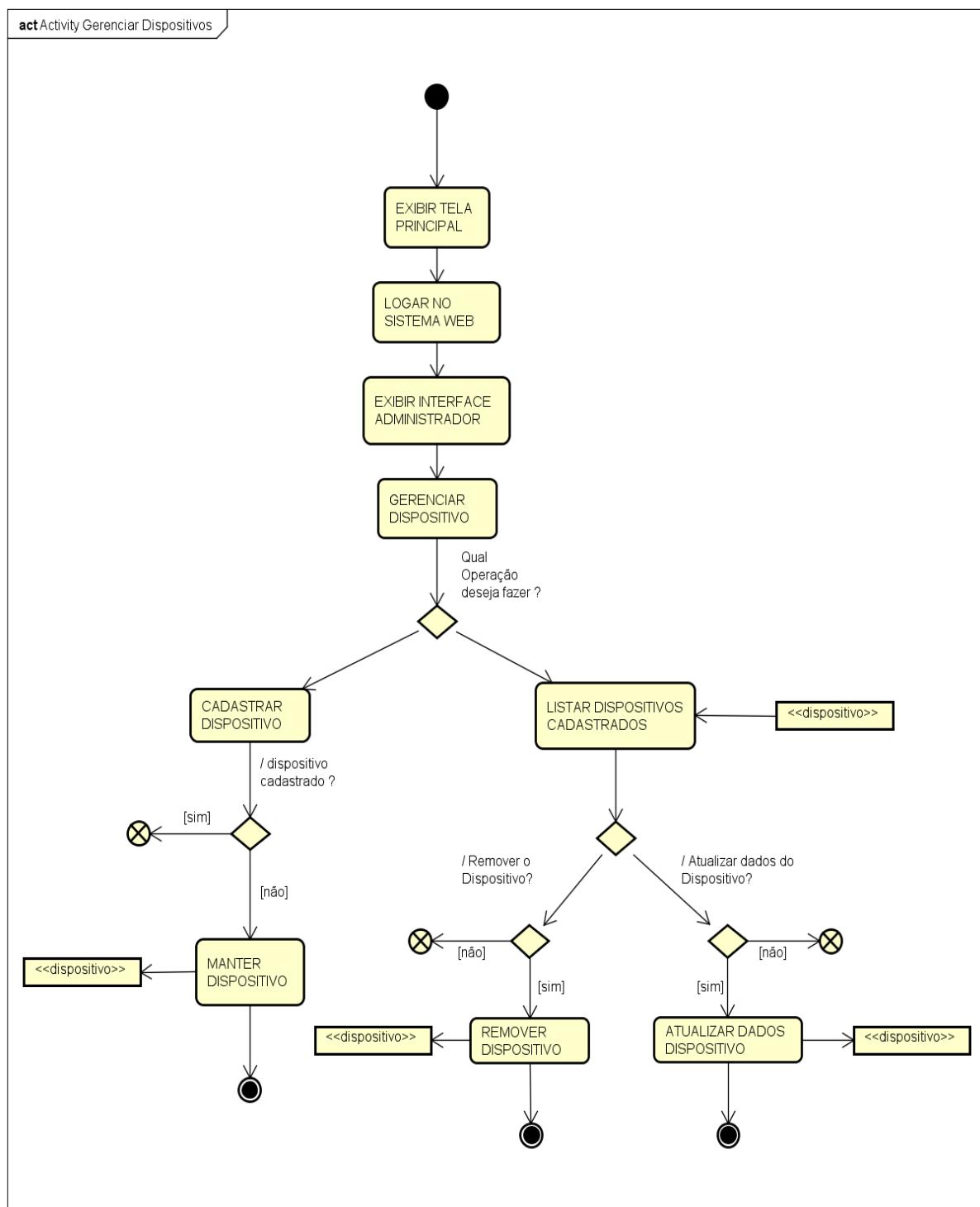


Fonte: Elaborado pelos autores, 2016.

### **3.15.5 Gerenciar Dispositivos**

Fluxo de atividade para manter o dispositivo, apresentando os dados já inseridos no sistema, cadastrando, alterando e removendo dados.

Figura 3.17 - Diagrama de Atividades: Gerenciar Dispositivos

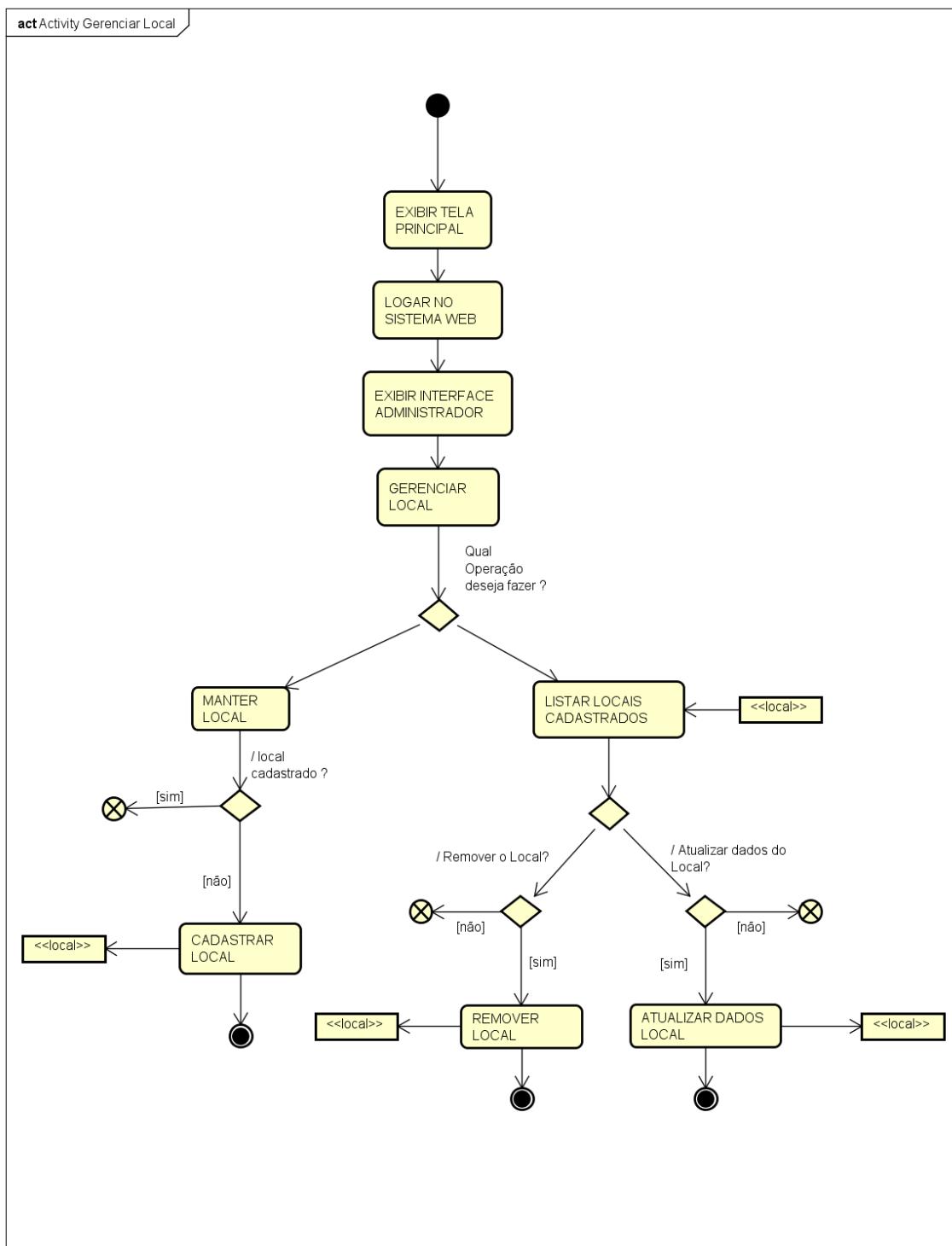


Fonte: Elaborado pelos autores, 2016.

### 3.15.6 Gerenciar Local

Fluxo de atividade para manter o local, apresentando os dados já inseridos no sistema, cadastrando, alterando e removendo dados.

Figura 3.18 - Diagrama de Atividades: Gerenciar Local

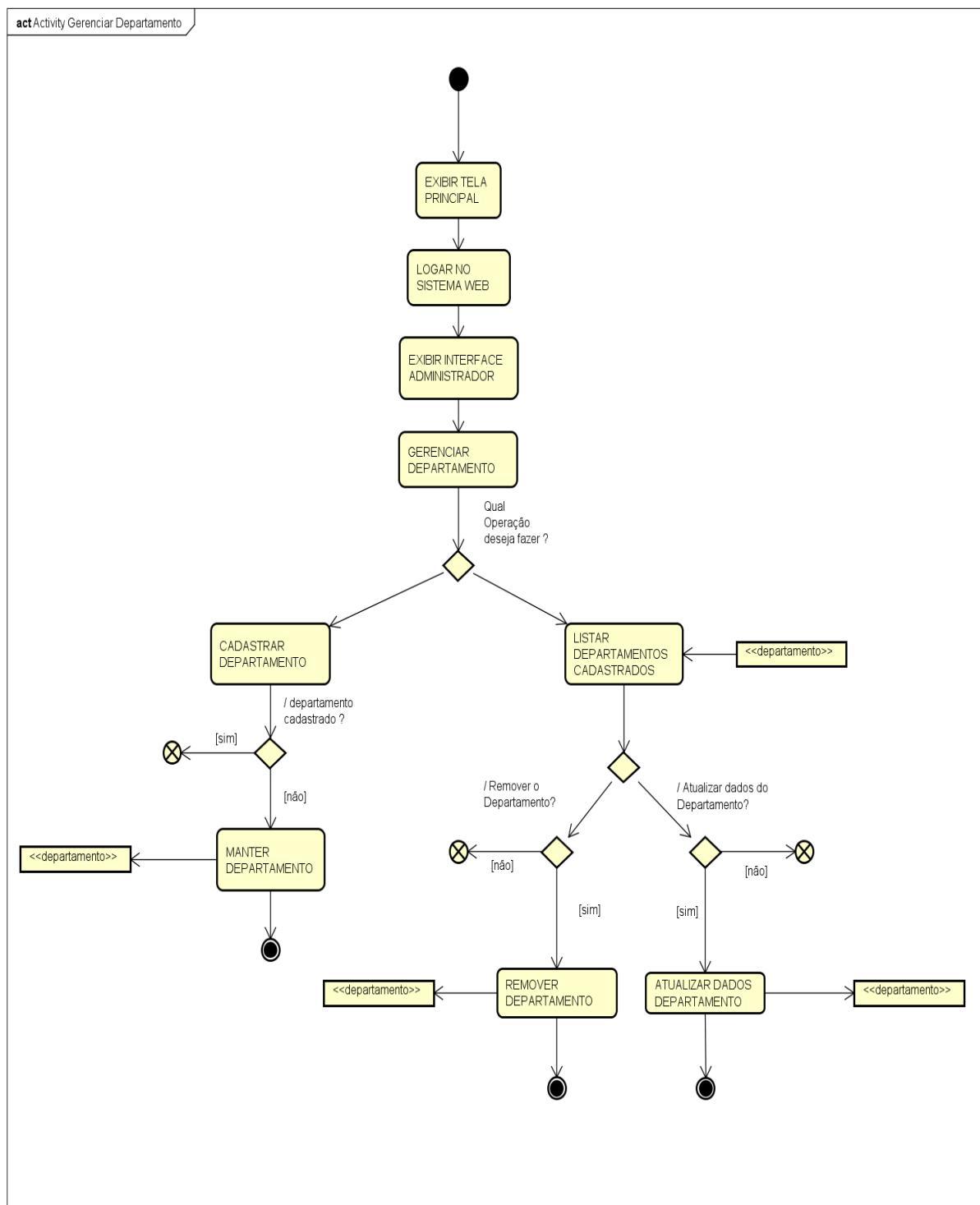


Fonte: Elaborado pelos autores, 2016.

### 3.15.7 Gerenciar Departamento

Fluxo de atividade para manter o departamento, apresentando os dados já inseridos no sistema, cadastrando, alterando e removendo dados.

Figura 3.19 - Diagrama de Atividades: Gerenciar Departamento

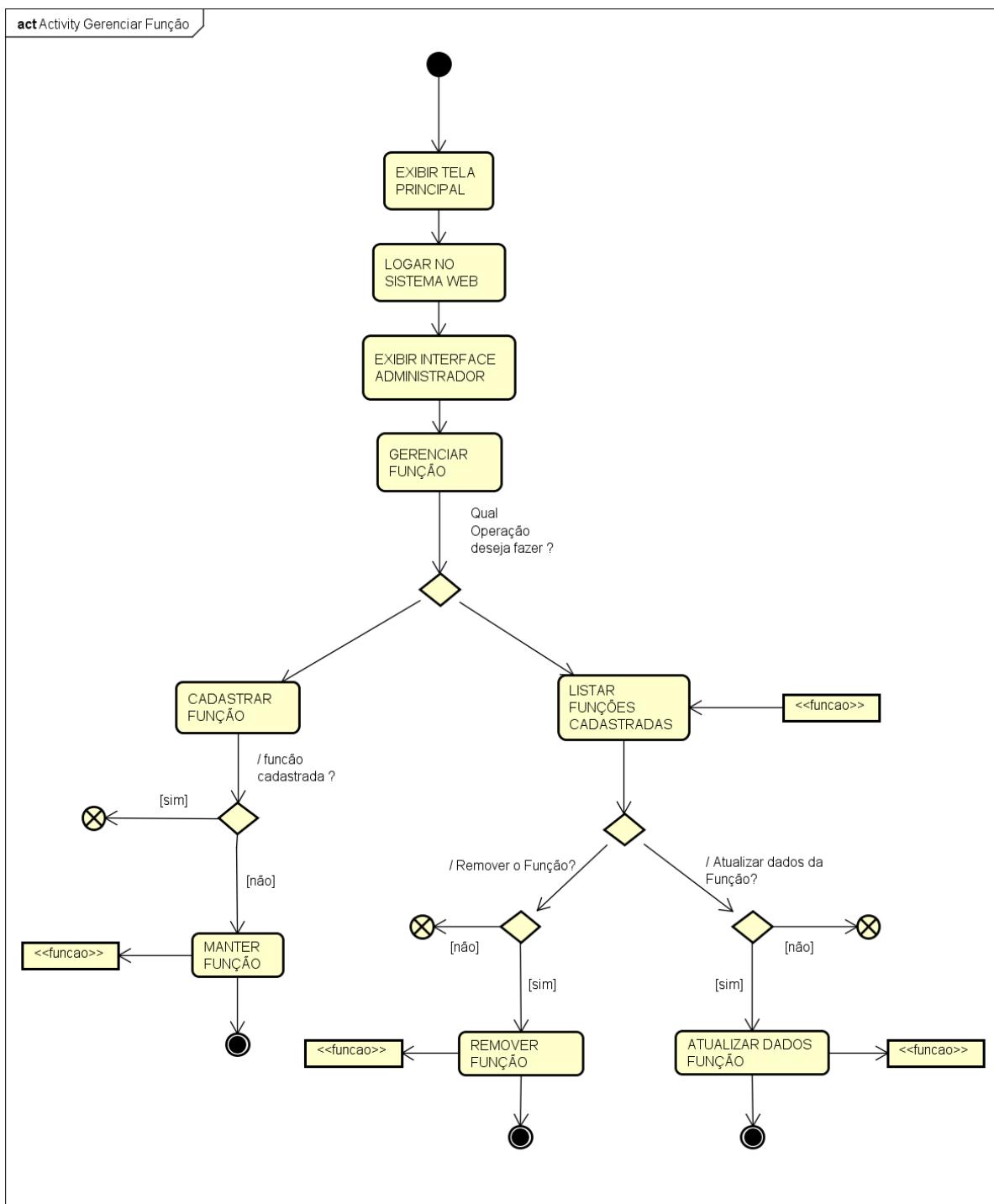


Fonte: Elaborado pelos autores, 2016.

### 3.15.8 Gerenciar Função

Fluxo de atividade para manter função, apresentando os dados já inseridos no sistema, cadastrando, alterando e removendo dados.

Figura 3.20 - Diagrama de Atividades: Gerenciar Função

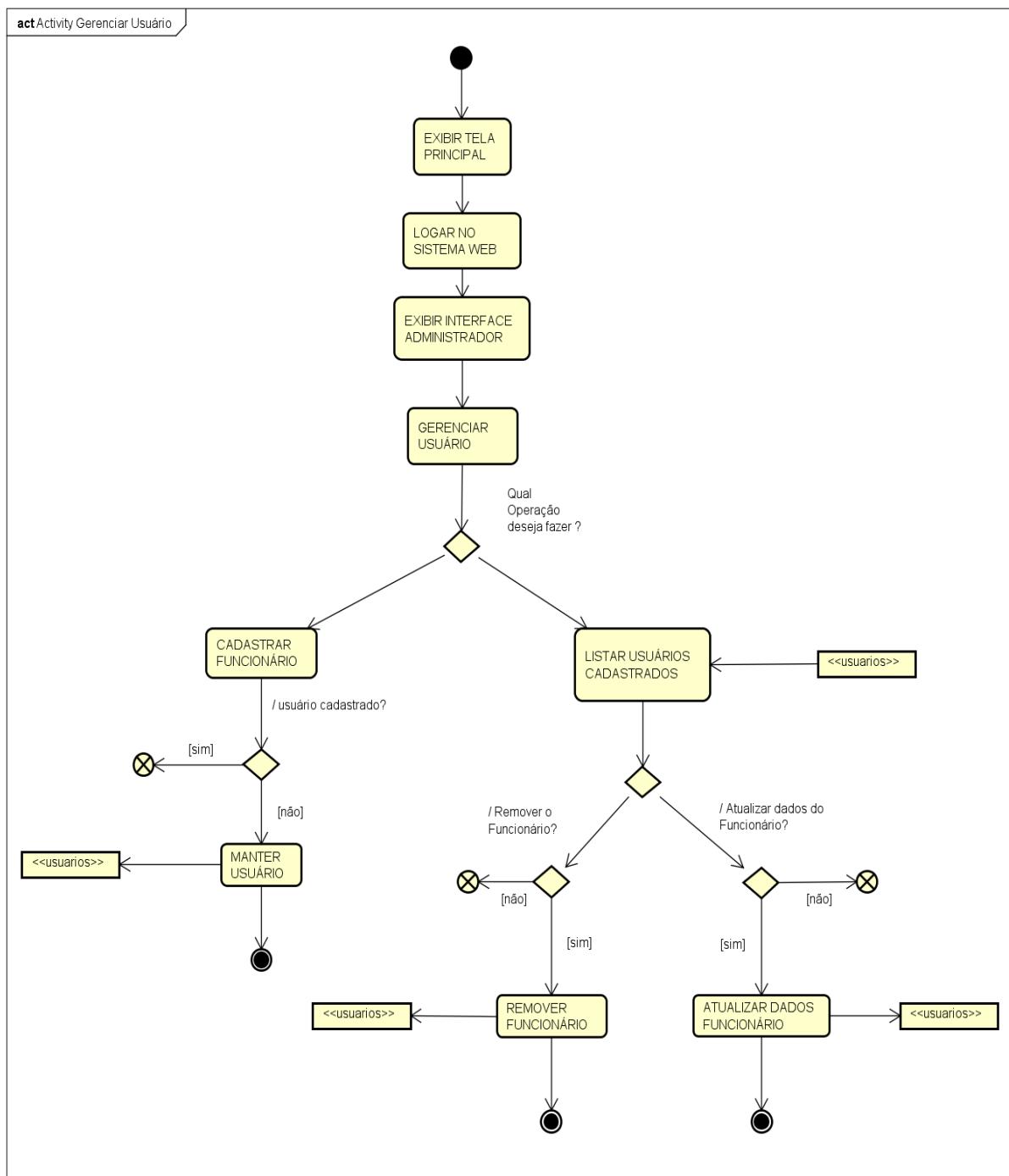


Fonte: Elaborado pelos autores, 2016.

### 3.15.9 Exibir Log de Registro

Fluxo de atividade para visualizar o log de registros, apresentando todos os dados de uso entre usuários e dispositivos.

Figura 3.21 - Diagrama de Atividades: Log de Registro Usuário Administrador

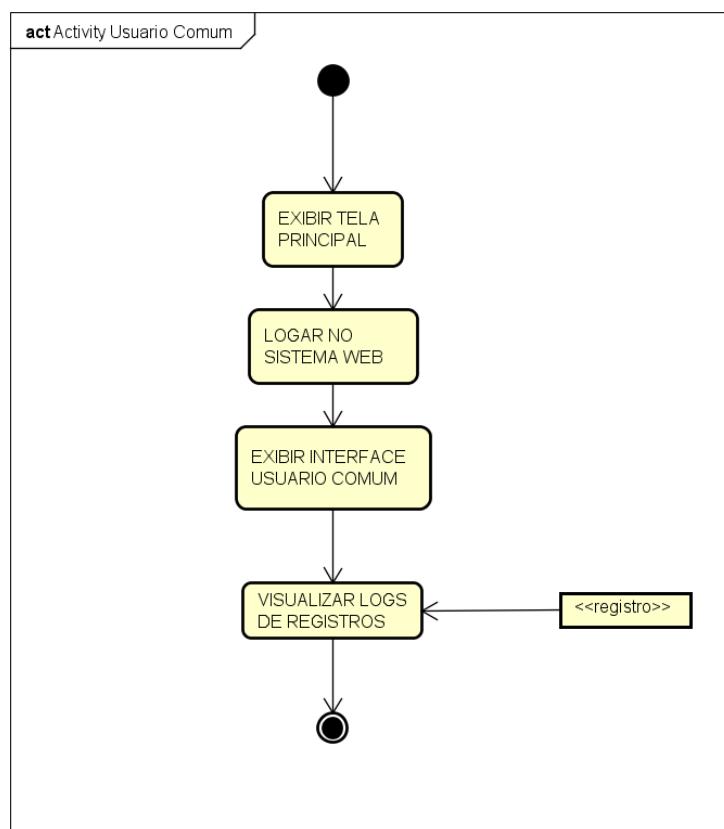


Fonte: Elaborado pelos autores, 2016.

### 3.15.10 Visualizar Log de Registros Usuário Comum

Fluxo de atividade para visualizar o log de registros, apresentando os dados limitados ao usuário *logado*.

Figura 3.22 - Diagrama de Atividades: Usuário Comum



powered by Astah

Fonte: Elaborado pelos autores, 2016.

## 3.16 DIAGRAMAS DE SEQUENCIA

De acordo com Medeiros (2004) o Diagrama de Sequência pode ser usado para mostrar a evolução de uma dada situação, em determinado momento do software, mostrar uma dada colaboração entre duas ou mais classes e pode, também, ser usado para mostrar a tradução de um Caso de Uso desde a interação com o usuário até a finalização do processo.

O Diagrama de Sequência exibe a ordem temporal em que as interações são realizadas entre os objetos envolvidos em um determinado processo, o mesmo é baseado em um diagrama de Caso de Uso e apoia-se no Diagrama de Classes para determinar os objetos das classes envolvidas em um processo, bem como os métodos disparados entre os mesmos. Um Diagrama de Sequência costuma

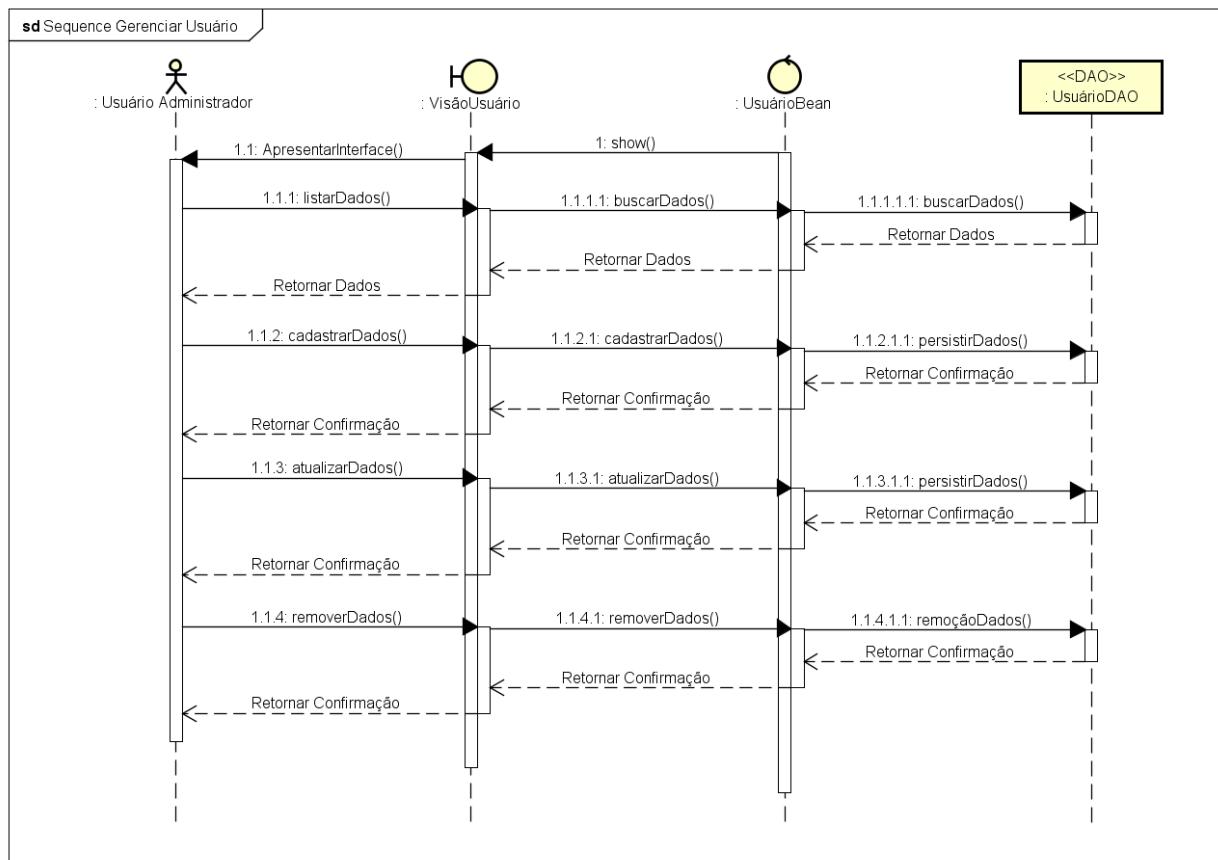
identificar o evento gerador do processo modelado, bem como o ator responsável por este evento, e determina como o processo deve se desenrolar e ser concluído por meio do envio de mensagens que em geral disparam método entre os objetos. (GUEDES, 2014).

Ele melhora o diagrama de classes, permitindo que acrescentemos ou retiremos métodos e/ou atributos desnecessários de um conjunto de classes. Serve como norte para o programador e dá-nos maior segurança de que o diagrama de classes atende às necessidades do negócio. (MEDEIROS, 2004).

### 3.16.1 Gerenciar Usuário

Na figura 3.23 é demonstrado o gerenciamento de usuário, o processo de listagem de dados já cadastrados, cadastrar, alterar e remover dados.

Figura 3.23 - Diagrama de Sequência: Gerenciar Usuário

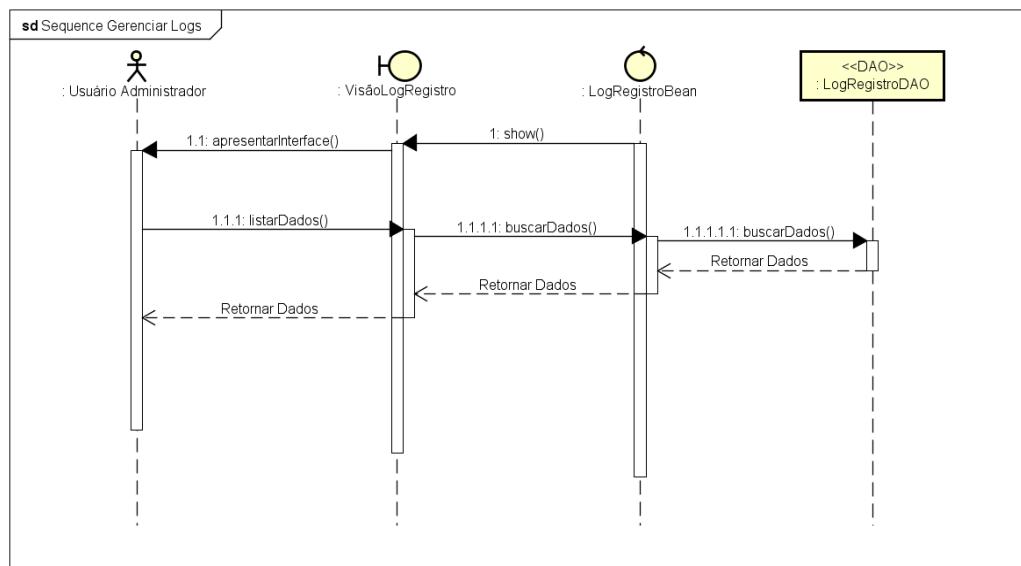


Fonte: Elaborado pelos autores, 2016.

### 3.16.2 Gerenciar Log de Registro

Na figura 3.24 é demonstrado o log de registros, ou seja, os acessos que são registrados.

Figura 3.24 - Diagrama de Sequência: Gerenciar Logs de Registro

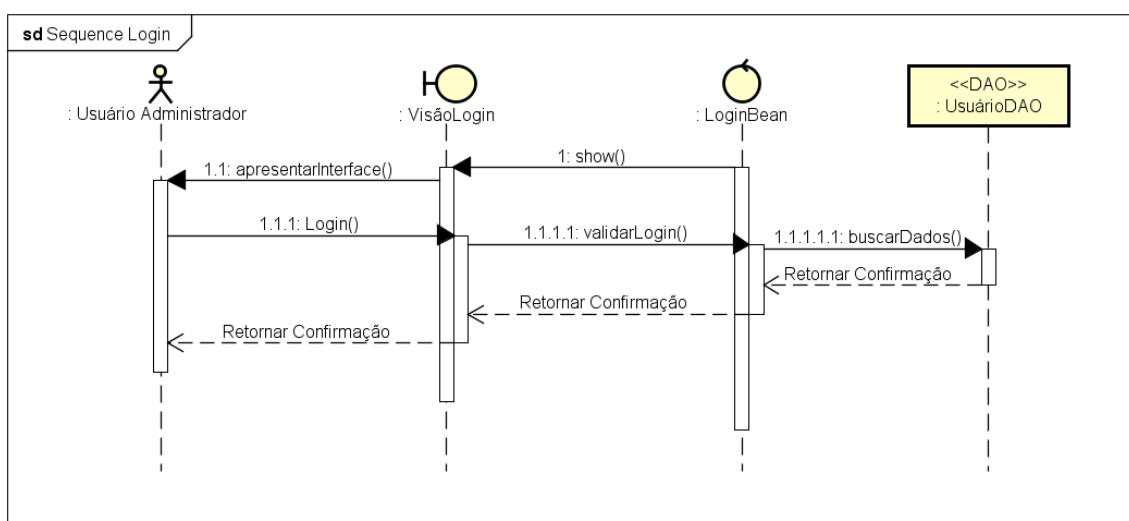


Fonte: Elaborado pelos autores, 2016.

### 3.16.3 Fazer Login

Na figura 3.25 é demonstrado o processo para realizar o *login*.

Figura 3.25 - Diagrama de Sequência: Fazer Login

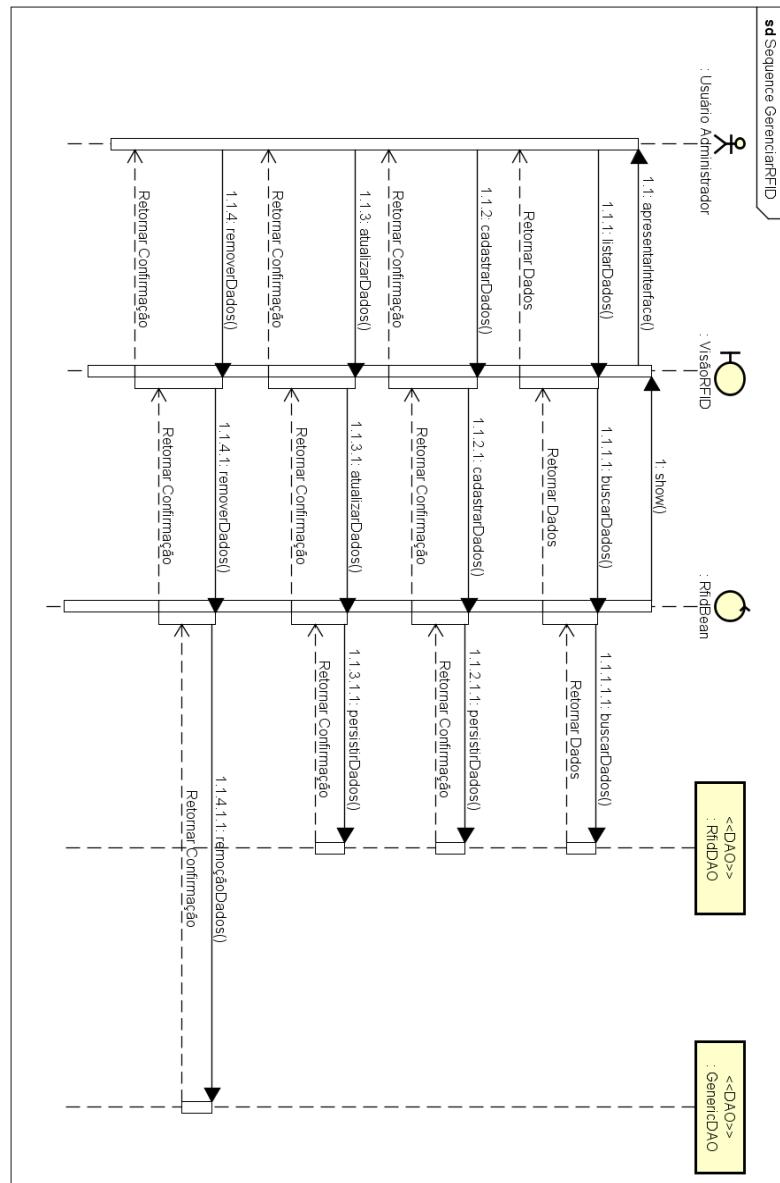


Fonte: Elaborado pelos autores, 2016.

### 3.16.4 Gerenciar Tag RFID

Na figura 3.26 é demonstrado o gerenciamento de usuário, o processo de listagem de dados já cadastrados, cadastrar, alterar e remover dados.

Figura 3.26 - Diagrama de Sequência: Gerenciar RFID

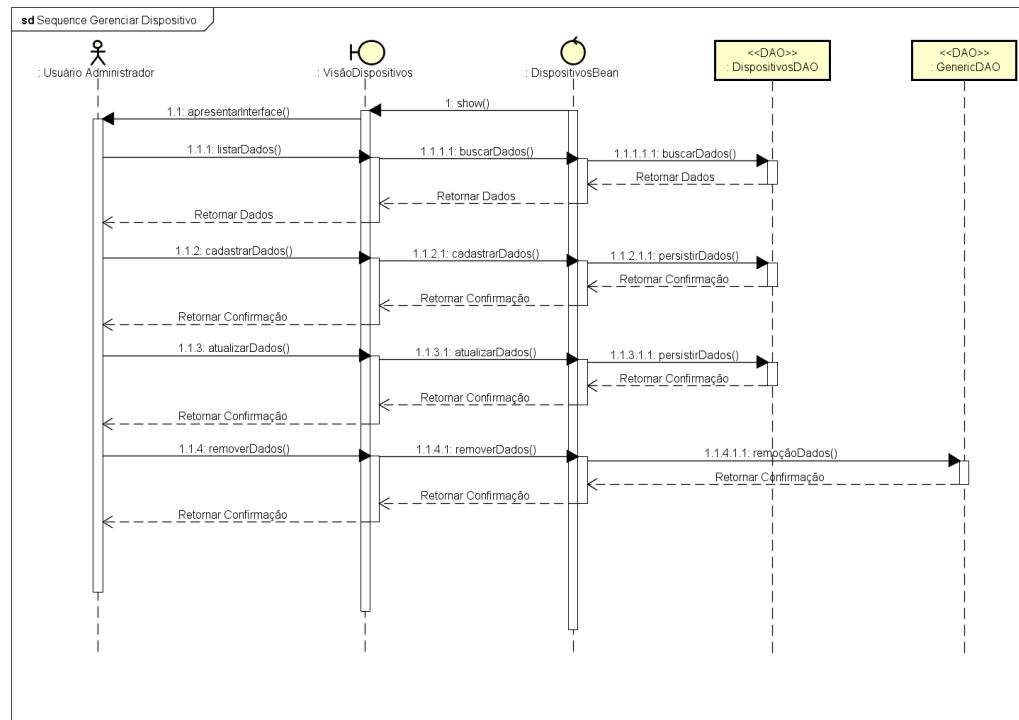


Fonte: Elaborado pelos autores, 2016.

### 3.16.5 Gerenciar Dispositivos

Na figura 3.27 é demonstrado o gerenciamento de usuário, o processo de listagem de dados já cadastrados, cadastrar, alterar e remover dados.

**Figura 3.27 - Diagrama de Sequência: Gerenciar Dispositivos**

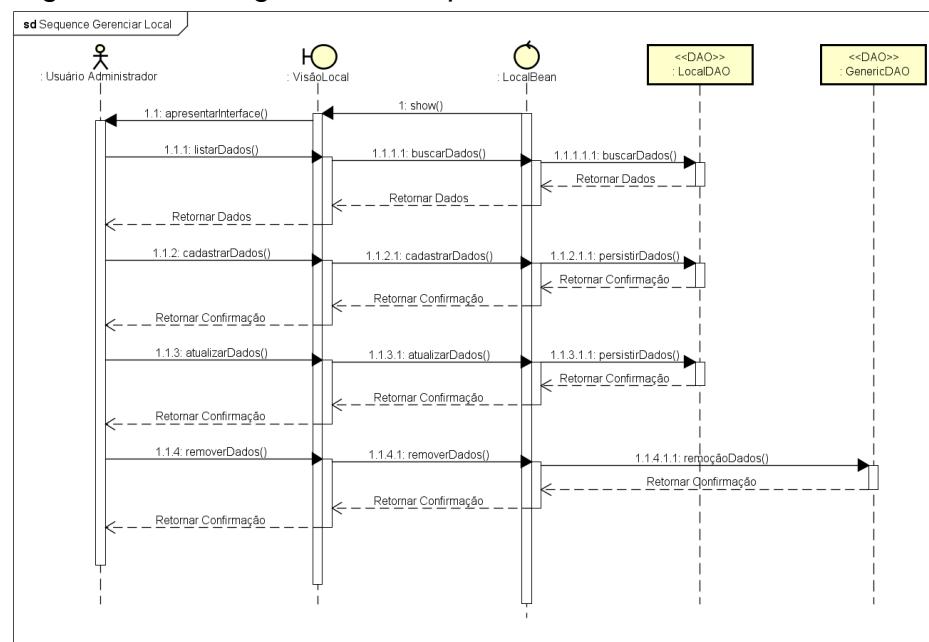


Fonte: Elaborado pelos autores, 2016.

### 3.16.6 Gerenciar Local

Na figura 3.28 é demonstrado o gerenciamento de usuário, o processo de listagem de dados já cadastrados, cadastrar, alterar e remover dados.

**Figura 3.28 - Diagrama de Sequência: Gerenciar Local**

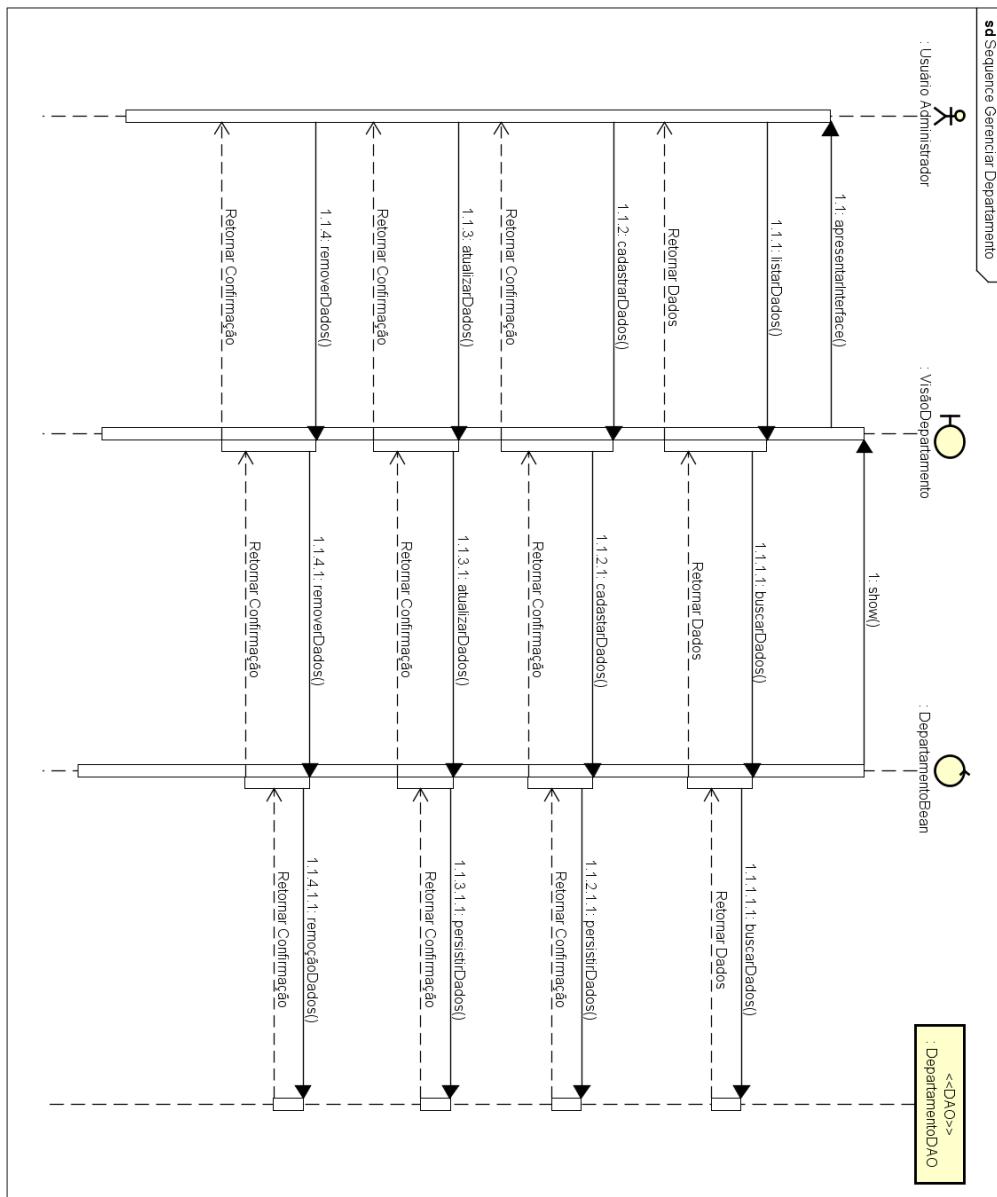


Fonte: Elaborado pelos autores, 2016.

### **3.16.7 Gerenciar Departamento**

Na figura 3.29 é demonstrado o gerenciamento de usuário, o processo de listagem de dados já cadastrados, cadastrar, alterar e remover dados.

Figura 3.29 - Diagrama de Sequência: Gerenciar Departamento

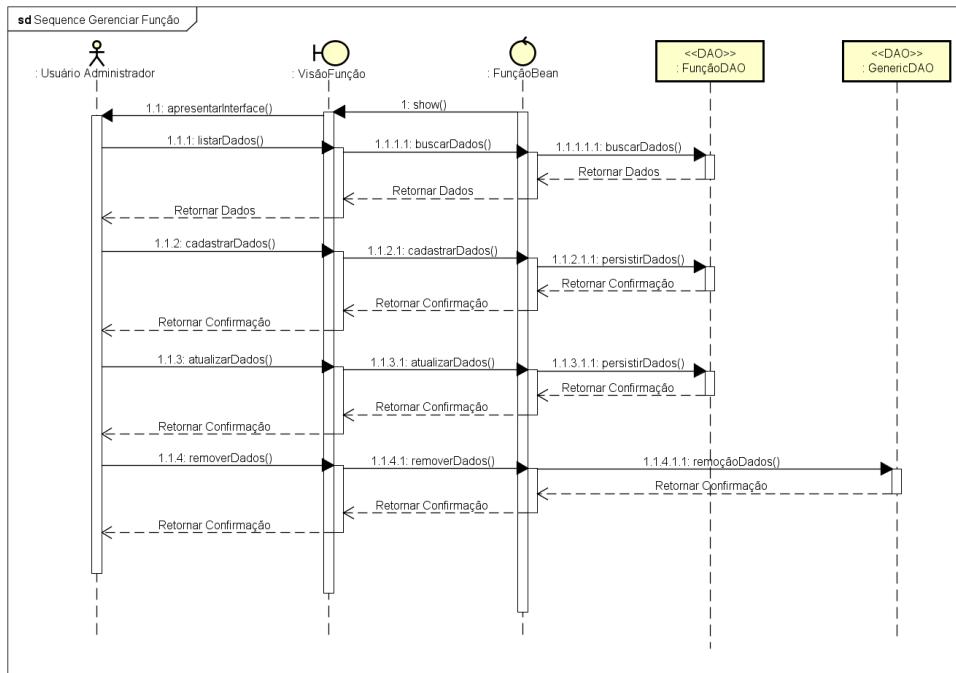


Fonte: Elaborado pelos autores, 2016.

### 3.16.8 Gerenciar Função

Na figura 3.30 é demonstrado o gerenciamento de usuário, o processo de listagem de dados já cadastrados, cadastrar, alterar e remover dados.

Figura 3.30 - Diagrama de Sequência: Gerenciar Função

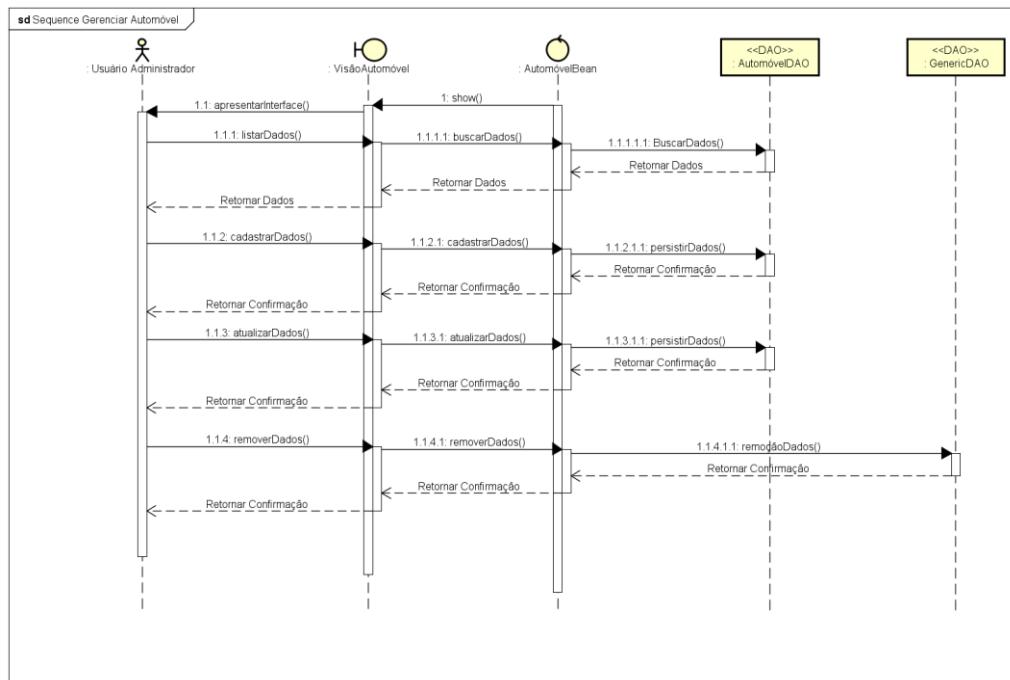


Fonte: Elaborado pelos autores, 2016.

### 3.16.9 Gerenciar Automóvel

Na figura 3.31 é demonstrado o gerenciamento de usuário, o processo de listagem de dados já cadastrados, cadastrar, alterar e remover dados.

Figura 3.31 - Diagrama de Sequência: Gerenciar Automóvel



Fonte: Elaborado pelos autores, 2016.

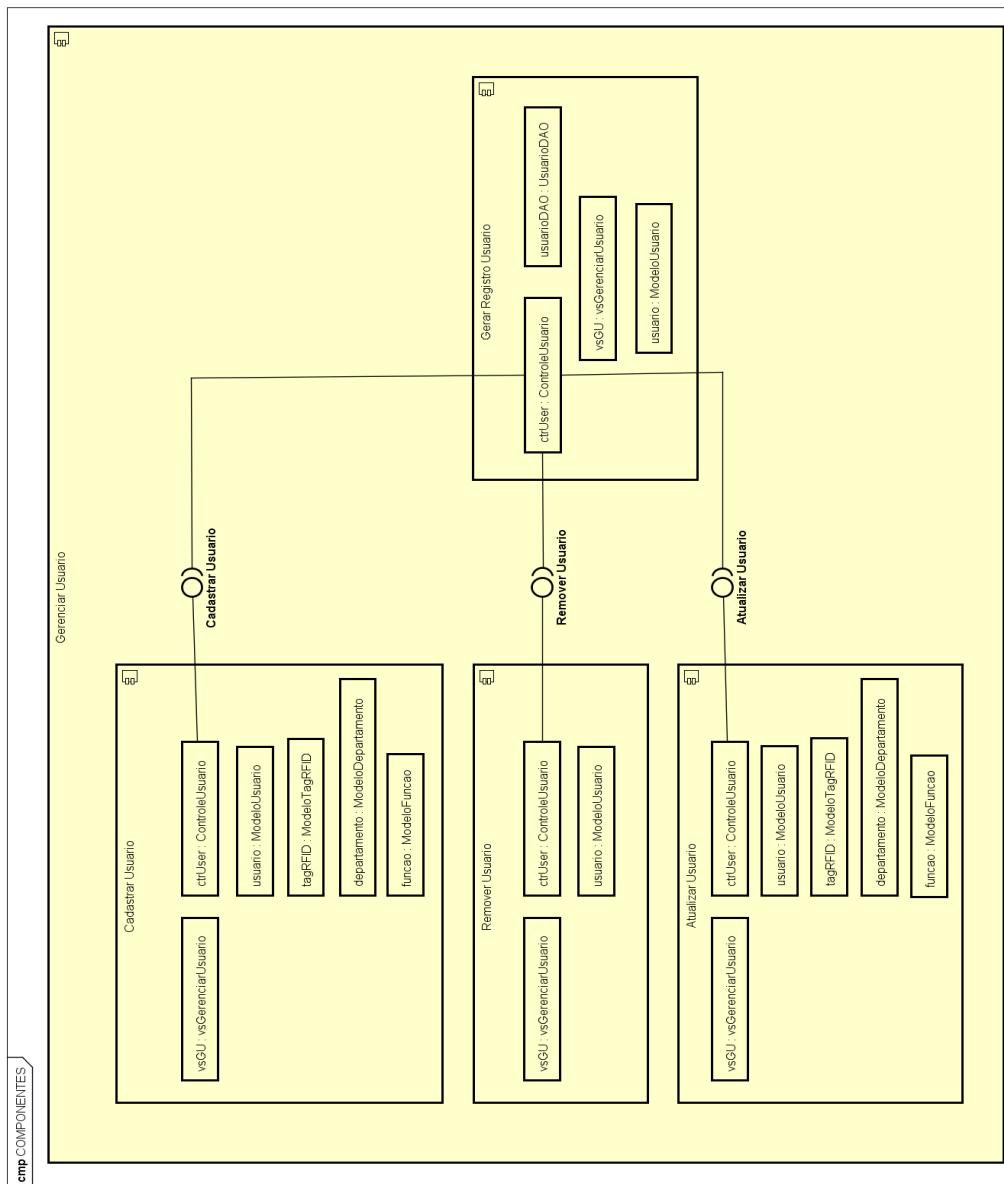
### 3.17 DIAGRAMAS DE COMPONENTES

Diagrama de componentes mostram vários componentes em um sistema e suas dependências, podendo ser criados separadamente ou combinados, apresentando uma visão estática de como o sistema está implementado e quais são os seus módulos de software.

Representa os módulos e as interfaces geradas e requeridas pelos módulos, ilustrando a modelagem do sistema e destacando a função de cada módulo.

#### 3.17.1 Gerenciar Usuário

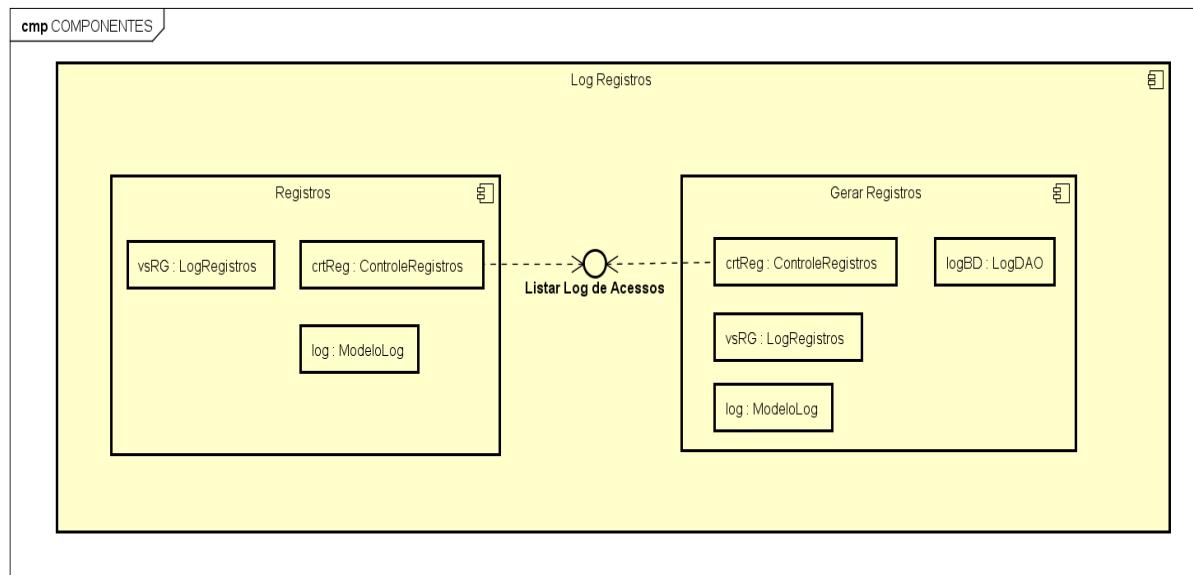
Figura 3.32 - Diagrama de Componentes: Gerenciar Usuário



Fonte: Elaborado pelos autores, 2016.

### 3.17.2 Gerenciar Log de Registro

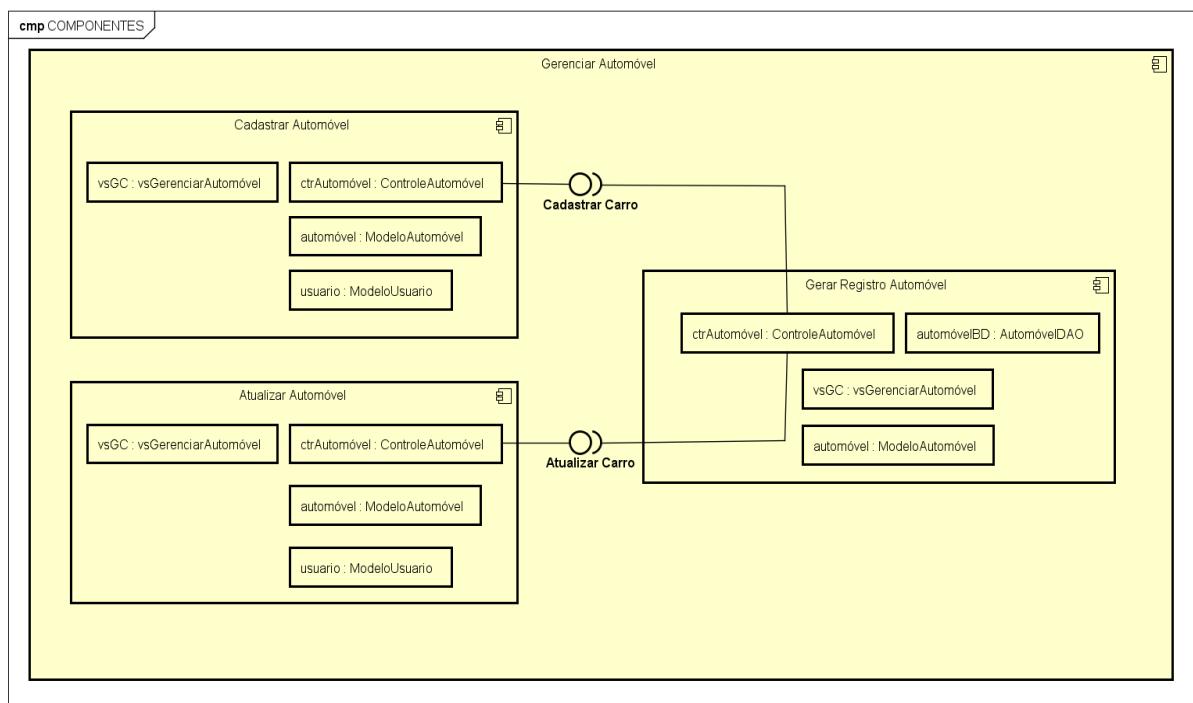
Figura 3.33 - Diagrama de Componentes: Gerenciar Log de Registro



Fonte: Elaborado pelos autores, 2016.

### 3.17.3 Gerenciar Automóvel

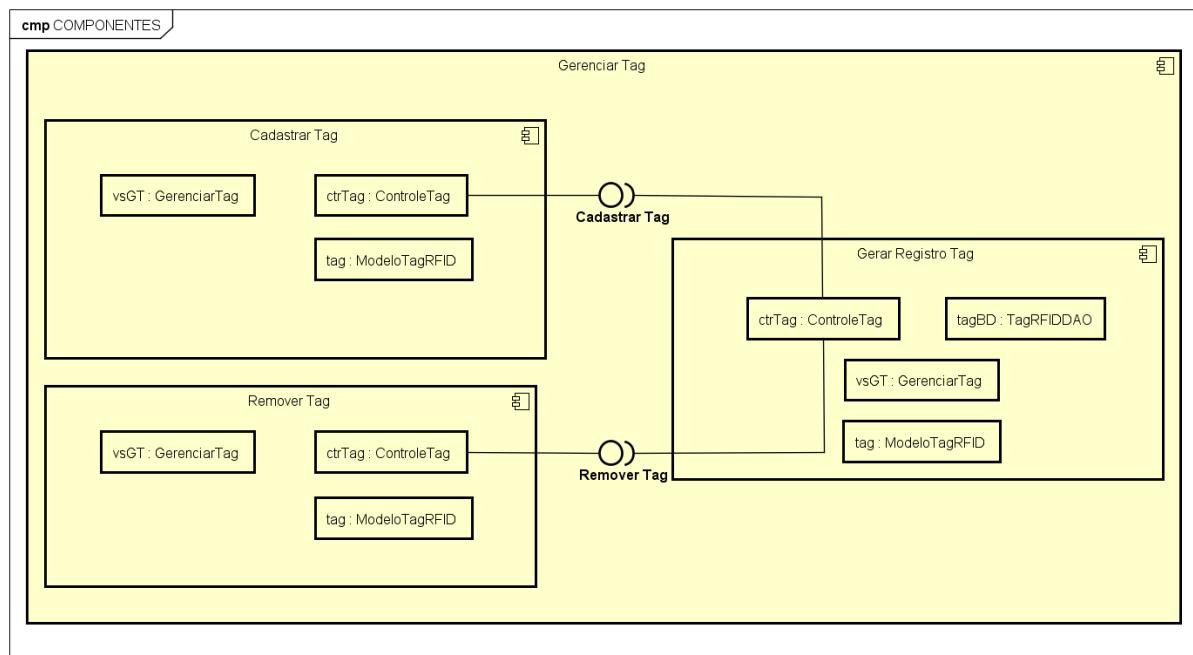
Figura 3.34 - Diagrama de Componentes: Gerenciar Automóvel



Fonte: Elaborado pelos autores, 2016.

### 3.17.4 Gerenciar Etiqueta RFID

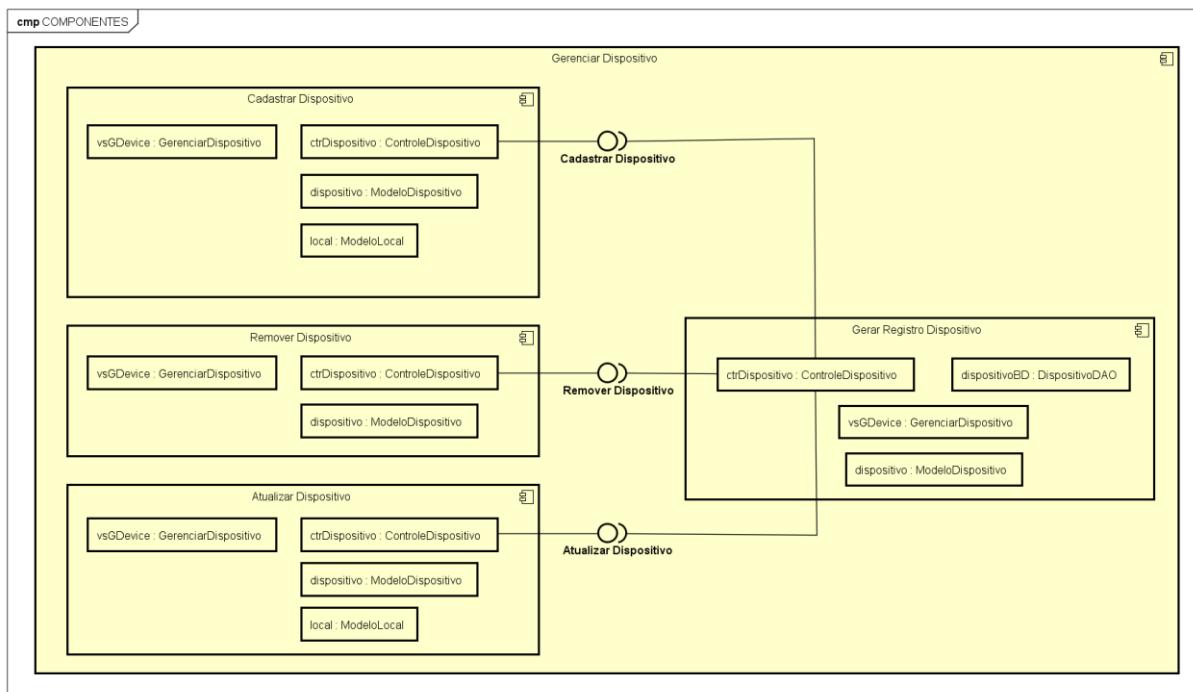
Figura 3.35 - Diagrama de Componentes: Gerenciar Etiqueta RFID



Fonte: Elaborado pelos autores, 2016.

### 3.17.5 Gerenciar Dispositivos

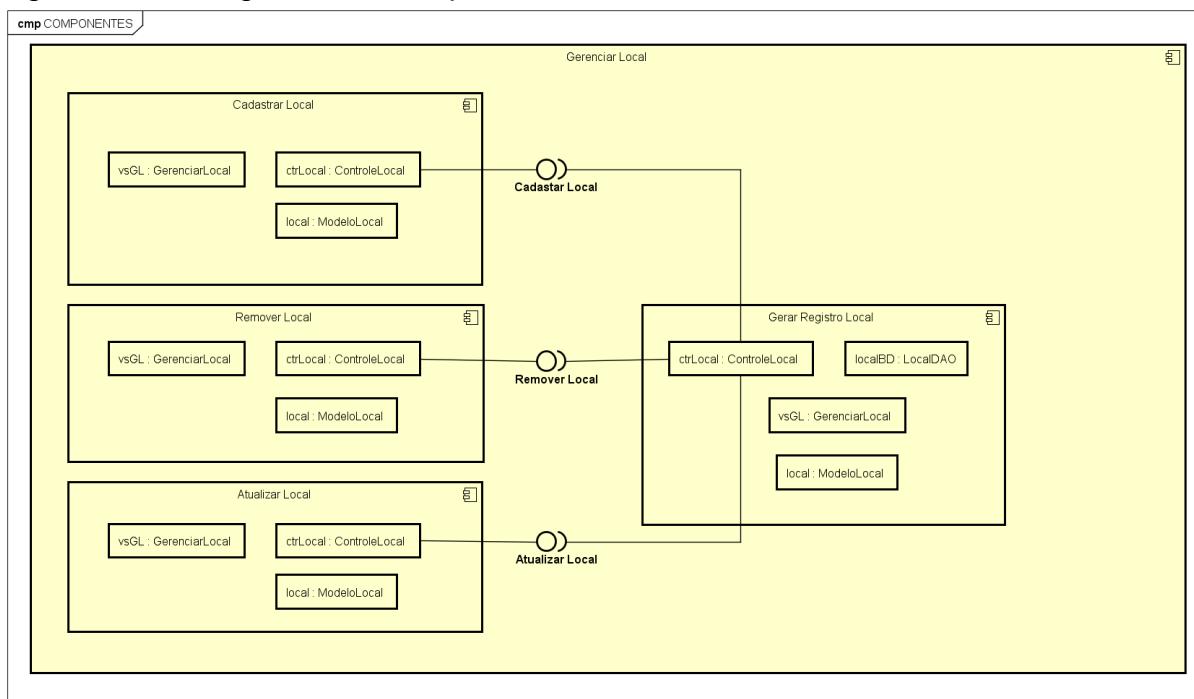
Figura 3.36 - Diagrama de Componentes: Gerenciar Dispositivos



Fonte: Elaborado pelos autores, 2016.

### 3.17.6 Gerenciar Local

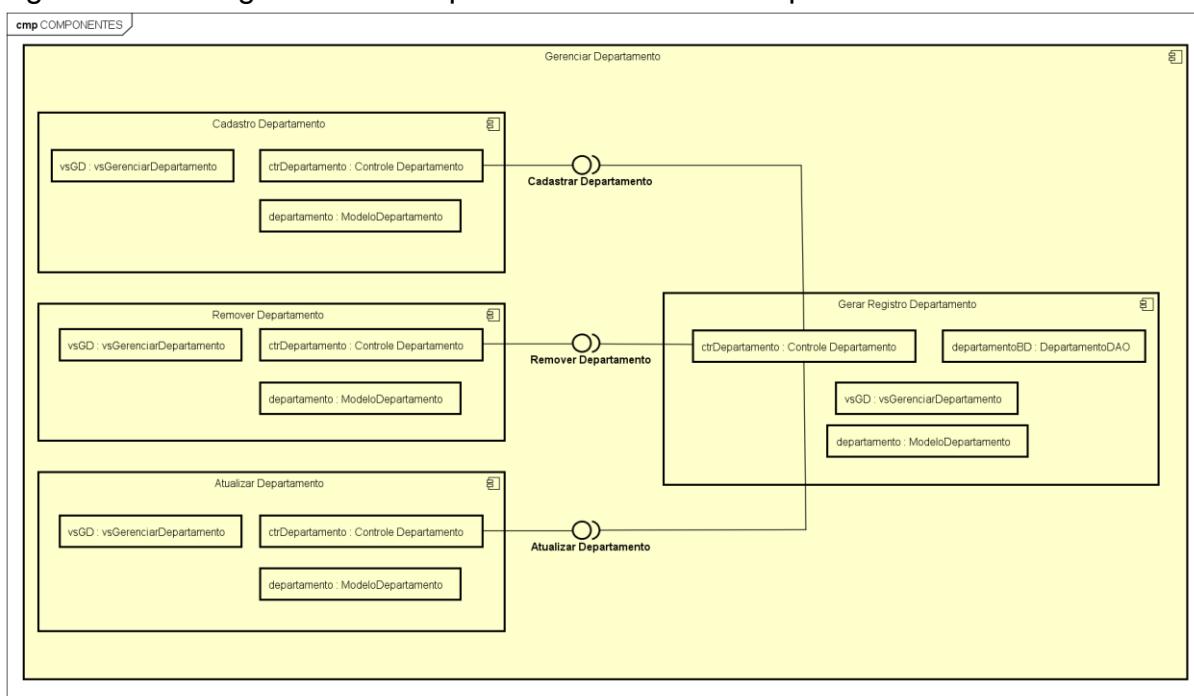
Figura 3.37 - Diagrama de Componentes: Gerenciar Local



Fonte: Elaborado pelos autores, 2016.

### 3.17.7 Gerenciar Departamento

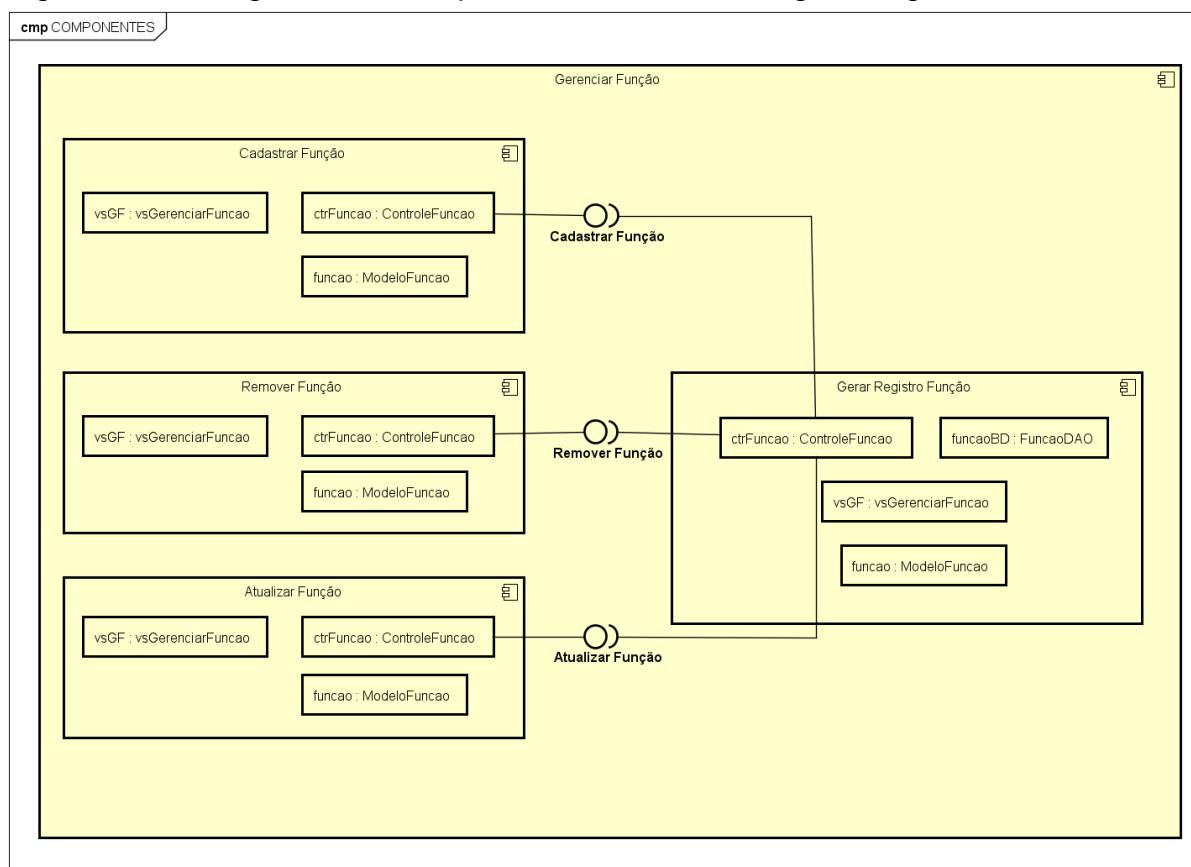
Figura 3.38 - Diagrama de Componentes: Gerenciar Departamento



Fonte: Elaborado pelos autores, 2016.

### 3.17.8 Gerenciar Função

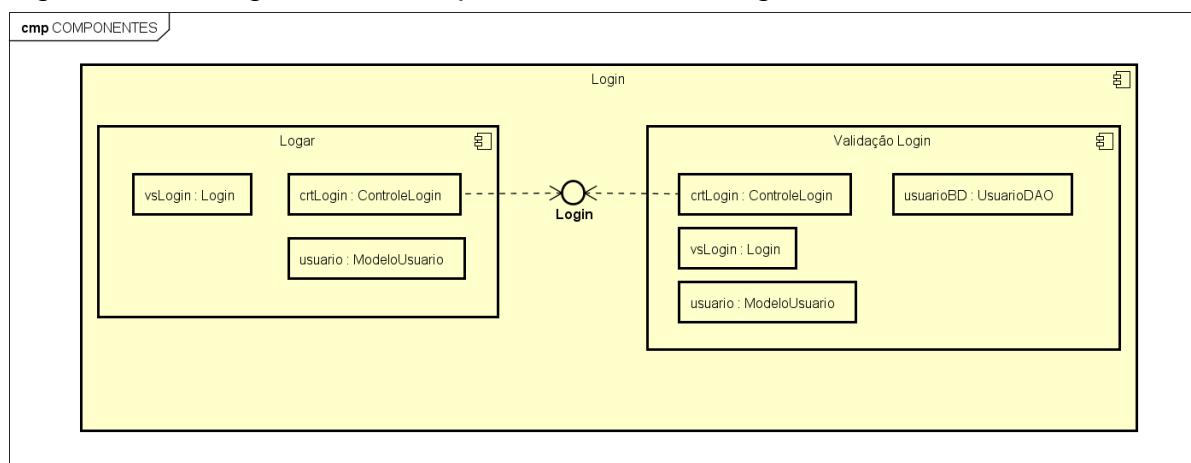
Figura 3.39 - Diagrama de Componentes: Gerenciar Log de Registro



Fonte: Elaborado pelos autores, 2016.

### 3.17.9 Fazer Login

Figura 3.40 - Diagrama de Componentes: Efetuar Login



Fonte: Elaborado pelos autores, 2016

## 4 IMPLEMENTAÇÃO

Este capítulo apresenta a implementação do sistema, algumas interfaces, de usuário seu funcionamento e exibindo partes do código fonte, assim como protótipo da automação da cancela.

A integração do sistema web com dispositivo da cancela ocorre através de uma conexão usando *socket* para troca de “sinais”, em sumo, por meio do sensor, o dispositivo envia um endereço hexadecimal da etiqueta RFID ao sistema, que por sua vez trata o dado recebido, verificando se a *tag* é válida ou não, caso seja válida, o registro de acesso é salvo e um sinal positivo retorna do sistema para o dispositivo, liberando o acesso, senão o sinal retornado é negativo e bloqueia o acesso. Esse meio de comunicação trata o dispositivo como um “servidor” e o sistema como “cliente”.

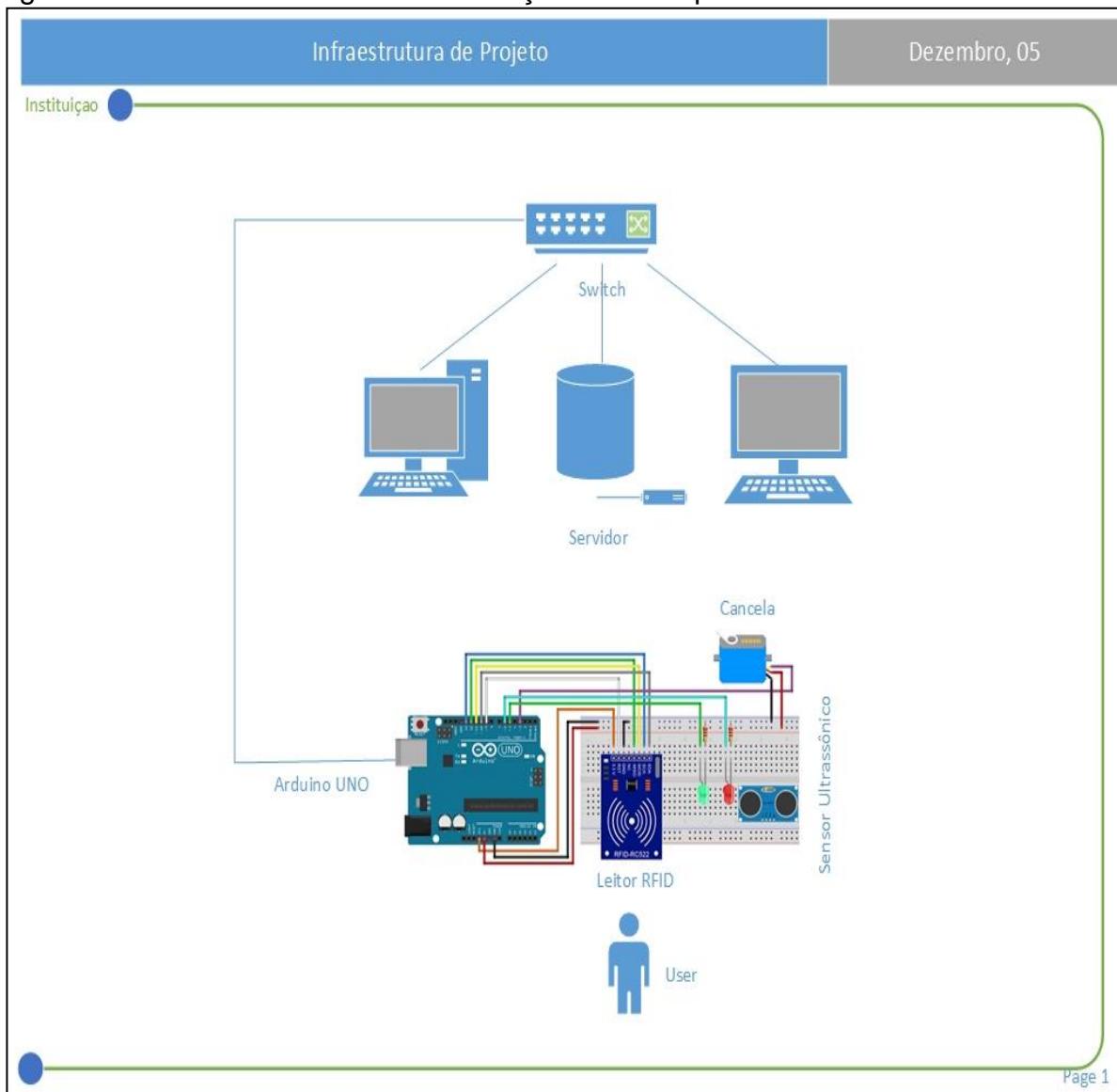
A Figura 4.1 foi ajustada para exemplificar o uso da cancela e na figura 4.2 é ilustrado um exemplo da comunicação do dispositivo e o sistema.

Figura 4.1 – Exemplo de cancela eletrônica com sensor



Fonte: Elaborado pelos autores, 2016

Figura 4.2 – Infraestrutura de comunicação entre dispositivo e o sistema

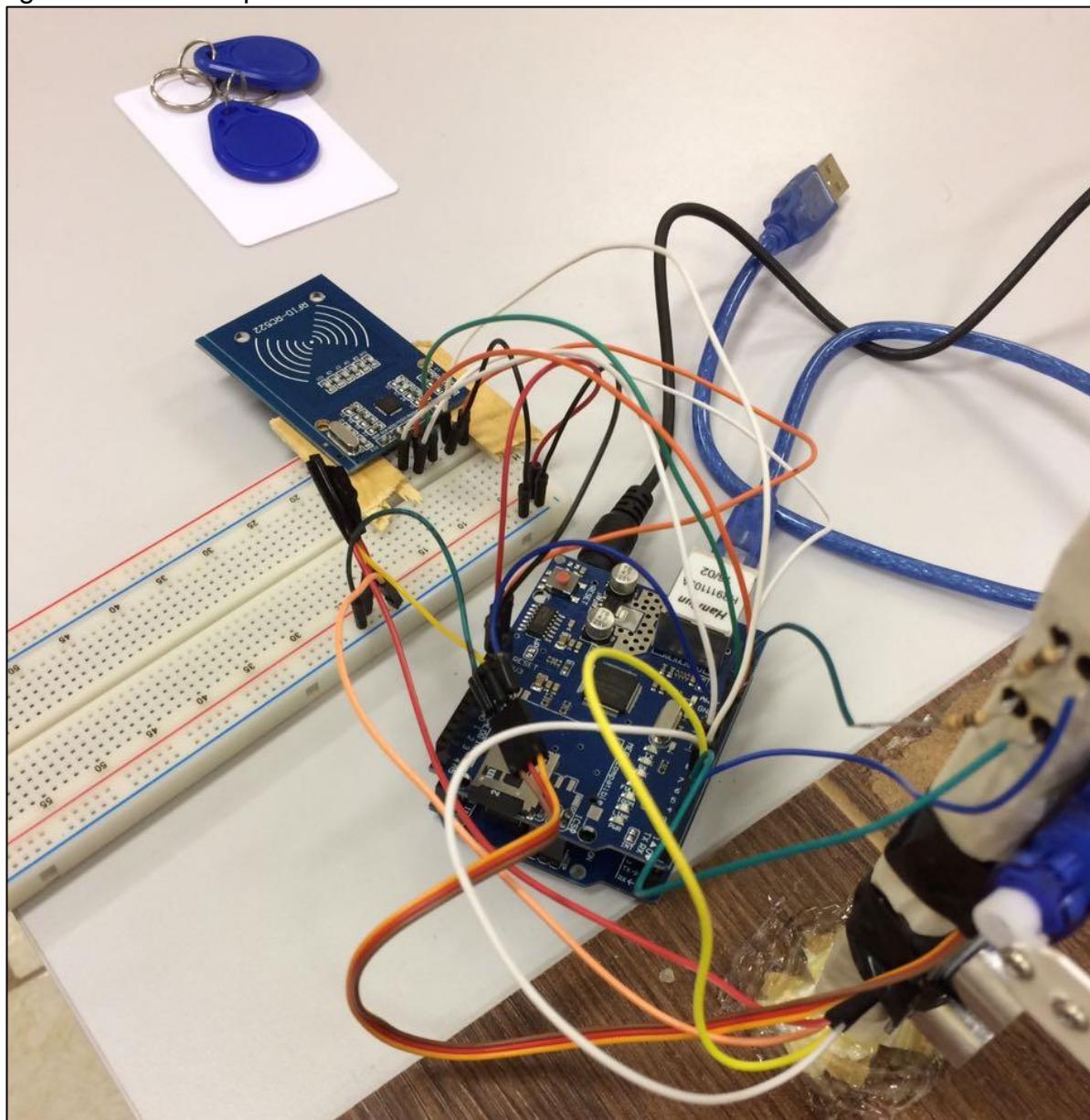


Fonte: Elaborado pelos autores, 2016

A partir da aprovação da comunicação pelo sistema da etiqueta RFID com o leitor de tags o Arduino verifica o sensor de proximidade que esta locado na parte externa da cancela, se o mesmo estiver operante envia um sinal para o servo motor realizar o processo de elevar a cancela a um determinado ângulo previamente definido, logo após a passagem do automóvel, caso o sensor verificar qualquer objeto sobre ele em uma estipulada altura não aceitável correndo o risco de colisão o mesmo não efetua o processo de descensão, ou seja, o braço da cancela não abaixará, mas se o sensor não verificar nenhum objeto próximo a ele conclui a atividade com sucesso.

Na figura 4.3 é apresentado o protótipo montado para a realização dos testes, este protótipo da imagem contém a placa Arduino alimentando uma protoboard que está conectando o sensor RFID, 2 LED, um servo motor e o sensor ultrassônico.

Figura 4.3 – Protótipo usado no trabalho



Fonte: Elaborado pelos autores, 2016

A imagem 4.4 e imagem 4.5 contém o código do Arduino, com as bibliotecas dos componentes usados, a definição onde os pinos são definidos e as “regras” que tratam dos dados recebidos

Figura 4.4 – Código do Arduino para comunicação e tratamento dos dados 1

```

//Bibliotecas para usar componentes
#include <SPI.h>
#include <MFRC522.h>
#include <Ultrasonic.h>
#include <Servo.h>
#include <Ethernet.h>

// DEFININDO AS PORTA PARA OS COMPONENTES
#define TRIGGER_PIN 5
#define ECHO_PIN 4

#define LED_VERMELHO 3
#define LED_VERDE 2

Servo myservo;
Ultrasonic ultrasonic (TRIGGER_PIN, ECHO_PIN);

#define SS_PIN 7
#define RST_PIN 6
MFRC522 mfrc522(SS_PIN, RST_PIN);

int x = 0;

// Definição do endereço físico(MAC ADDRESS) da placa de rede
byte mac[] = {0x48, 0xF4, 0xF7, 0x09, 0xB2, 0x09};
IPAddress ip(10, 10, 116, 201); // Define o endereço IP
IPAddress gateway(10, 10, 0, 1); // Define o gateway
IPAddress subnet(255, 255, 0, 0); // Define a máscara de rede
EthernetServer server(80); // Define a porta de acesso
EthernetClient client;

void setup() {
    // Inicializando pela 1ª vez
    Serial.begin(9600);
    // Inicia o Ethernet Shield
    Ethernet.begin(mac, ip, gateway, subnet);
    server.begin();
    // Exibe o endereço IP
    Serial.print("server is at ");
    Serial.println(Ethernet.localIP());
    pinMode(LED_VERMELHO, OUTPUT);
    pinMode(LED_VERDE, OUTPUT);
    myservo.attach(8); //Define a porta de controle do Servo
    Serial.begin(9600); // Inicia a serial
    SPI.begin(); // Inicia SPI bus
    Serial.println();
    myservo.write(0);
    digitalWrite(LED_VERMELHO, HIGH);
    digitalWrite(LED_VERDE, LOW);
    delay(4000);

}

int conectado = 0;
void loop() {
// Verifica a conexão do cliente

```

Fonte: Elaborado pelos autores, 2016

Figura 4.5 – Código do Arduino para comunicação e tratamento dos dados 2

```

client = server.available();
//if (client) {
if (client.connected()) {
conectado = 1;
if (client.available() && conectado == 1) {
String comando = "";
// Recebe o comando do cliente
while (client.available()) {
comando += (char) client.read();
}
Serial.println(comando);

Serial.print("Mensagem : ");
// Verifica de tipo do comando liberando ou não o acesso
if (comando == "LIBERADO") {
Serial.println("USUARIO LIBERADO !");
Serial.println();
myservo.write(45);
digitalWrite(LED_VERMELHO, LOW);
digitalWrite(LED_VERDE, HIGH);
delay(6000);

// O sensor ultrassônico verifica se o veículo está em baixo da
//cancela para ela não baixar e colidir com o veículo
float cmMsec, inMsec;
long microsec = ultrasonic.timing();
cmMsec = ultrasonic.convert(microsec, Ultrasonic::CM);
inMsec = ultrasonic.convert(microsec, Ultrasonic::IN);
Serial.print(", CM: ");
Serial.print(cmMsec);
} else {
Serial.println("TAG NAO LIBERADA");
for (x = 0; x <= 5; x++) {

digitalWrite(LED_VERMELHO, HIGH);
digitalWrite(LED_VERDE, LOW);
delay(200);
digitalWrite(LED_VERMELHO, LOW);
digitalWrite(LED_VERDE, HIGH);
delay(200);
}
}
}
}
client.stop();
// Sensor ultrassônico verifica se ainda há o veículo passando
float cmMsec1, inMsec1;
long microsec1 = ultrasonic.timing();
cmMsec1 = ultrasonic.convert(microsec1, Ultrasonic::CM);
inMsec1 = ultrasonic.convert(microsec1, Ultrasonic::IN);
if (cmMsec1 > 20) {
Serial.print(", CM: ");
Serial.println(cmMsec1);
delay(2000);
myservo.write(0);
}
}
}

```

Fonte: Elaborado pelos autores, 2016

Na figura 4.6, tem-se a tela inicial do sistema, onde o usuário deve fazer o *login* no sistema, seus dados de acesso serão verificados, validando ou não o seu acesso e caso seja válido, o usuário é redirecionado a página de acordo o tipo, sendo “usuário administrativo” ou “usuário comum”.

Figura 4.6 - Interface de Login



Fonte: Elaborado pelos autores, 2016

Após o *login*, é exibida a tela de registros, nesta interface estão os dados relacionados ao log de acesso de um dispositivo e o usuário, ou seja, qual usuário que possui *tag* passou por qual dispositivo, em que horas e data específica. Caso o usuário *logado* seja do tipo “administrador”, o menu exibido contém opções de gerenciamento das entidades e funções do sistema. Se o tipo de usuário for “comum”, o menu apresenta somente os registros do mesmo.

Figura 4.7 - Interface de Log de Registro

Registro de Acesso						
Menu	ID	Nome	Tag	Data	Hora	Local
Gerenciar	1	Nikola Tesla	7E A4 C6 B2	14/08/2016	13:49	Secretaria
• Cargo	2	João Pedro	D3 A5 B4 CC	07/09/2016	14:30	Laboratório A
• Carro	3	Maria	AE 5C 6B EE	20/10/2016	19:00	Biblioteca
• Departamento	4	Marcel	D3 A5 B4 CC	27/11/2016	08:10	Secretaria
• Dispositivo	5	Anderson	7D 3B CA 2E	28/11/2016	15:59	Sala 2
Acessos						
• Permissões						
• Registros						
• Liberar/Verificar						
Opcões						
• Conta						
• Sair						

Fonte: Elaborado pelos autores, 2016

Por exemplo, a tela de gerenciamento de “Usuários”, tendo como o primeiro conteúdo a lista de usuários já cadastrados, na primeira aba, sendo possível alterar ou remover o usuário.

Figura 4.8 - Interface de Gerenciamento de Usuário

ID	Nome	TAG	Email	Info	Alterar	Excluir
1	Nikola Tesla	A1 B2 C3 D4	tesla@email.com	<input type="button" value="P"/>	<input type="button" value="A"/>	<input type="button" value="E"/>
2	João Pedro	F4 U2 J4 G1	joao@email.com	<input type="button" value="P"/>	<input type="button" value="A"/>	<input type="button" value="E"/>
3	Maria Eduarda	AA BB CC DD	maria@email.com	<input type="button" value="P"/>	<input type="button" value="A"/>	<input type="button" value="E"/>
4	Anderson	DC H2 A2 BD	anderson@email.com	<input type="button" value="P"/>	<input type="button" value="A"/>	<input type="button" value="E"/>
5	Marcel	C1 D7 C3 EA	marcel@email.com	<input type="button" value="P"/>	<input type="button" value="A"/>	<input type="button" value="E"/>

Fonte: Elaborado pelos autores, 2016

Também é possível visualizar todas as informações cadastradas relacionadas a cada usuário como exibido na figura 4.9.

Figura 4.9 - Interface de Gerenciamento de Usuário com caixa de Informações

ID	Nome
1	Nikola Tesla
2	João Pedro
3	Maria Eduarda
4	Anderson
5	Marcel

**Usuário Info**

ID:	2
Nome:	João Pedro
TAG:	F4 U2 J4 G1
CPF:	347.346.324-32
Email:	joao@email.com
Departamento:	TI
Cargo:	Auxiliar TI
Telefone:	(14) 98172-8378
Tipo:	Administrador
Login:	joao_pedro

Fonte: Elaborado pelos autores, 2016

Na outra aba é exibido o formulário de cadastro, em que são preenchidas as informações necessárias para o uso do sistema, além de associar ou não uma tag ao usuário, pois ele pode fazer uso somente do sistema web como administrador e não usar o dispositivo.

Figura 4.10 - Interface de Cadastro de Usuário

**Gerenciar Usuários**

**Menu**

- Gerenciar
  - Automóvel
  - Cargo
  - Departamento
  - Dispositivo
  - Local
  - Tag
  - Usuário
- Acessos
- Opções
  - Conta
  - Sair

**Usuários** | **Cadastrar**

\*Nome: João Pedro  
TAG: F1 D3 CC A4  
\*CPF: 489.023.849-28  
\*Email: joao@email.com  
\*Telefone(Cel.): (14) 99199-1919  
\*Departamento: Administração  
\*Cargo: Coordenador  
\*Tipo: Administrador  
\*Login: joao  
\*Senha: \*\*\*\*\*

Fonte: Elaborado pelos autores, 2016

As interfaces gráficas de gerenciamento não diferem muito uma das outras, as mudanças se aplicam nos tipos de atributos. Algumas possuem particularidades para seus relacionamentos, como a tela de cadastro de carro que precisa referenciar o usuário caso ele faça uso do sistema com *tag*, assim como outras telas também referenciam outras entidades.

Figura 4.11 - Interface de Cadastro de Automóveis

**Gerenciar Carros**

**Menu**

- Gerenciar
  - Cargo
  - Carro
  - Departamento
  - Dispositivo
  - Local
  - Tag
  - Usuário
- Acessos
- Opções
  - Conta
  - Sair

**Automóveis** | **Cadastrar**

\*Usuário: João Pedro  
\*Marca: Ford  
\*Modelo: Mustang  
\*Cor: Vermelho  
\*Tipo: Muscle  
\*Placa: AWP-1995

Fonte: Elaborado pelos autores, 2016

Na interface de Permissão de Acesso são exibidos os dispositivos e usuários cadastrados, associando quais dispositivos o usuário tem pode ter acesso.

A interface de Gerenciamento de Dispositivo lista os dispositivos cadastrados e possui a aba de cadastro.

**Figura 4.12 - Interface de Informações de Cadastro de Dispositivo**

A interface de Gerenciamento de Dispositivo exibe uma lista de dispositivos cadastrados. A interface é dividida em duas seções principais: "Dispositivos" e "Cadastrar". A seção "Cadastrar" contém campos para inserir informações de um novo dispositivo, incluindo: \*Dispositivo: Trava Eletrônica; \*Local: Laboratório A; \*IP: 192.168.1.109; \*Porta: 80; \*User: admin; \*Password: admin. Abaixo dos campos, há botões para Salvar, Limpar e Cancelar. No lado esquerdo, um menu lateral com opções: Gerenciar (Cargo, Carro, Departamento, Dispositivo, Local, Tag, Usuário), Acessos (Permissões, Registros, Liberar/Verificar) e Opções (Conta, Sair).

Fonte: Elaborado pelos autores, 2016

A figura 4.13 exibe a interface onde se pode verificar a validade uma tag e\ou liberar o acesso imediatamente.

**Figura 4.13 - Interface para verificar a tag e liberar o acesso ao dispositivo**

A interface para verificar a tag e liberar o acesso ao dispositivo exibe uma lista de dispositivos. A interface é dividida em duas seções principais: "Verificar TAG" e "Liberar Acesso". A seção "Verificar TAG" contém um campo para inserir a tag e um botão para realizar a verificação. A seção "Liberar Acesso" contém um botão para liberar o acesso. No lado esquerdo, um menu lateral com opções: Gerenciar (Automóvel, Cargo, Departamento, Dispositivo, Local, Tag, Usuário), Acessos (Permissões, Registros, Liberar/Verificar) e Opções (Conta, Sair).

Fonte: Elaborado pelos autores, 2016

A tela “Conta” apresenta as informações referente ao usuário *logado* na sessão.

Figura 4.14 - Interface de Informações da Conta de Usuário

The screenshot shows a user interface for managing account information. On the left, there is a sidebar menu with three main sections: 'Gerenciar' (Manage), 'Acessos' (Accesses), and 'Opções' (Options). Under 'Gerenciar', there are links for 'Cargo', 'Carro', 'Departamento', 'Dispositivo', 'Local', 'Tag', and 'Usuário'. Under 'Acessos', there are links for 'Permissões', 'Registros', and 'Liberar/Verificar'. Under 'Opções', there are links for 'Conta' and 'Sair'. The main content area is titled 'Informações do Usuário' (User Information) and displays the following data in a table:

ID:	2
Nome:	João Pedro
TAG:	F4 U2 J4 G1
CPF:	435.345.353-45
Email:	joao@email.com
Telefone(Cel.):	(34) 54353-6535
Departamento:	Auxiliar TI
Cargo:	TI
Tipo:	Administrador
Login:	joao_pedro

Fonte: Elaborado pelos autores, 2016

## CONCLUSÃO

A tecnologia avança depressa, em intervalos cada vez mais curtos, por conta disso, a necessidade de implantação da ciência tecnológica faz-se inevitável pelo fato da mesma prover qualidade de vida e facilidade em desempenho de atividades muitas vezes impossíveis de serem realizados sem. A Tecnologia da Informação é a área de conhecimento responsável por criar, administrar e manter a gestão da informação em diversos aspectos da humanidade. Partindo deste princípio e accordado com a realidade da instituição onde não se encontra métodos de gerência, monitoramento e controle de acessos o estudo relacionado a este trabalho é apresentado com a proposta de desenvolvimento de um sistema para gerenciamento de controle de acesso e os registros dos dispositivos conectados por uma rede institucional, tudo isso sucedido por meio de uma placa micro controladora, sensores, etiquetas de rádio frequênci a e o sistema implementado.

O sistema foi desenvolvido na plataforma Web propiciando o acesso a partir de qualquer navegador interligado a internet independentemente da localização do usuário, possibilitando incumprir informações ao sistema de fora da instituição. O protótipo vai possibilitar a identificação automática por meio de sinais de rádio e admitir a recuperação e o armazenamento de dados por meio de dispositivos denominados etiquetas RFID, a placa controladora Arduino será responsável pela comunicação com a tag RFID e a transmissão dos logs de acessos por meio físico através da conexão de rede durante o trajeto do Arduino até a estação do servidor locado na instituição. Além disso a placa Arduino também se vê necessária para gerenciar o leitor de tags e coordenar toda a operação física do dispositivo que no momento de sua comunicação com a rádio frequência da tag RFID e o leitor de sinal do mesmo permitirá a função mecânica do equipamento que está sendo superintendido.

No presente projeto foi utilizada a linguagem de programação Java como citado anteriormente, utilizando API's de comunicação, *Netbeans* para desenvolver e aplicar a linguagem de fato, *Servlets*, a UML para a construção e documentação de softwares por meio da aplicabilidade de diagramas, *Primefaces*, Placa controladora Arduino para efetivar a comunicação de dispositivo e sistema, conceitos de MVC e orientações acerca de instruções ao realizar o trabalho.

Após a implantação do sistema, o estabelecimento de toda a estrutura física e a inserção de dispositivos gerenciáveis como cancelas eletrônicas e portas, dentre outros acredita-se que consequentemente haverá uma otimização nos processos das atividades e uma maior gestão e controle e segurança de acesso sobre todos os ambientes escolares e, por conseguinte de todas as entidades da faculdade.

Dificuldades surgiram durante o desenvolvimento do projeto, como por exemplo a integração entre o sistema web e o dispositivo, ou seja, a comunicação dentre os mesmos, pois não há na grade do curso nenhuma disciplina relacionada a Arduino até o presente momento da conclusão deste trabalho, também ocorreram problemas técnicos quanto ao sensor RFID que estava com mal funcionamento, atrasando e impossibilitando alguns testes finais.

O projeto final possui algumas limitações devido as dificuldades e problemas encontrados durante o decorrer do desenvolvimento deste trabalho, sendo uma das restrições a comunicação do sistema com múltiplos dispositivos, embora o projeto já esteja estruturado para tratar essas comunicações. Referente aos problemas no sensor RFID, ficou indisposto a leitura das etiquetas de rádio frequência (*tag*) pelo leitor RFID.

Com relação as futuras melhorias pode-se pensar em prover novos serviços como a definição e correlação de qual usuário tem acesso a qual dispositivo e com essa concepção estabelecer métodos de gerência para a atividade, trabalhando a comunicação de múltiplas conexões entre o sistema e os dispositivos, providenciar formas de manter todos os serviços oferecidos em atividade com equipamentos que armazenam energia, equipamentos que geram energia, links de redundância para caso de falta de rede, elucidar diretrizes para a tomada de decisão em caso de falha ou falta de utilitários como energia, internet, mal funcionamento de dispositivos dentre outras possíveis situações problemas, tudo isso com a proposta de procurar sempre manter os serviços em funcionamento.

Com a factual realização da proposta do projeto o ambiente institucional permanece mais seguro e gerenciado, portanto assim oferecendo muito mais controle e administração sobre a instituição e todos os seus ativos processos.

## REFERÊNCIAS BIBLIOGRÁFICAS

ARAÚJO, T. **Top 5 Projetos com Arduino e Motores.** 22 ago. 2014. Disponível em: <<http://blog.fazedores.com/top-5-projetos-com-arduino-e-motores/>>. Acessado em: 30 abr. 2016.

ARDUINO. **Arduino LCD Screen.** ARDUINO.CC. Disponível em: <<https://www.arduino.cc/en/Main/GTFT>>. Acesso em: 10 mai. 2016.

\_\_\_\_\_. **Arduino Ethernet Shiel V2.** ARDUINO.CC. 2016. Disponível em: <<https://www.arduino.cc/en/Main/ArduinoEthernetShield>>. Acesso em: 08 mai. 2016.

\_\_\_\_\_. **Arduino Mega 2560 Board.** ARDUINO.CC. 2016. Disponível em: <<https://www.arduino.cc/en/Main/arduinoBoardMega2560>>. Acesso em: 06 mai. 2016.

\_\_\_\_\_. **Arduino Nano Pinout Diagram.** 07 fev. 2013. Disponível em: <<http://forum.arduino.cc/index.php?topic=147582.0>>. Acessado em: 30 abr. 2016

\_\_\_\_\_. **Arduino UNO Board.** ARDUINO.CC. 2016. Disponível em: <<https://www.arduino.cc/en/main/arduinoBoardUno>>. Acesso em: 06 mai. 2016.

\_\_\_\_\_. **Introduction.** ARDUINO.CC. 2016. Disponível em: <<https://www.arduino.cc/en/Guide/Introduction#>>. Acesso em: 05 mai. 2016.

ARDUINO E CIA. **Controle de Acesso com Módulo RFID RC522.** ARDUINO E CIA. 08 dez. 2014. Disponível em: <<http://www.arduinoecia.com.br/2014/12/controle-de-acesso-modulo-rfid-rc522.html>>. Acesso em: 19 mar. 2016.

BAPTISTELLA, A. **Abordando a arquiterura MVC, e design partterns: Observer, Composite, Strategy.** 2009. <<http://www.artigonal.com/tecnologiasartigos/abordando-a-arquitetura-mvc-e-design-patterns-observer-composite-strategy-1017208.html>> Acesso em: 24 out. 2016.

BIBLIOTECA DA ENGENHARIA. **Resistores.** 2015. Disponível em: <<http://www.bibliotecadaengenharia.com/2016/03/resistores-codigo-cores.html>>. Acesso em: 23 mar. 2016.

CIRIACO, D. **Como funciona a RFID?.** TecMundo, 17 ago. 2009. Disponível em: <<http://www.tecmundo.com.br/tendencias/2601-como-funciona-a-rfid-.htm>> Acesso em: 19 mar. 2016.

DEITEL, P. **Java Como Programar.** 8. ed. São Paulo: Person Education, 2010.

DREAM INC.. **Protoboard.** Disponível em: <[http://www.dreaminc.com.br/sala\\_de\\_aula/wp-content/uploads/Protoboard-Tutorial.pdf](http://www.dreaminc.com.br/sala_de_aula/wp-content/uploads/Protoboard-Tutorial.pdf)>. Acesso em: 20 mar. 2016.

DENATRAN. **Frota de veículos.** Disponível em: <<http://www.denatran.gov.br/frota2016.htm>> Acesso em: 22 mar. 2016.

ELETRÔNICA DIDÁTICA. **Protoboard.** ELETRÔNICA DIDÁTICA. Disponível em: <<http://www.eletronicadidatica.com.br/protoboard.html>>. Acesso em: 11 mai. 2016.

FRONTEIRA TEC. **Portoboard, para que server e como utilizá-lo!** 13 jun. 2013. Disponível em <<http://fronteiratec.com/blog/protoboard-para-que-serve-e-como-utiliza-lo/>>. Acesso em: 11 mai. 2016.

GARRETT, F. **Como Funciona Raspberry Pi?** TecMundo. 29 nov. 2014. Disponível em: <<http://www.techtudo.com.br/noticias/noticia/2014/11/como-funciona-o-raspberry-pi-entenda-tecnologia-e-sua-aplicabilidade.html>>. Acessado em 28 abr. 2016.

GONÇALVES, E. **Desenvolvendo Aplicações Web com JSP, Servlets, JavaServer Faces, Hibernate, EJB 3 Persistence e Ajax.** 1. ed. Rio de Janeiro: Ciência Moderna, 2007.

GUEDES, G. **T.A. UML 2 guia prático.** 2. ed. São Paulo: Novatec, 2014. Disponível em: <<https://www.novatec.com.br/livros/uml2-2ed/capitulo9788575223857.pdf>>. Acesso em: 10 nov. 2016.

KUSHNER, D. **The Making of Arduino.** 26 out. 2011. Disponível em: <<http://spectrum.ieee.org/geek-life/hands-on/the-making-of-arduino/0>> Acesso em: 06 mai. 2016.

MICROBERTS, M. **Arduino básico.** Tradução Rafael Zanolli. 1 ed. Novatec Editora Ltda., 2011.

MECÂNICA INDUSTRIAL. **O que é um sensor ultrassônico.** 2016. Disponível em: <<http://www.mecanicaindustrial.com.br/598-o-que-e-um-sensor-ultrassonico/>>. Acesso em: 13 mai. 2016.

MEDEIROS, M. **Desenvolvendo Software com UML 2.0 definitivo.** São Paulo: Pearson Makron Books. 2004.

MORIMOTO, C. E. **Categorias de Cabos.** 01 abr. 2008. Disponível em: <<http://www.hardware.com.br/livros/redes/categorias-cabos.html>>. Acesso em 06 mai. 2016

PAPEL DIGITAL. **Cabo de rede.** 2016. Disponível em: <<https://www.palpitdigital.com/cabo-de-rede/>>. Acesso em: 03 mai. 2016.

PERKONS. **Tecnologia a favor do trânsito.** 01 jun. 2007. Disponível em: <<http://www.perkons.com.br/detalhe-banco-de-pauta/1/tecnologia-a-favor-do-transito>>. Acesso em: 23 mar. 2016.

REED, F. **How Do Servo Motors Work.** Disponível em: <<http://www.jameco.com/jameco/workshop/howitworks/how-servo-motors-work.html>>. Acesso em: 13 mai. 2016.

SILVA, D. C. M. **Resistores.** Brasil Escola. Disponível em: <<http://brasilescola.uol.com.br/fisica/resistores.htm>>. Acesso em: 07 abr. 2016.

SINTES, A. **Aprenda programação orientada a objetos em 21 dias.** 1ª ed. São Paulo: Makron Books, 2002.

SOUZA, F. **Placas Arduino – História até o Arduino UNO.** 21 nov. 2013. Disponível em: <<http://www.embarcados.com.br/historia-ate-o-arduino-uno/>>. Acesso em: 28 abr. 2016.

JIMBO. **Resistors.** SPARKFUN Disponivel em: <<https://learn.sparkfun.com/tutorials/resistors>>. Acesso em: 16 mai. 2016.

. **Arduino – Primeiros Passos.** 06 nov. 2013. Disponível em: <<http://www.embarcados.com.br/arduino-primeiros-passos/>>. Acesso em: 29 abr. 2016.

. **Arduino LEONARDO.** 03 abr. 2014. Disponível em: <<http://www.embarcados.com.br/arduino-leonardo/>>. Acesso em: 29 abr. 2016.

. **Arduino MEGA 2560.** 28 abr. 2014. Disponível em: <<http://www.embarcados.com.br/arduino-mega-2560/>>. Acesso em: 29 abr. 2016.

. **Arduino UNO.** 29 nov. 2013. Disponível em: <<http://www.embarcados.com.br/arduino-uno/>>. Acesso em: 28 abr. 2016.

THOMSEN, A. **Controle de Acesso usando leitor RFID com Arduino.** 22 abr. 2014. Disponível em: <<http://blog.filipeflop.com/wireless/controle-acesso-leitor-rfid-arduino.html>>. Acesso em: 29 abr. 2016.