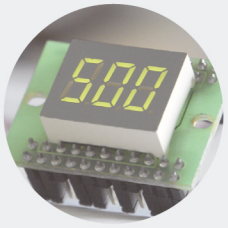
INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA ELECTROTÉCNICA E DE COMPUTADORES






Sistema para controlo de empréstimos em um armazém utilizando RFID

IGOR WASZCZYNSKI
julho de 2017

POLITÉCNICO
DO PORTO

Sistema para controlo de empréstimos em um armazém utilizando RFID

Igor Waszczynski

Departamento de Engenharia Eletrotécnica

Instituto Superior de Engenharia do Porto

2017

Este relatório satisfaz, parcialmente, os requisitos que constam da Ficha da Unidade Curricular de

Tese / Dissertação (TEDI), do 2º ano, do Mestrado em Engenharia Eletrotécnica e de Computadores

Candidato: Igor Waszczynski, Nº 1151903, 1151903@isep.ipp.pt

Orientação científica: Cecília Maria do Rio Fernandes Moreira Reis, cmr@isep.ipp.pt e Hermes Irineu Del Monego, hmonego@utfpr.edu.br

Departamento de Engenharia Eletrotécnica

Instituto Superior de Engenharia do Porto

4 de Julho de 2017

Agradecimentos

Aos meu pais por todo o apoio e amor incondicional.

Ao Prof.Dr.Hermes Irineu Del Monego e a Engª.Cecília Reis por todas as orientações, experiências e conhecimentos compartilhados durante esta trajetória e também por seus esforços e contribuições na realização da parceria entre a Universidade Tecnológica Federal do Paraná e o Instituto Superior de Engenharia do Porto, que proporcionam aos estudantes esta oportunidade singular de dupla titulação.

Aos meus familiares, professores, amigos e todos aqueles que participaram direta ou indiretamente da minha formação.

Homenagem Póstuma

Expresso minha gratidão ao Prof.Dr.Hilton José Silva de Azevedo que atuou como coordenador do curso de Engenharia Eletrônica na Universidade Tecnológica Federal do Paraná Campus Curitiba. Um dos idealizadores do programa de dupla titulação entre Universidade Tecnológica Federal do Paraná e o Instituto Superior de Engenharia do Porto.

Meus eternos reconhecimentos por ter sido um grande incentivador nesta minha jornada, posso afirmar que sem a sua valiosa contribuição esta obra não seria viabilizada.

Resumo

O projeto descrito no presente relatório enquadra-se no âmbito dos sistemas de informação, por possuir as características de armazenamento, organização e estruturação de dados, fornecendo e facilitando o acesso a informação.

Surgiu com o objetivo de apresentar uma ideia, utilizando-se a tecnologia de Radio-Frequency Identification (RFID) para identificação dos equipamentos, aprimorando assim o atual modelo de controle dos empréstimos no armazém do Departamento Acadêmico de Eletrônica (DAELN) da Universidade Tecnológica Federal do Paraná (UTFPR) que utiliza formulários de papel.

Desenvolvido para uma plataforma web, o sistema contém uma base de dados criada em SQL, o servidor escrito em Node.js e para execução das páginas web foi utilizada a linguagem JavaSpript. O novo sistema focou-se na resolução das deficiências anteriores e na implantação de novas funcionalidades tais quais geração de relatórios e abate de equipamentos.

Este projeto contribui com a universidade fornecendo um sistema de controlo eficaz, reduzindo a quantidade papel atualmente utilizada, proporcionado maior segurança aos equipamentos e também fornecendo informações relevantes aos seus utilizadores.

Palavras-chave

Sistema de informação, Node.js, RFID, JavaScript, SQL.

Abstract

The project described in this report fits within the scope of information systems because it has some characteristics - allows storage, organization and data structure in order to provide and facilitate information access.

Its goal is to present an idea to improve the current form of loan control, using RFID technology to identify the equipments in DAELN's warehouse at Federal University of Technology – Paraná .

Developed for a web platform the system database was created using SQL, its server was written in Node.js and the web pages were programed using JavaScript. The new system has focused on the resolution of the previous deficiencies and in the implementation of new functionalities such as generate reports and remove equipments.

This project contributes to the university providing an effective control system. Minimizing the current amount of paper used, keeping the equipments safe, and also providing relevant information to its users.

Keywords

Information systems, Node.js, RFID, JavaScript, SQL.

Índice

Índice de Figuras

Índice de Tabelas

Acrónimos

AJAX Asynchronous JavaScript and XML

API Application Program Interface

BD Base de Dados

Cefet-PR Centro Federal de Educação Tecnológica do Paraná

CI Circuito Integrado

CPU Central Processing Unit

CSS Cascading Style Sheets

DAELN Departamento Acadêmico de Eletrônica

E/S Entrada Saída

HTTP Hypertext Transfer Protocol

HTML HyperText Markup Language

IDE Integrated Development Environment

ISEP Instituto Superior de Engenharia do Porto

JDBC Java Database Connectivity

JSON JavaScript Object Notation

MS-DOS Microsoft Disk Operating System

NPM Node Package Manager

ODBC Open Database Connectivity

PDF Portable Document Format

PHP Portable Document Format

RFID Radio-Frequency Identification

SGBD Sistemas de Gestão de Base de Dados

SGBDR Sistemas de Gestão de Base de Dados Relacional

SI Sistema de informação

SPI Serial Peripheral Interface

SQL Structured Query Language

UART Universal Asynchronous Receiver/Transmitter

UID Unique Identifier

UML Unified Modeling Language

URL Uniform Resource Locator

UTFPR Universidade Tecnológica Federal do Paraná

WMS Warehouse Management System

XHR XMLHttpRequest

XML eXtensible Markup Language

Capítulo 1

Introdução

O presente capítulo faz uma introdução ao trabalho da dissertação, assim como a sua contextualização e apresentação dos objetivos. Por fim, descreve a estrutura do documento e os temas abordados em cada um dos capítulos.

1.1 Contextualização

A inspiração para este trabalho surgiu da observação a partir do modelo utilizado para controlo de presença dos alunos no Instituto Superior de Engenharia do Porto (ISEP), verificou-se que um sistema semelhante poderia ser utilizando para aperfeiçoar o processo de empréstimos dos equipamentos no armazém do Departamento Acadêmico de Eletrônica (DAELN) da Universidade Tecnológica Federal do Paraná (UTFPR).

A história da UTFPR, tem início na cidade de Curitiba, capital do estado brasileiro do Paraná em janeiro de 1910. Com o nome de Escolas de Aprendizes Artífices, a instituição fornecia formação educacional aos menos favorecidos da sociedade curitibana. Aos poucos a escola foi crescendo, o número estudantes aumentando e o ensino tornando-se cada vez mais profissional. Até que em 1937 passou a se chamar Liceu Industrial do Paraná, cinco anos depois (1942) foi instituída a rede federal de instituições de ensino

industrial e o Liceu Industrial do Paraná passou a chamar-se Escola Técnica de Curitiba, dando início aos primeiros cursos técnicos. Em 1959 o ensino técnico no Brasil foi unificado. A escola ganhou, maior autonomia e passou a chamar-se Escola Técnica Federal do Paraná [1].

Foram implantados aos primeiros cursos de curta duração de Engenharia em 1974. Quatro anos depois (1978), a Instituição foi transformada em Centro Federal de Educação Tecnológica do Paraná (Cefet-PR), passando a ministrar cursos de graduação plena. Em 1998, a diretoria do então Cefet-PR, criou um projeto de transformação da Instituição em Universidade Tecnológica. A primeira Universidade Tecnológica do Brasil. Atualmente possui treze campus no estado do Paraná (figura 1.1), mais de dois mil e quinhentos professores e aproximadamente trinta e um mil alunos matriculados [2].

Figura 1.1: Localização de todos os 13 campus da UTFPR no estado do Paraná - Com o campus Curitiba em destaque

O DAELN da UTFPR - Campus Curitiba-Centro (figura 1.2), possui mais de 20 laboratórios. Seu armazém contém quase 800 equipamentos disponíveis para empréstimos, atendendo as necessidades de centenas de alunos e professores, entretanto, o sistema de controlo dos equipamentos continua sem evolução a muitos anos.

Figura 1.2: Campus Curitiba-Centro da UTFPR

Atualmente a requisição de equipamentos é realizada manualmente, mediante preenchimento de um formulário em uma folha de papel (modelo no Anexo A) e retenção do crachá de identificação do requerente, um funcionário do departamento verifica todas as informações e conclui o empréstimo, completando o formulário com o(s) número(s) de referencia do(s) equipamento(s). A devolução de todos os equipamentos emprestados deve ocorrer até o final do expediente vigente, estando tudo de acordo o crachá é devolvido.

1.2 Objetivos

Este trabalho visa o desenvolvimento de um sistema capaz de automatizar a forma de empréstimo dos equipamentos no armazém do DAELN. Pretende eliminar os atuais formulários de papel, aumentar o controlo e segurança sobre os equipamentos da universidade, reduzir o trabalho dos funcionários e as margens de erros. Também pretende realizar a identificação dos utilizadores de um determinado equipamento, geração de relatórios com o histórico de empréstimos e possíveis avarias dos equipamentos.

1.2.1 Objetivo Geral

Idealizar, desenvolver e testar um sistema que facilite o empréstimo e controlo dos equipamentos, adaptando a atual realidade e necessidades da UTFPR. O sistema permite registar e manipular as informações referentes aos equipamentos e seus utilizadores.

Para o desenvolvimento do projeto iniciou-se com uma pequena lista de tarefas a serem executadas, conforme abaixo discriminadas:

1. Análise da forma como os empréstimos são realizados;
2. Levantamento e análise dos dados requisitados;
3. Criação de uma base de dados no sistema;
4. Realizações de testes;
5. Implementação do sistema.

1.2.2 Objetivo Específico

Criar uma base de dados com todos os alunos, professores e funcionários aptos ao empréstimo que estão na universidade, assim como registar todos os equipamentos do armazém.

Para identificação dos utilizadores e equipamentos serão utilizados cartões RFID, sendo que os empréstimos são restritos àqueles cujos cartões estão registados no sistema.

Optou-se pela utilização de *tags* RFID passivas (secção 2.7.2.2), considerando-se as vantagens custo beneficio diante das *tags* ativas (secção 2.7.2.1).

Para o desenvolvimento do sisteme optou-se pela plataforma *web*, considerando as facilidades de associação com a base de dados e a possibilidade de expandir e/ou integrar com outros sistemas.

1.3 Organização do relatório

Este documento encontra-se dividido em quatro capítulos. Neste primeiro capítulo efetua-se uma primeira abordagem ao projeto. O capítulo 2 explica com detalhes as tecnologias utilizadas no desenvolvimento do projeto. O capítulo 3 descreve a arquitetura do projeto, partes de códigos relevantes para funcionamento do sistema e funções especificas, assim como explicações para utilização do sistema e testes da aplicação. Por fim o último capítulo apresenta as conclusões, os objetivos realizados, as contribuições do projeto, limitações e trabalho futuro.

Capítulo 2

Estado da arte, tecnologias e ferramentas

Ao longo deste capítulo é feita uma breve referência ao sistema de informação e ao sistema de gestão de armazéns. A seguinte secção cita os principais diagramas da linguagem de modelação unificada. É realizada uma abordagem a base de dados, seus sistemas de gestão e manipulação. O capítulo aborda também a tecnologia RFID, Arduino e as linguagens de programação utilizadas.

2.1 Sistema de informação

Sistema de informação (SI) é uma expressão utilizada para descrever sistemas responsáveis pela coleta, organização, armazenamento e comunicação de dados. Tais sistemas são compostos por pessoas e computadores capazes de processar e interpretar as informações.

O SI tem auxiliado empresas e organizações na análise de dados, permitindo desenvolvimento e evolução em diversos campos [3]. Informação é um conjunto de dados (fatos obtidos através de leituras, observações, cálculos e medições) transformados de tal forma que ajudam a reduzir o futuro incerto

e, portanto, contribuem para o processo de tomada de decisão. A figura 2.1 exemplifica a transformação de dados em informação [4].

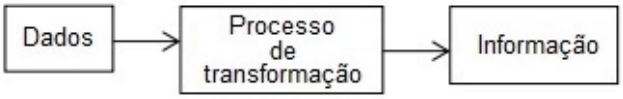


Figura 2.1: Processo de transformação de dados em informação - Adaptado de *Introduction to Management Information Systems*

A visão mais clássica do SI, representado por uma pirâmide (Figura 2.2), transmite a ideia que reflete a hierarquia das organizações. O nível inferior de gestão é aquele responsável pelas operações diárias, que seguem procedimentos organizacionais para alcançar seus objetivos. A gestão de nível médio representa a tomada de decisões táticas com perspectivas de médio prazo (de seis meses até dois anos), é frequentemente associada a unidades específicas para projetos específicos da organização. Por último a gestão de nível superior é responsável por tomadas de decisões estratégicas de longo prazo (cinco anos ou mais). Como os níveis de gestão possuem objetivos destintos, os sistemas de informação envolvido também diferem [5].

Figura 2.2: Pirâmide de hierarquia das organizações - Adaptado de *Introduction to Management Information system*

Com o aumento da complexidade e crescimento das organizações a necessidade por maiores informações torna-se eminente, e como consequência é exigido mais do SI, que deve ser capaz de se adaptar, proporcionando

informações de qualidade [6][7].

O sistema de gestão de armazéns em inglês Warehouse Management System (WMS) é um SI específico, responsável pela gestão mais precisa dos bens de um armazém, fornece informações relevantes tanto em nível operacional quanto de gestão.

Em nível operacional o WMS permite as organizações realizarem o controlo da entradas e saídas de bens e equipamentos, como também otimização dos espaços de armazenagem e informações de controlo em tempo real através de sistemas de identificação como o códigos de barras e RFID.

Para uma correta aplicação de um WMS é necessário ter em consideração os seguintes elementos [3]:

Hardware deve ter capacidade suficiente para conseguir trabalhar com o volume de dados e transações previstas.

Software deve ser cuidadosamente projetado e avaliado, para que possa ser eficaz e capaz de fornecer informações relevantes.

Informação deve ser detalhada de tal forma que a construção do sistema seja baseada nas necessidades reais da organização.

Pessoas e processos os dados obtidos e os processos devem ser realizados com precisão, e os colaboradores devem receber formação adequada para trabalhar de forma correta com o sistema.

2.2 Linguagem de modelagem unificada

Linguagem de modelagem unificada, em inglês Unified Modeling Language (UML), é utilizada no processo de projeto de *sofware*, representando um sistema de forma padronizada com o objetivo de definir o desenho e a estrutura do projeto.

Um SI é melhor representado e visualizado através da utilização de diagramas UML. Para facilitar a detecção de falhas e erros, assim como identificar alterações a serem realizadas, com o intuito de melhorar o sistema [8].

Os principais diagramas UML utilizados são:

Diagrama de Casos de Uso: Utilizado para identificar o que um sistema faz, descreve os serviços que devem ser disponibilizados a cada um dos diversos utilizadores (atores). A figura 2.3 apresenta um exemplo de diagrama de casos de uso de um SI de gestão para um grupo de pizzarias, que permite aos clientes efetuar encomendas na loja e através da Internet.

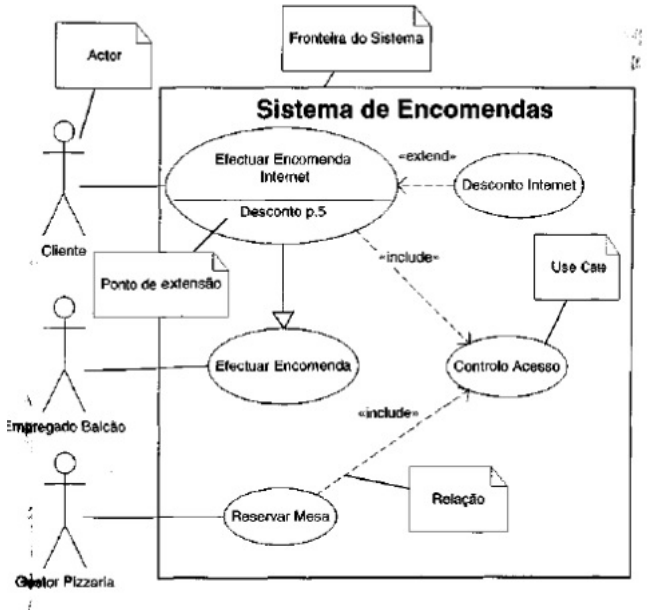


Figura 2.3: Exemplo de diagrama de casos de uso [9]

Diagrama de Classes: Utilizado para descrever a estrutura de informação (classes e suas relações) utilizada no sistema. Uma classe define os diferentes atributos e métodos de uma parte do SI. A figura 2.4 apresenta um exemplo para o diagrama de classes.

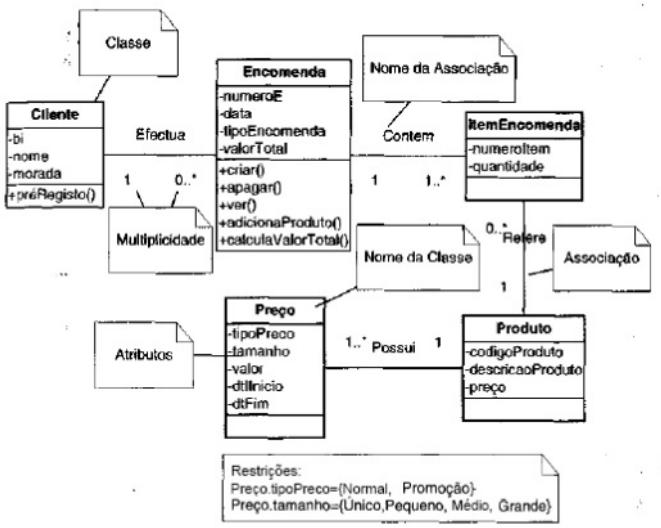


Figura 2.4: Exemplo de diagrama de classes [9]

Diagrama de Entidade-Relação: Utilizado para descrever as diferentes classes do sistema, as relações e restrições entre as propriedades de cada classe. A figura 2.5 apresenta vários exemplos de diagramas Entidade-Relação, descrevendo a cardinalidade e relacionamento de cada um.

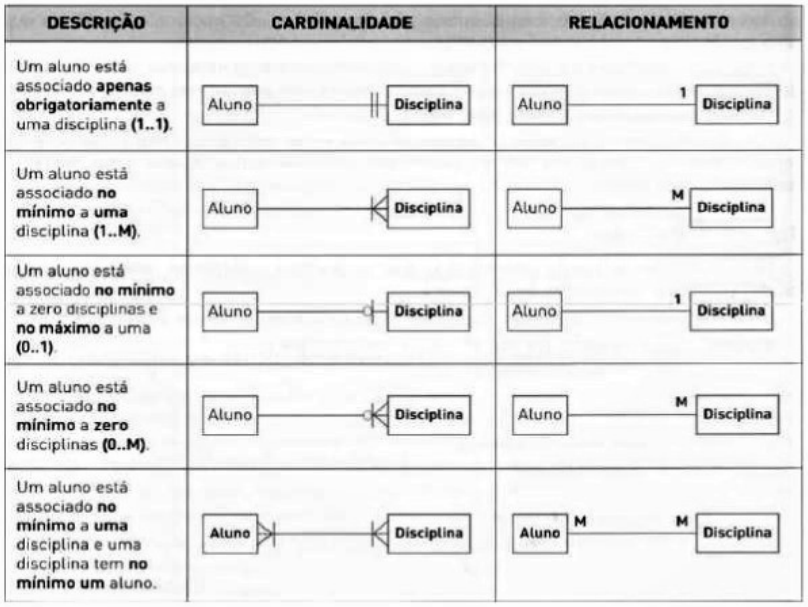


Figura 2.5: Exemplo de diagrama de entidade-relação [10]

2.3 Base de dados

"Uma Base de Dados (BD) é uma coleção de dados que, tipicamente, descreve as atividades de uma ou mais organizações relacionadas"[11]. Utilizando uma BD é possível realizar o armazenamento de informações diversas e relaciona-las entre si, dentro de um sistema.

Esse armazenamento ocorre na forma de tabelas, que são compostas por linhas e colunas. As colunas separam os dados por categorias e cada linha representa um registo formado por uma lista ordenada de colunas.

O utilizador possui de forma organizada acesso aos dados, e pode realizar uma série de operações, afim de extrair informações, tais como [12]:

- Adicionar novos campos à BD;
- Inserir dados em campos existentes;
- Extrair dados de campos existentes;
- Excluir dados de campos existentes;
- Alterar dados de campos existentes;
- Remover campos existentes à BD.

A BD de um armazém de uma universidade, por exemplo, pode conter as seguintes informações:

- *Entidades* como alunos, professores, funcionários e equipamentos
- *Relacionamentos* entre as entidades, como empréstimo dos equipamentos por alunos e professores, realizado pelos funcionários. Devolução dos equipamentos emprestados.

2.4 Gestão de dados

Existem diversos tipos de **Sistemas de Gestão de Base de Dados (SGBD)** em uso, porém os **Sistemas de Gestão de Base de Dados Relacional (SGBDR)** são os mais dominantes atualmente [11].

2.4.1 Sistemas de Gestão de Base de Dados

São sistemas compostos por diversos programas, projetado para auxiliar na manutenção, criação, pesquisa, atualização e administração de vastos conjuntos de dados [11][12][13].

Algumas das vantagens na utilização de SGBD, são [11]:

- Independência de dados;
- Acesso eficiente aos dados;
- Integridade e segurança dos dados;
- Administração de dados;
- Acesso concorrente e recuperação de falhas;
- Redução no tempo para desenvolvimento de aplicações.

Temos no mercado uma vasta gama de oferta de SGBD, dentre eles posso citar alguns como o MySQL, MongoDB, MariaDB, Microsoft SQL Server e o Oracle. Geralmente um SGBD não é compatível com outro, porém é possível realizar interoperações entre eles utilizando as linguagens padrões, tais como, o Structured Query Language (SQL) (secção 2.5), Open Database Connectivity (ODBC) e o Java Database Connectivity (JDBC).

2.4.2 Sistemas de Gestão de Base de Dados Relacional

Baseado no modelo relacional inventado por Edgar Frank Codd em 1970, esse sistema consiste coleção de uma ou mais relações, em que cada relação

é uma tabela, compostas por linhas e colunas [11].

O principal construtor para representar dados no modelo relacional é a relação. Uma relação consiste em um **esquema de relação** e em uma **instância de relação** [11].

Esquema de relação: É descrito pelas colunas (ou campo ou atributo) e pelo domínio (descrição dos valores de cada atributo).

Instância de relação: É conjunto de tuplas (ou registos), no qual cada tupla tem o mesmo número de campos que o esquema de relação. Associando-se a uma tabela as tuplas são as linhas.

Atributos únicos podem atuar como chaves, que identificam de forma única cada registo. Uma chave utilizada para identificar determinada linha em uma tabela é chamada de chave primária. Através de uma chave primária é possível realizar o relacionamento de dados entre duas ou mais tabelas

2.5 Linguagem de Consulta Estruturada

A linguagem de consulta estruturada em inglês Structured Query Language (SQL) foi originalmente desenvolvida, como linguagem de consulta para SGBDR e acabou se tornando a linguagem mais comum utilizada para se comunicar com a base de dados relacionais [11]. Esta linguagem possui comandos que permitem armazenar, manipular, deletar e recuperar dados em base de dados.

Não é possível comunicar todas as base de dados com SQL, geralmente a comunicação ocorre apenas com aquelas que utilizam o modelo relacional. Para recuperar informações de uma base de dados, é necessário escrever um pedaço de código de consulta chamado *query*. A seguir é apresentado algumas das principais operações de *query* realizadas pelo SQL, são elas selecionar, atualizar, criar e deletar dados.

Quando se deseja selecionar dados, o comando `SELECT` é utilizado.

```
SELECT *
FROM equipamento;
```

Código 2.1: Código para seleção de dados em SQL

No código 4.1 o comando `SELECT` seleciona todos ('*') os atributos da tabela equipamento.

Para atualizar dados em uma tabela, faz-se uso do comando `UPDATE`.

```
UPDATE equipamento
SET status = 'Disponivel'
WHERE equipamento.uid = '8E7C90AC';
```

Código 2.2: Código para atualização de dados utilizando SQL

No código 2.2 o valor correspondente ao atributo *status* tem seu valor atualizado para Disponível, porém isto ocorre somente para valores onde o *uid* do equipamento é igual a `8E7C90AC`.

Para inserção de dados em uma tabela utiliza-se os comandos `INSERT` e `VALUES`.

```
INSERT
INTO emprestimo (id_emprestimo,
data_emprestimo,
uid_funcionario,
uid_usuario,
uid_equipamento,
status)
VALUES (NULL,
CURRENT_TIMESTAMP,
'04EF07920F4A84',
'0468FEAA0F4A80',
```

```
    '8E7C90AB',
13  'Emprestimo');
```

Código 2.3: Inserção de dados utilizando SQL

Após a execução do código 2.3, os valores descritos na tabela 2.1 serão inseridos a tabela empréstimo da base de dados:

Tabela 2.1: Valores com os quais cada campo da da tabela empréstimo será populado

CAMPO	VALOR
id_emprestimo	NULL
data_emprestimo	CURRENT_TIMESTAMP
uid_emprestimo	04EF07920F4A84
uid_usuario	0468FEAA0F4A80
uid_equipamento	8E7C90AC
status	Emprestimo

No código 2.3, para cada inserção de valores a chave primária *id_ emprestimo* é auto-incrementada. Neste caso, nenhum valor foi inserido (`NULL`), logo a base de dados irá realizar a tarefa de auto-incremento. O atributo *data_ emprestimo* recebeu o valor `CURRENT_TIMESTAMP`, quando esse valor é passado, a base de dados realiza o preenchimento com horário local.

O comando `DELETE` é utilizado para remover valores da base de dados.

```
1  DELETE FROM emprestimo
   WHERE emprestimo.id_emprestimo = 51;
```

Código 2.4: Remoção de dados utilizando SQL

Após execução do código 2.4 o campo cujo *id_ emprestimo* tenha valor igual a `51` é removido da tabela empresto da base de dados.

2.6 Identificação por radiofrequência

Identificação por radiofrequência em inglês Radio-Frequency Identification (RFID) é uma tecnologia de identificação automática sem contato físico com os objetos. Através da emissão de sinais de rádio é capazes de identificar e acessar informações contidas em cartões. Possui inúmeras áreas de aplicação, entretanto, na logística vem experienciando uma demanda crescente [14].

Sistemas que dispõem de RFID para controlo de inventário acessam informações em tempo real [15]. Desta forma pesquisas, comunicações e análises de decisões são realizadas de forma rápida, com redução de intervenção humana e eliminação de erros [15].

Melhores informações em um inventário proporcionam diversos benefícios, tais como, saber quantos e quais os equipamentos estão armazenados, identificação do utilizador e o período de utilização. Além do mais é possível rastrear os equipamentos em manutenção, e gerar relatórios.

Cartões RFID possuem um número de identificação única, em inglês Unique Identifier (UID). A leitura do cartão pelo leitor RFID elimina a necessidade de um funcionário para anotar ou escanear cada código de barras, isso reduz na quantidade de trabalho e diminui as chances de erro nas leituras, há ainda a vantagem de que inúmeros cartões possam ser lidos simultaneamente (dependendo da antena). Como o fluxo de informações é imediato, os dados em um sistema podem ser automaticamente atualizadas, provendo maior controlo e segurança dos itens armazenados [15].

2.7 Sistema RFID em termos de *hardware*

Um sistema RFID é constituído por três componentes básicos. São eles o leitor, antena e *tags*. A função do leitor é emitir pulsos de sinais eletromagnéticos que serão distribuídos pela antena. Dentro de uma determinada zona

estes sinais ativam o Circuito Integrado (CI) presente nas *tags* e realizam a leitura de seu conteúdo [16].

Uma *tag* RFID pode ser divida em duas partes. O CI, e a antena. O CI é um microcomputador que armazena uma série de números (cada CI tem em sí um único número de identificação) e também um sistema lógico que identifica quando esta na presença de um leitor. Para que a comunicação possa ocorrer tanto as *tags* quantos os leitores necessitam operar na mesma frequência [15].

Na figura 2.6 é possível verificar a forma de funcionamento de um sistema RFID.

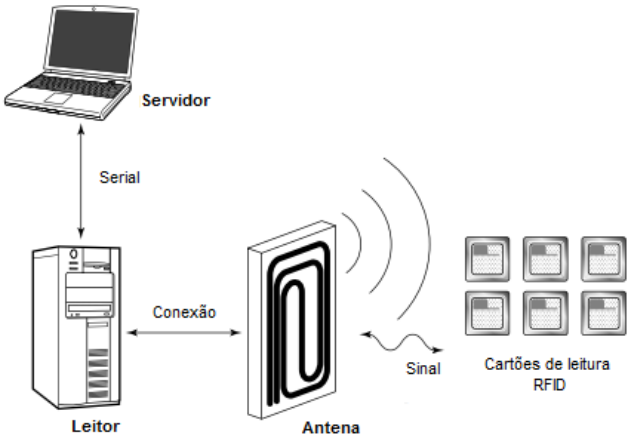


Figura 2.6: Funcionamento do sistema [15]

2.7.1 Leitor RFID

Existem inúmeros leitores RFID, que operam em diversas frequências. Neste projeto entretanto foi utilizado o módulo MF-RC522 (Figura 2.7), capaz de ler e escrever em cartões MIFARE Classic (Anexo B), comunica-se utilizando um campo eletromagnético de 13.56MHz, especificado na ISO 14443 [17][18]. Esta norma define um padrão internacional à cartões de proximidade utilizados para identificação, através dela também é possível identificar erros de detecção, pois possui protocolos específicos para transmissão

e comunicação [19].

Figura 2.7: Leitor e gravador de RFID - MF-RC522

2.7.2 *Tags* RFID

Tags RFID são dispositivos instalados fisicamente nos objetos, após a instalação esses podem ser identificados perante os leitores RFID através de uma resposta ao sinal emitido pelo leitor. Neste documento são apresentadas as características de duas *tags*, as ativas e passivas.

2.7.2.1 *Tags* ativas

As *tags* ativas são alimentadas por pequenas baterias, assim é possível realizar sua leitura a distâncias muito maiores se comparadas às *tags* passivas, entretanto possuem um custo mais elevado e para mante-las em funcionamento é necessário realizar a troca periódica das baterias (caso isso não se possa fazer é necessário descarta-las e substituir por outras *tags*), desta forma possuem gastos constantes sendo necessário avaliar se seus custos de implantação e manutenção trazem os benefícios desejados.

2.7.2.2 *Tags* passivas

Tags passivas são aquelas que quando a fonte de energia que as faz operar provem do exterior, normalmente extraída do campo eletromagnético irradiado pelo leitor. Sua distância de leitura é menor se comparada as *tags* ativas, porém possuem uma vida útil muito maior e custos para implantação mais baixos.

2.8 Arduino

Arduino não somente é uma ferramenta, mas também uma comunidade e forma de pensar que tem influenciando a forma de utilizar e entender a tecnologia [20].

De acordo com a página oficial da Arduino "Arduino é uma plataforma eletrônica, de código aberto baseada em *hardware* e *software* de fácil utilização "[21]. Existe uma vasta variedade de placas, módulos, *shields* (elementos que podem ser conectados a uma placa para fornecer recursos extras) e kits Arduino. Devido a grande quantidade de opções, é possível testar diferentes alternativas e criar novas soluções de forma flexível, prática e rápida.

A plataforma Arduino é *Open Source*, ou seja, qualquer um tem acesso à todo trabalho e pode utilizá-lo sem precisar pagar qualquer tipo de taxa ou *royalties* [22].

2.8.1 Comunidade Arduino

O Arduino conta com uma enorme comunidade, repleta de pessoas criativas, que providenciam suporte e estão sempre em busca de constante desenvolvimento e evolução. Porque qualquer um pode criar um código para o Arduino e compartilhá-lo *on-line*, a comunidade vem crescendo mais a cada dia [23].

2.8.2 *Hardware* - Arduino UNO

O Arduino UNO é uma plataforma de programação e prototipagem baseada no microprocessador ATmega328P [24]. A placa (Figura 2.8) contém uma conexão USB, que torna possível a ligação com o computador. Além disso, contém 14 entradas/saídas digital (da qual 6 delas podem ser utilizadas para saída de sinal PWM), 6 entradas para sinais analógicos, um cristal 16 MHz de quartzo, entrada para alimentação externa (6-20V), *header* ICSP e um botão de reinicialização [24][25]. Pode operar energizado tanto pelo computador através do cabo USB, quanto por uma bateria de 9V ou outra fonte de alimentação [25].

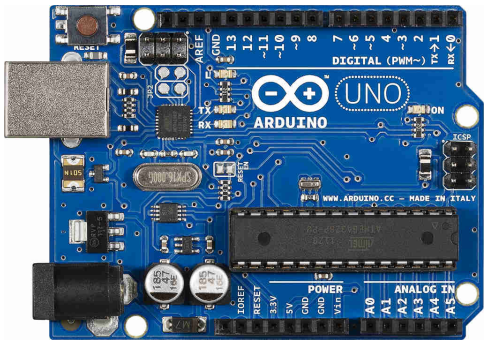


Figura 2.8: Placa Arduino UNO [24]

2.8.3 *Software* - Arduino IDE

A programação da placa é realizada através do envio de um conjunto de instruções ao microcontrolador. Para isso, é necessário o uso da linguagem de programação Arduino e do software Arduino (IDE) [26].

O ambiente de desenvolvimento integrado do Arduino, em inglês Integrated Development Environment (IDE) Arduino , é uma plataforma de código aberto, que permite escrever o código e enviá-lo para a placa. Ele é executado no Windows, Mac OS X e Linux [25]. O ambiente é escrito em Java, baseado no ambiente de programação Processing e outros softwares de código aberto. Este software pode ser usado com qualquer placa Arduino [27].

O IDE do Arduino (Figura 2.9) contém um editor de texto para escrever código, uma área de mensagem, um console de texto, uma barra de ferramentas com botões para funções comuns e uma série de menus.

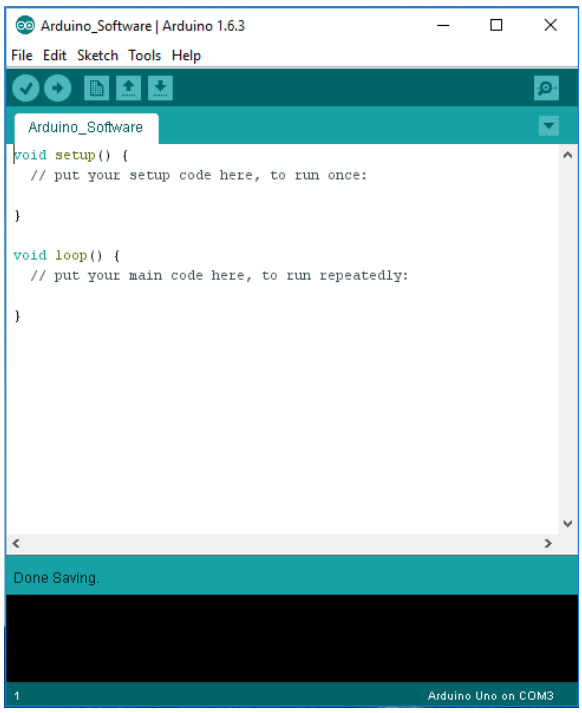


Figura 2.9: Tela inicial do software Arduino IDE

Através da comunicação serial, é possível conectar-se ao hardware para realizar o envio dos programas desenvolvidos. O ATmega328 presente no Arduino UNO já vem pré-programado com o bootloader, isso permite o recebimento do novo código sem a necessidade de nenhum programa de hardware externo.

2.9 Linguagem de Marcação de Hipertexto

A linguagem de marcação de hipertexto em inglês HyperText Markup Language (HTML), fornece a estrutura essencial para as páginas da *web* e sua utilização é necessária para criação de sites.

HTML é uma linguagem de marcação, não de programação. Ele é um arquivo de texto com diversas *tags*, utilizadas para definir a estrutura de uma página *web* [28]. As *tags* marcam o conteúdo a fim de especifica-lo e relaciona-lo com outros conteúdos da página.

O HTML foi originalmente criado para documentos de pesquisa, de modo que estilos padrões foram adicionados para tornar o texto mais fácil de ler [29]. Os títulos são em negrito e ficam menores dependendo do nível de título usado, para mostrar visualmente a hierarquia. Os parágrafos e cabeçalhos também têm espaços entre os elementos para facilitar a legibilidade. Hyperlinks são azuis e sublinhados, e as listas incluem automaticamente marcadores ou números. Porém os estilos são básico e padronizados, para criação diferentes formas de apresentação é necessário a utilização de Cascading Style Sheets (CSS) [30].

2.10 Folhas de estilos em cascata

Folhas de estilos em cascata do inglês Cascading Style Sheets (CSS), é o padrão W3C para definir a apresentação de documentos escritos em HTML.

Apresentação é a maneira como o documento é exibido, logo é utilizado para criar efeitos e estilos na pagina *web* e controlar a apresentação em meios de comunicação diferente (portátil, telemóvel e impressão) [28].

2.11 Linguagem de programação JavaScript

JavaScript é a linguagem de programação da *web*, inventada para a *web*, para adicionar comportamento e interatividade às páginas *web*.

É muito comum ver o JavaScript referido como uma das três principais linguagens de programação *web*, as outras duas (abordadas anteriormente) é o HTML (secção 2.9), que fornece a estrutura da pagina e o CSS (secção 2.10) por gerar uma melhor apresentação. O JavaScript entretanto é responsável

por adicionar comportamento e interatividade as páginas [31].

Javascript é uma linguagem de script, ou seja, através dele não é possível escrever um aplicativo de *desktop*, da mesma forma que se pode fazer utilizando C ++ ou Java. Ele funciona apenas dentro de outro aplicativo, como por exemplo um navegador de *web*. Os navegadores possuem ferramentas internas para interpretar o JavaScript. Em resumo, o sistema operacional contém o navegador que por sua vez abre uma pagina *web* que pode possuir em sí um código JavaScript. Essa característica o limita, pois não tem acesso ao sistema de arquivos do computador na qual ele está sendo processado, palavras para abrir ou salvar arquivos não existem, elas poderiam gerar riscos de segurança, não há palavras para comunicação com uma base de dados ou com a porta USB [31].

Quando um utilizador em seu navegador e solicita uma página de um site, o servidor enviara HTML e CSS como texto, sem formatação, o navegador que irá cuidar da interpretação e renderização de sua melhor forma, seja em um portátil, telemóvel ou computador. O mesmo ocorre com o JavaScript.

O JavaScript, é uma linguagem que trabalha no lado do cliente, ou seja, ela é enviada par o computador do utilizador e lá processada. Diferente do PHP, onde é primeiro executada no servidor e somente então o resultado será enviado ao cliente.

Como foi desenvolvido para ser uma linguagem *sript*, o JavaScript foi feito para ser executado apenas em navegadores *web*, porém no decorrer dos anos, a linguagem foi ganhando força e popularidade, atualmente aplicações e produtos no lado do servidor utilizam JavaScript, é o caso de programas tais como o Adobe Acrobat e Photoshop, servidores Node.js e Google Apps Sript.

2.11.1 jQuery

JQuery é uma das bibliotecas de JavaScript mais populares utilizadas [31]. O JQuery torna todos os tipos de cenários comuns no desenvolvimento de sites e aplicativos muito mais fácil. Ele fornece uma maneira padronizada e simplificada de extrair, manipular e criar conteúdo de página da *web*. Operações como recuperar dados via Asynchronous JavaScript and XML (AJAX) torna-se incrivelmente simples [31].

2.11.2 Javascript Assíncrono e XML

Quando um navegador solicita uma página de um site, o navegador está atuando como um cliente. O *site* é entregue através de uma máquina que serve as informações, por isso chamada de servidor. Quando uma página é solicitada , o cliente faz uma solicitação para o servidor e o servidor retorna uma página. Porém às vezes, queremos pedir informações adicionais as fornecidas [32].

Sem AJAX é necessário fazer um pedido de volta para o servidor, que iria retornar uma nova página inteira. Na maioria dos casos isso significa que o servidor acaba enviando informações redundantes. Com o AJAX, o servidor não precisa enviar a página inteira, apenas envia o que foi requisitado.

AJAX é um grupo de tecnologias e significa, Asynchronous JavaScript and XML (AJAX) (JavaScript assíncrono e eXtensible Markup Language (XML)). A letra A vem de assíncrono e significando que o cliente pode solicitar novas informações ao servidor a qualquer momento, sem precisar esperar a página recarregar. Uma nova solicitação pode ser acionada por meio de diversos eventos, tais como clicar em um botão, sobre uma imagem, escrever em um determinado campo, dentre vários outros. O JavaScript trabalha com os eventos que acionam um novo pedido, realiza pedidos de novos dados para o servidor e cuida de atualizar apenas a parte do documento que precisa ser alterado [31].

O JavaScript se comunica com o servidor através de um conjunto de métodos de programação chamados de Application Program Interface (API) e usa o que é chamado de solicitação XMLHttpRequest (XHR). A API XHR permite ao navegador enviar e solicitar dados a um servidor. Os dados transferidos para e do servidor podem possuir diferentes formatos, não necessariamente XML, mas também arquivos de texto, HTML ou um objeto JavaScript, como o JavaScript Object Notation (JSON).

2.12 Node.js

Ryan Dahl criador do *Node.js*, afirma que "o objetivo do Node é fornecer uma maneira fácil para se criar programas de rede escaláveis"[33].

O Node.js supera o PHP em desempenho computacional, tornando-se mais eficiente em termos de memória e utilizando todo o poder de processamento do servidor [34].

Node.js é uma aplicação em tempo de execução, construída sob a mesma plataforma JavaScript do navegador Google Chrome. Ele permite construir aplicações e páginas *web* com JavaSpript no lado do servidor. A programação do Node.js, diferente da maioria dos servidores, é orientada a evento [33]. Além disso o Node.js é não bloqueante, por possuir uma característica bastante interessante de não impedir com que outras coisas ocorram enquanto executa uma tarefa. A união destas característica o tornam extremamente eficaz, permitindo a realização de operações de **Entrada Saída (E/S)** rapidamente.

Operações de E/S: São operações de leitura e escrita em sistemas de arquivos, redes e processos.

Operações de cálculo: São operações realizadas rapidamente pela Central Processing Unit (CPU)

Um servidor despende maior parte do seu tempo realizando operações de E/S. Servidores que possuem muitos utilizadores conectados à sua rede, não podem ficar com tudo parado à espera da conclusão destas operações. A solução para isso são os servidores não bloqueante (asíncronos) [35]. Diferente dos tradicionais (síncronos), eles funcionam permitindo com que ações não dependentes de outras operações, fiquem livres para trabalhar.

E/S tradicional (síncrona): Operações não dependentes do resultado ficam impossibilitadas de realizar qualquer ação, até que o resultado seja obtido, o código 2.5, mostra como tal programação ocorre, e seus resultados

```
var resultado = db.query("select x from table_Y");
facaAlgumaCoisaCom(resultado);  //Espera pelo resultado!
facaAlgumaCoisaSemOResultado(); //Execucaoo e bloqueada!

```

Código 2.5: Demonstração de uma operação de E/S tradicional (síncrona)

A figura 2.10, exemplifica como o processo de E/S síncrona ocorre.

Figura 2.10: Exemplificação de um processo de E/S síncrona [35]

E/S não bloqueante (assíncrona): Operações não dependentes do seu resultado podem ser executadas paralelamente, como exemplificado através do código 2.6.

```
1   db.query("select x from table_Y",function (resultado){
    facaAlgumaCoisaCom(resultado);
3   //Espera pelo resultado!
    });
5   facaAlgumaCoisaSemOResultado();
    //E executada sem nemhum atraso!
7
```

Código 2.6: Demonstração de uma operação de E/S não bloqueante (assíncrona)

A figura 2.11, exemplifica como o processo de E/S assíncrona ocorre.

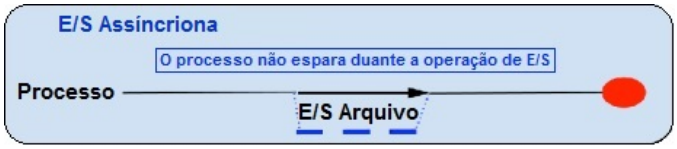


Figura 2.11: Exemplificação de um processo de E/S assíncrona [35]

O Node.js é super leve, e acaba não realizando muitas coisas por sí só. Necessita de diferentes módulos (*frameworks*) adicionados ao seu núcleo para que se torne realmente útil [36]. Embora esses módulos não sejam tecnicamente o próprio Node, eles são extremamente importantes [36].

2.13 Módulos

O sistema de módulos no Node facilita a criação de extensões para a plataforma, e permite compartilhar facilmente códigos e reutilizar bibliotecas [36]. O repositório Node Package Manager (NPM) contém diversos pacotes de códigos disponibilizados por desenvolvedores para que outros utilizadores possam utilizar [37]. Além disso o repositório conta com diversos exemplos, e documentos que auxiliam na utilização dos módulos.

2.13.1 Express

Express é um modulo bastante popular que adiciona funcionalidades de servidor ao Node.js [36]. Tornando mais fácil a criação de páginas *web* e aplicações. Ele cria um mecanismo de roteamento fazendo com que as aplicações possam passar por diferentes tipos de requisições facilmente. O express permite a inserção de outros módulos junto a sí para execução de tarefas mais especificas, ferramentas com essa capacidade são chamadas de *middleware* [38].

2.13.2 Socket.IO

O Socket.IO, permite em tempo real a comunicação bidirecional (entre o cliente e servidor). Isso é possível, pois possui uma biblioteca tanto no navegador do cliente quanto no servidor Node.js [36]. A transmissão para múltiplos *sockets* e o armazenamento de dados associados a cada cliente ocorre de forma assíncrona.

Capítulo 3

Projeto e desenvolvimento

Neste capítulo, são descritos os detalhes para desenvolvimento do projeto. A estrutura da base de dados após a realização da análise de requisitos e planeamento. A configuração do hardware e partes essenciais do código. Por fim, faz-se uma referência aos testes realizados no sistema.

3.1 Arquitetura

Este sistema baseia-se numa arquitetura cliente-servidor. A aplicação foi desenvolvia em Node.js/JavaScript, as informações para o funcionamento encontram-se numa base de dados MySQL. Para que o utilizador aceda à aplicação, é necessário uma navegador *web*. Para o correto funcionamento do servidor é essencial a comunicação com a base de dados e com o leitor de cartões RFID.

3.2 Casos de uso

Após a análise de requisitos, o estudo da implementação do sistema passou pela identificação dos diversos casos de usos para cada usurário:

- Realização do empréstimo de equipamentos (pode-se emprestar quan-

tos equipamentos desejar, não é necessário devolver todos os equipamentos para realização de novos empréstimos);

- Devolução de equipamentos (não necessita ser realizada na mesma ordem dos empréstimo);
- Autorizar o envio de equipamento para manutenções;
- Retornar equipamentos da manutenção;
- Consultar os equipamentos e utilizadores cadastrados no sistema;
- Consultar empréstimos, devoluções e manutenções;
- Imprimir relatórios;
- Adicionar/Excluir equipamentos.

O diagrama de casos de uso (Figura 3.1), mostra os atores do sistema e as principais ações que cada um pode realizar.

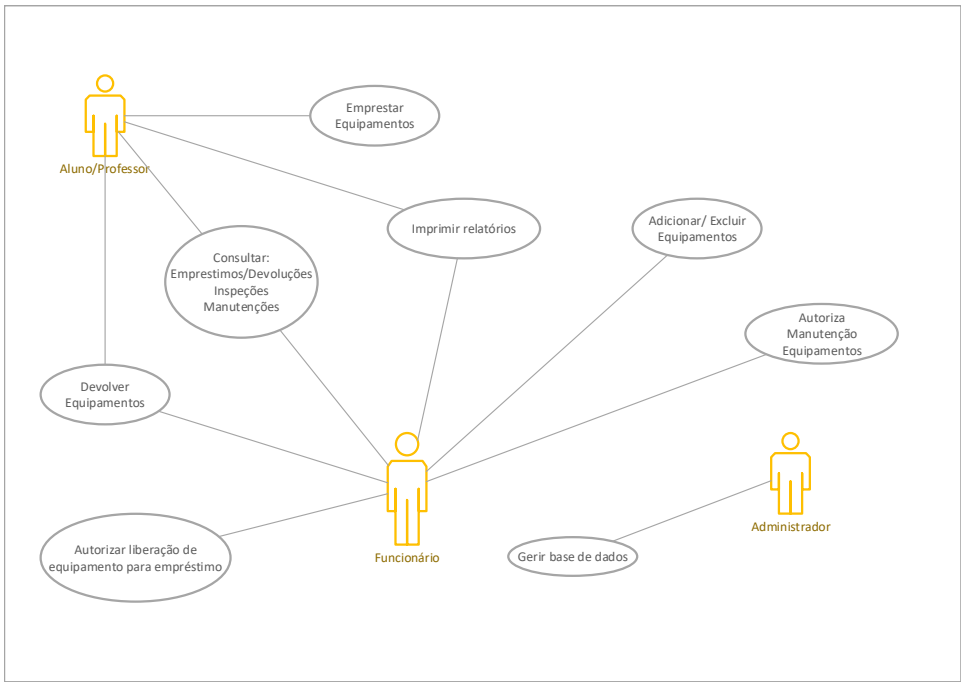


Figura 3.1: Diagrama geral de casos de uso

No Anexo C encontra-se o funcionamento detalhado do sistema, através da descrição pormenorizada dos diferentes casos de uso.

3.3 Modelo entidade relacionamento

Após identificação e análises das necessidades do sistema através do diagrama de casos de uso, foi elaborado o diagrama de entidade relação (Figura 3.2).

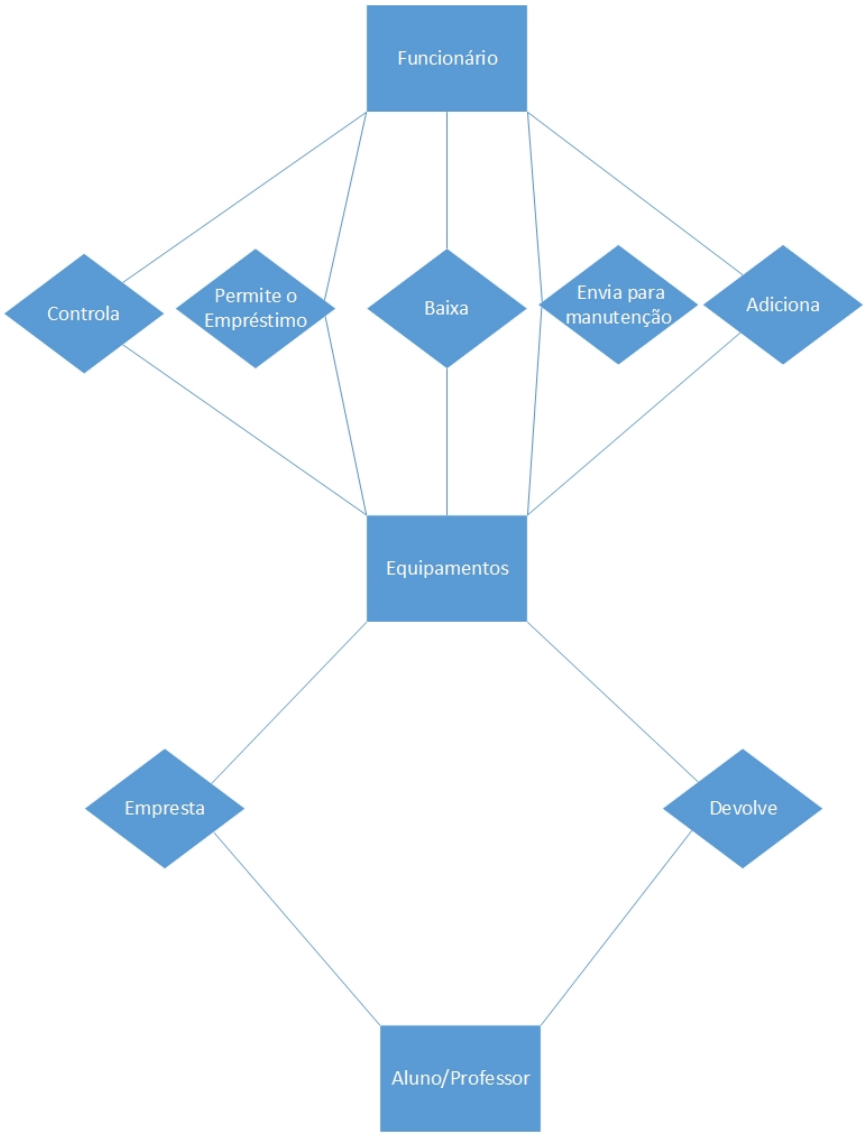


Figura 3.2: Modelo entidade relacionamento

3.4 Modelo da base de dados

Após elaborado o diagrama de entidade relação, criou-se uma base de dados, com as tabelas e relacionamentos mostrados na figura 3.3.

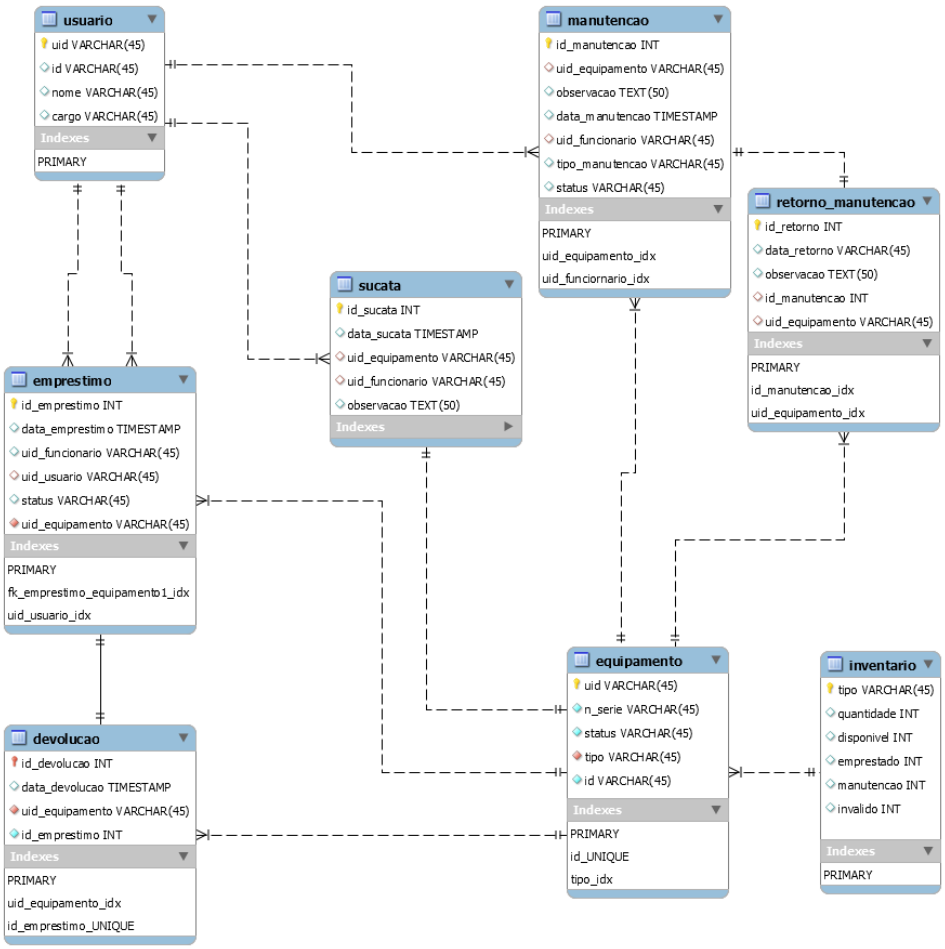


Figura 3.3: Diagrama entidade-relação do sistema

3.5 Ambientes de desenvolvimento

A seguir serão abordados os ambientes de desenvolvimentos utilizados ao longo do projeto.

3.5.1 Ambiente de desenvolvimento da aplicação

Para a programação do sistema fez-se utilização de um editor de texto e código, chamado Atom (Figura 3.4). Esse editor é gratuito e *open-source* [39]. Compatível com MacOS, Linux e Microsoft Windows oferece suporte para a escrita das seguintes linguagens: C/C++, C#, Clojure, CSS, CoffeeScript, GitHub Flavored Markdown, Go, Git, HTML, JavaScript, Java, JSON, Julia, Less, Make, Mustache, Objective-C, PHP, Perl, Property List (Apple), Python, Ruby on Rails, Ruby, Sass, Shell script, SQL, TOML, XML, YAML [40].

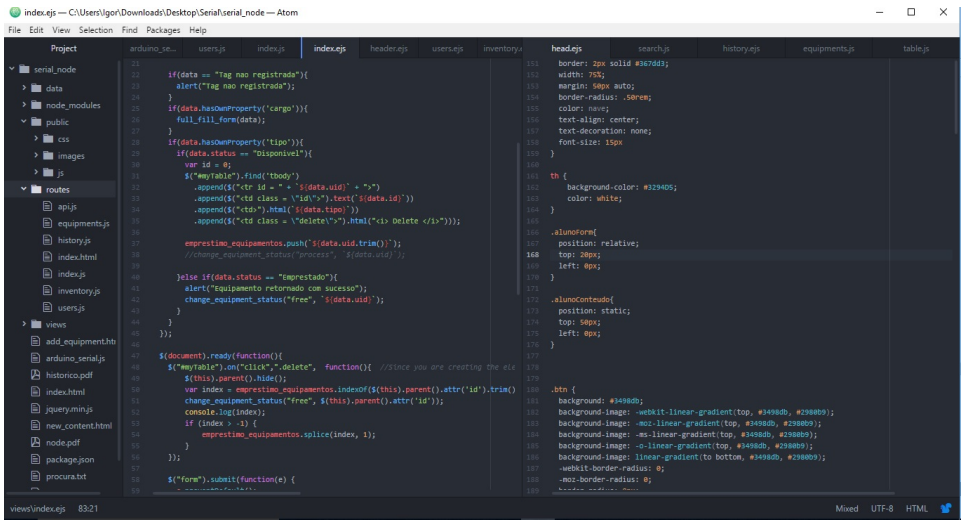


Figura 3.4: Editor de texto e código Atom, com um código aberto

3.5.2 Ambiente de desenvolvimento da base de dados

A aplicação utilizada foi o PhpMyAdmin (Figura 3.5), por ser um sistema simples e de fácil aplicação. Permite a manipulação de um banco de dados MySQL, através de uma interface PHP [41]. Para programação da base de dados no PhpMyAdmin é necessário conexão com um servidor web e uma base de dados.

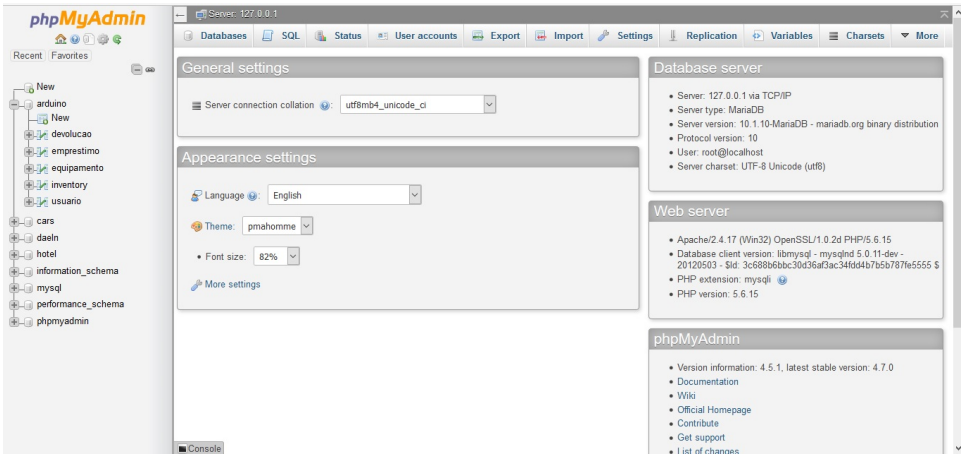


Figura 3.5: Página principal do PhpMyAdmin

3.5.3 Ambiente de inicialização e controlo do servidor

Para abrir uma aplicação node e verificar as mensagens emitidas pelo servidor é necessário uma ferramenta de linha de comando. Para realizar esta tarefa foi utilizado o Microsoft Disk Operating System (MS-DOS) do Windows.

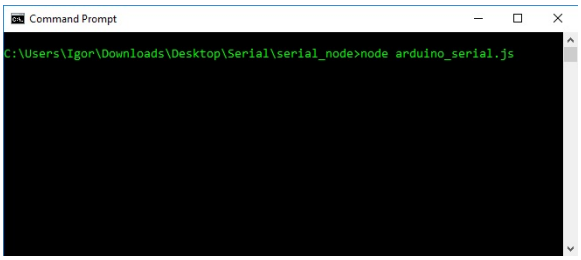


Figura 3.6: Prompt de Comando do Windows (MS-DOS) aberto

3.6 Tabelas da base de dados

A base de dados da aplicação, conta com oito tabelas, todas de alguma forma relacionadas entre sí. Isto possibilita maior rapidez de processamento para acesso as informações o que reduz o número de tabelas necessárias. A seguir estão descritas as tabelas cridas e os campos de cada uma com suas respectivas informações.

Tabela 3.1: Base de dados - Tabela usuário

Campo	**Informação**
uid	*Chave primaria da tabela* UID proveniente da *Tag RFID*
nome	Nome do utilizador
id	ID do utilizador
cargo	Informa o cargo do utilizador

Tabela 3.2: Base de dados - Tabela inventário

Campo	**Informação**
tipo	*Chave primaria da tabela* Informa o equipamento
quantidade	Quantidade total de cada equipamento
disponivel	Quantidade de equipamentos disponíveis
emprestado	Quantidade de equipamentos emprestados
manutencao	Quantidade de equipamentos em manutenção
invalido	Quantidade de equipamentos em inativos

Tabela 3.3: Base de dados - Tabela equipamento

Campo	**Informação**
uid	*Chave primaria da tabela* UID proveniente da *Tag RFID*
id	Número de identificação interno
numero_serie	Número de série do equipamento
tipo	*Foreign key*: {tipo; da tabela inventario}
status	Opções: {Disponível, Processo, Emprestado}

Tabela 3.4: Base de dados - Tabela empréstimo

Campo	**Informação**
id_emprestimo	*Chave primaria da tabela* ID auto-incrementa a cada empréstimo
data_emprestimo	Quando o empréstimo foi efetuado
uid_funcionario	*Foreign key*: {uid; da tabela Funcionario}
uid_usuario	*Foreign key*: {uid; da tabela Aluno ou Professor}
uid_equipamento	*Foreign key*: {uid; da tabela Equipamento}
status	Opções: {Emprestado, Retorno}

Tabela 3.5: Base de dados - Tabela manutenção

Campo	**Informação**
id_manutencao	*Chave primaria da tabela* ID auto-incrementa a cada empréstimo
data_manutencao	Quando o empréstimo foi efetuado
uid_funcionario	*Foreign key*: {uid; da tabela Funcionario}
uid_equipamento	*Foreign key*: {uid; da tabela Equipamento}
tipo_manutencao	Opções: {Manutenção preventiva e reparativa}
observacao	Informações antes de ser enviado à manutenção
status	Opções: {Manutenção, Retorno}

Tabela 3.6: Base de dados - Tabela devolução

Campo	**Informação**
id_devolução	*Chave primaria da tabela* ID auto-incrementa a cada devolução
data_devolucao	Quando a devolução foi efetuado
uid_equipamento	*Foreign key*: {uid; da tabela Equipamento}
id_emprestimo	*Foreign key*: {id_emprestimo; da tabela Emprestimo}

Tabela 3.7: Base de dados - Tabela retorno de manutenção

Campo	**Informação**
id_retorno	*Chave primaria da tabela* ID auto-incrementa a cada retorno
data_retorno	Quando o retorno foi efetuado
observacao	Informações a respeito do retorno
uid_equipamento	*Foreign key*: {uid; da tabela Equipamento}
id_manutencao	*Foreign key*: {id_mantencao; da tabela Manutencao}

Tabela 3.8: Base de dados - Tabela sucata

Campo	**Informação**
id_sucata	*Chave primaria da tabela* ID auto-incrementa a cada baixa
data_sucata	Quando a baixa foi efetuada
observacao	Informações a respeito da baixa
uid_equipamento	*Foreign key*: {uid; da tabela Equipamento}
uid_funcionario	*Foreign key*: {uid_funcionario; da tabela Funcionario}

3.7 Comunicação entre o leitor RFID e o Arduino

Para comunicação entre o leitor RFID e o Arduino fez-se uso da interface Periférica Serial do inglês Serial Peripheral Interface (SPI), um protocolo de dados em série síncrono. Utilizado por microcontroladores permite comunicação com um ou mais dispositivos periféricos de forma rápida. Na conexão SPI, sempre há um dispositivo (mestre) em geral um microcontrolador que controla os dispositivos periféricos (escravos). Normalmente, existem três linhas comuns a todos os dispositivos, são elas:

MISO (Master In Slave Out): O dispositivo escravo envia dados ao mestre.

MOSI (Master Out Slave In): O dispositivo mestre envia dados ao escravo.

SCK (Serial Clock): Os pulsos de *clock* (gerado pelo mestre) são utilizados para sincronia de transmissão de dados.

E uma linha específica para cada dispositivo:

SS (Slave Select): Utilizado pelo mestre para habilitar e desabilitar dispositivos específicos.

Quando o seleção de escravo (SS) de um dispositivo esta em baixa, a comunicação com o mestre é realizada. Quando está em alta, o mestre é ignorado. Desta forma é possível compartilhar as mesmas linhas MISO, MOSI e CLK entre diversos dispositivos periféricos, a figura 3.7 exemplifica como é realizada a ligação e fluxo de dados entre um dispositivo SPI mestre e escravo.

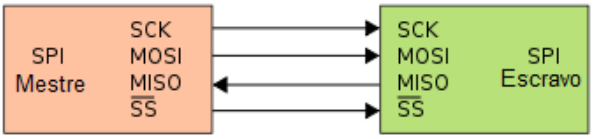


Figura 3.7: Funcionamento da comunicação SPI

A tabela 3.9 associa a ligação realizada entre os pinos do leitor RFID e da placa Arduino UNO para correto funcionamento com o código descrito no Anexo D.

Tabela 3.9: Esquema de conexão

Pinos no Leitor RFID	Pinos no Arduino UNO
SS	Digital 10
SCK	Digital 13
MOSI	Digital 11
MISO	Digital 12
GND	GND
RST	Digital 9
3,3V	3,3V

A figura 3.8 representa graficamente as conexões realizadas (tabela 3.9) entre o Arduino UNO e leitor de cartões RFID MF-RC522.

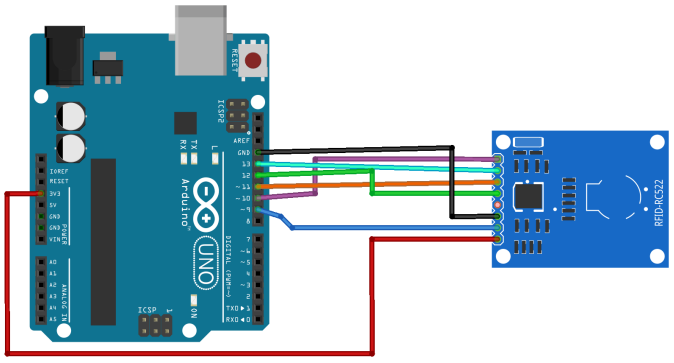


Figura 3.8: Conexão entre a placa Arduino UNO e leitor de cartões RFID MF-RC522

3.8 Comunicação entre o Arduino eo servidor

A comunicação serial foi utilizada para comunicar o Arduino com o servidor (mais detalhes do código na secção 3.9.4), e receber as informações referentes a leitura das *tags* RFID. Foi utilizado o protocolo de Transmissão e Recepção Assíncrona Universal, em inglês Universal Asynchronous Receiver/Transmitter (UART), funciona a base da configuração do mesmo *clock* de operação nos controladores que estão se comunicando, essa comunicação

utiliza apenas dois fios, um de transmissão e outro de recepção. O *baud rate*, define a taxa de dados transmitidos, neste caso foi configurado para ser 9600 bps (linha 21 do código 4.1), a mesma taxa deve ser utilizada no servidor afim de ler os dados que foram enviados corretamente (código 3.5).

3.9 Códigos

3.9.1 Instalando e incluindo módulos no Node

Para instalar um módulo no Node, basta informar o programa gestor de pacotes (os mais utilizados são o *dpkg, rpm e o npm*) seguido da palavra *install* e o nome do módulo desejado na ferramenta de linha de comando, o código 3.1 foi utilizado para instalar o módulo express.

```
npm install express
```

Código 3.1: Instalando o módulo express

Após a instalação do módulo, para inclui-lo no Node é necessário utilizar a função *require()* , o código 3.2, exemplifica como é realizada a inclusão do módulo express ao Node.

```
1 var http = require("express");
```

Código 3.2: Incluindo o módulo express

3.9.2 Inicializando o servidor com o express

O código 3.3, inicializa o servidor com o express. Nele a variável *express*, exige o módulo express (previamente instalada no Node). As funcionalidades do express são armazenadas dentro de uma função, para facilitar o trabalho, uma instancia dessa funções foi criada, e associado a variável *app*.

```
var express = require("express");
var app = express();
```

Código 3.3: Inicialização do Express

3.9.3 Criando rotas no express

Uma pagina web é solicitada através do localizador uniforme de recursos em inglês Uniform Resource Locator (URL). Pode ser realizado através da barra de endereços do navegador, ou por meio de um link clicado na página. O URL tem todas as informações necessárias para acessar um determinado documento em um servidor web.

O navegador envia uma solicitação de protocolo de transferência de hipertexto em ingês Hypertext Transfer Protocol (HTTP) ao servidor mencionado na URL, resgatando um arquivo especifico. O servidor realiza a resgate do arquivo solicitado e emite uma resposta HTTP. Quando a página é encontrada, recupera o arquivo solicitado e devolve ao navegador que decompõem o documento e a pagina é montada na janela do navegador. Caso a página não seja encontrada retorna um erro 404 [28].

O express permite acesso a diversos métodos de solicitação HTTP, tais como o método GET, POST e DELETE [42]. No código 3.4, quando alguém vai a algum local especifico do site (no caso, a página de pesquisa), definido pela localização "/search", uma função é executada, e o utilizador é direcionado para a página desejada [43].

```
router.get("/search", function(req, res){
res.render('search.ejs', {
pageTitle: "Busca"
});
});
```

Código 3.4: Criando rotas no express

3.9.4 Comunicando o servidor com a porta serial

O código 3.5, inicializa a conexão com a porta serial. A variável *serial*, exige o módulo serialport (previamente instalada no Node). A função *mySerial* recebe como parâmetros os dados necessários para comunicação, indicando a porta de comunicação (COM3), o *baud rate*, neste caso de 9600bps e o parser para realizar leitura individual das linhas.

```
var serial = require('serialport');
var mySerial = new serial("COM3", {
    budRate:9600,
    parser: serial.parsers.readline("\n")
});
mySerial.on("open", function(){
    console.log("Porta Aberta");
});
```

Código 3.5: Iniciando conexão com a porta Serial (COM3)

3.9.5 Inicializando o MySQL

O código 3.6, inicializa a conexão com a base de dados. A variável *mysql*, exige o módulo mysql (previamente instalada no Node). A função *creatConnection* recebe como parâmetros os dados necessários para inicialização da conexão com a base de dados. Uma instancia dessa funções foi criada, e associado a variável *connection*. Em seguida o servidor irá emitir uma mensagem informando se a conexão com a base de dados foi bem sucedida ou não. Por fim a conexão é fechada, por meio da função *end()*.

```
var mysql      = require('mysql');

var connection = mysql.createConnection({
host     : 'localhost',
```

```
port     :  3305,
user     : 'root',
password : '',
database : 'arduino'
});

connection.connect(function(err){
if(!err) {
console.log("A base de dados esta conectada ...");
}else {
console.log("Erro ao conectar a base de dados ...");
}
});

connection.end();
```

Código 3.6: Iniciando conexão com o MySQL

3.9.6 Geração de relatórios em PDF

O código 3.7, demonstra como ocorre a geração de arquivos Portable Document Format (PDF). Primeiramente é necessário o módulo PDFKit (previamente instalada no Node) e o FS (File System), este último não necessita instalação, pois já vem junto ao Node. A variável *myDoc* declara um novo objeto pdf. A função *pipe()*, recebe como argumento o nome com o qual deseja-se salvar o arquivo PDF, e o seu diretório. Em seguida define-se a fonte que será utilizada e seu tamanho. Para inserir novas linhas te texto faz-se uso da função *text()*. Para encerar a escrita do documento é necessário finaliza-lo com a função *end()*

```

var pdf = require('pdfkit');
var  fs = require('fs');
var myDoc = new pdf;
```

```
38

40 myDoc.pipe(fs.createWriteStream(
   "Relatorios/Manutencao/Manutencao.pdf"));
42 myDoc.font('Times-Roman')
       .fontSize(32)
44     .text('Relatorio de Manutencao', 100, 100)
       .fontSize(12)
46     .text('Relatorio gerado: ' + dateFormat(now))

48 myDoc.end();
```

Código 3.7: Criando um arquivo PDF

3.9.7 Inicializando uma aplicação Node.js

Para inicialização de um aplicação node, digitar no *prompt* de comando do windows (diretório que contem todos os arquivos) a palavra node, seguida pelo nome do arquivo onde foi realizada a inicialização do servidor express

3.9.8 Lista de equipamentos

Quando deseja-se selecionar um tipo de equipamento, a aplicação fornece uma lista com todos os equipamentos presentes no inventário. Esta lista provém de um arquivo JSON gerado através da identificação de todos os equipamentos na tabela inventário da base de dados. O código 3.8 mostra como o arquivo é criado.

```
   var pool = req.app.get('pool');
50 pool.query('SELECT `tipo` FROM `inventory`', function(error,
              results, fields){
52                if(`${results}`){
                  all_inventory.push(results);
54            }
```

```
var output = {
    all_inventory: all_inventory[0]
}
res.json(output);

fs.writeFile('./data/dataBase.json', JSON.stringify(output),
             'utf8', function(err){
                 console.log(err);
             });
});
```

Código 3.8: Criação de uma API, com todos os equipamentos do inventário

Na parte do cliente o AJAX utiliza este arquivo para criação da lista, conforme o código 3.9.

```
$.getJSON('http://localhost:3000/api', function(data){
var selectedEquipment = "#equipments" +
    $(".selecao_equipamentos").length;
$.each(data.all_inventory, function (i, item) {
        $(selectedEquipment).append($('<option>', {
            value: item.tipo,
            text : item.tipo
        }));
    });
});
```

Código 3.9: Criação da lista de equipamentos

Faz-se utilização desta lista na página busca (secção 3.10.8) quando se deseja pesquisar algum equipamento e na página adicionar (secção 3.10.1) quando se deseja adicionar um equipamento novo.

3.10 Testes e aplicações

Passos a serem seguidos para inicialização do servidor:

1. Inicializar a base de dados.

2. Inicializar a conexão serial que irá realizar a leitura dos cartões RFID.

3. Digitar no *prompt* de comando referente a pasta principal o código `node arduino_serial.js`.

Após a inicialização do servidor, para que o utilizador possa abrir a página inicial, que dará acesso as ferramentas do sistema, deverá realizar os seguintes procedimentos:

1. Acessar um navegador web.

2. Digitar na barra de enderenços *localhost:* seguido pela porta de acesso (previamente defina como 3000)

Após efetuado esses procedimentos o servidor identifica o pedido de acesso, sendo válido, retorna a página inicial ao navegador. Esta página contém um menu (figura 3.9) com as seguintes opções: Empréstimo/Devolução/Retorno, Usuários, Equipamentos, Relatórios, Inventário, Busca, Adicionar e Manutenção/Baixa.

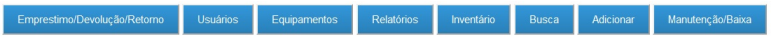


Figura 3.9: Cabeçalho da página inicial

3.10.1 Como adicionar um novo utilizador ou equipamento

Selecionando a opção Adicionar no menu principal, o utilizador do sistema é redirecionado a uma página (figura 3.10) onde poderá registar uma nova UID. Após ser processada a leitura de um novo cartão, ainda não registado na base de dados, a UID é preenchida automaticamente, sendo que não é possível preenchê-la manualmente.

Figura 3.10: Página adiciona

Para evitar recadastramento de cartões, o sistema recusa UID's já cadastradas, emitindo uma mensagem de alerta (Figura 3.11).

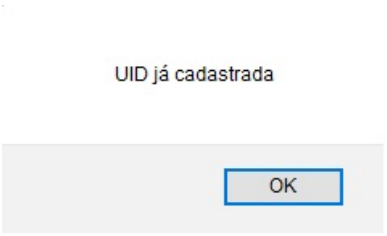


Figura 3.11: Página adiciona - Mensagem de erro quando identificada uma UID já cadastrada.

Após o autopreenchimento do campo UID, o sistema abre as opções para o cadastro de um utilizador ou equipamento em tempo real (Figura 3.12).

Figura 3.12: Página adiciona - Opção para registar um novo utilizador ou equipamento

Clicando-se no campo usuário (figura 3.13), o sistema fornece as opções para o cadastro de um aluno, professor ou funcionário, assim como os campos para preenchimento do nome e id. Para finalizar o cadastro é necessário confirmar através do botão Adicionar.

Figura 3.13: Página adiciona - Cadastrando um novo utilizador

Clicando-se no campo equipamento (figura 3.14), o sistema fornece a relação de equipamentos que podem ser fornecidos pelo armazém e um campo para preenchimento do número do patrimônio. Após optar por um dos itens da relação apresentada pelo sistema e o preenchimento do campo com o número do patrimônio e o número de série é necessário a confirmação através do botão Adicionar para finalizar o cadastro.

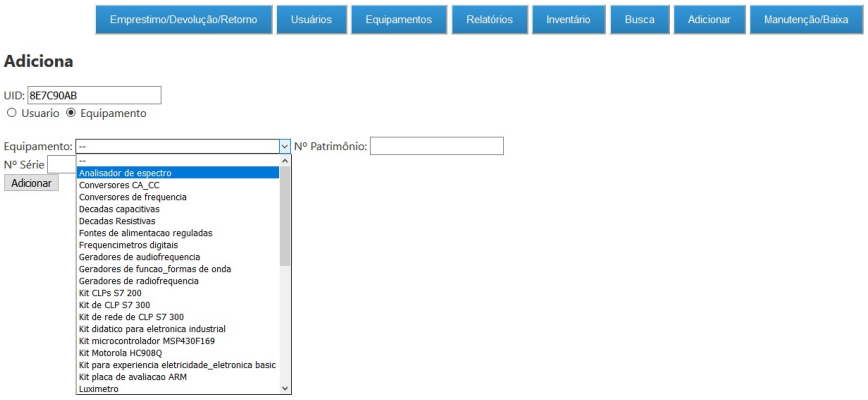


Figura 3.14: Página adiciona - Cadastrando um novo equipamento

3.10.2 Como realizar empréstimos

Selecionando a opção Empréstimos no menu principal, o utilizador do sistema é redirecionado a uma página para preencher o formulário de empréstimo. Conforme situações descritas a seguir:

1. Id e nome, do funcionário (previamente cadastrado a base de dados) responsável pelo empréstimo.

2. Id e nome do o aluno ou professor (previamente cadastrado base de dados) que solitou o material.

3. Identificação dos equipamentos que estão sendo retirados.

4. Opção limpar - É utilizada no caso de haver necessidade de mudar dados referentes ao funcionário, aluno ou professor.

5. Opção deletar (acessivo apenas a equipamentos) - É utilizado para excluir equipamentos no formulário de empréstimo

6. Opção Confirmar - o formulário é enviado ao servidor, onde será processado, estando tudo de acordo será aceito e o empréstimo realizado, ou se houver algum erro de preenchimento, uma mensagem informando que o empréstimo não foi concluído irá aparecer no canto superior esquerdo da página.
   Possíveis erros de preenchimento:

   (a) Falta de identificação do funcionário, aluno ou professor

   (b) Ausência de preenchimento do campo de equipamento para empréstimo

Após todos os campos preenchidos (figura 3.15) é necessário uma confirmação manual, clicando-se no botão confirma na parte inferior do formulário.

Figura 3.15: Página inicial - Todos os campos preenchidos

3.10.3 Visualização de todos os utilizadores cadastrados no sistema

Selecionando a opção Usuários no menu principal, o utilizador do sistema é redirecionado a uma página (figura 3.16) onde é possível visualizar as informações relacionadas ao cadastro, de todos os utilizadores registados no sistema.

Figura 3.16: Página de usuários

3.10.4 Visualização do histórico de cada utilizador

Na página usuários (figura 3.16), clicando-se no número de identificação do utilizador o utilizador é redirecionado a uma página com a opção de acessar o histórico dos empréstimos, pendencias, devoluções, locações e manutenção de cada utilizador.

3.10.5 Visualização de todos os equipamentos cadastrados no sistema

Selecionando a opção Equipamentos no menu principal, o utilizador do sistema é redirecionado a uma página (figura 3.17) onde é possível visualizar a descrição, identificação e a situação de todos os equipamentos registados no sistema.

Clicando-se sobre o campo ID de cada equipamento, o utilizador será direcionado a uma página, com o histórico de manutenções ou empréstimos relativas ao equipamento selecionado.

Para uma melhor visualização, no campo status o sistema apresenta uma diferenciação de cores: verde para disponível, amarelo para emprestado, ver-

melho para manutenção branco para inválido.

Figura 3.17: Página de equipamentos

3.10.6 Gerando um relatório

Selecionando a opção Relatório no menu principal, o utilizador do sistema é redirecionado a uma página (figura 3.18) onde são apresentadas três opções: Relatório de empréstimos, Relatório de manutenções e Relatório de baixas do patrimônio.

Figura 3.18: Página de relatórios

Estas opções permitem ao utilizador acessar para visualização ou impressão, o relatório com todo o histórico de empréstimos, manutenção ou baixa dos equipamentos. Podendo ser aplicado um filtro de data. A figura 3.19 mostra a página relatório de manutenção.

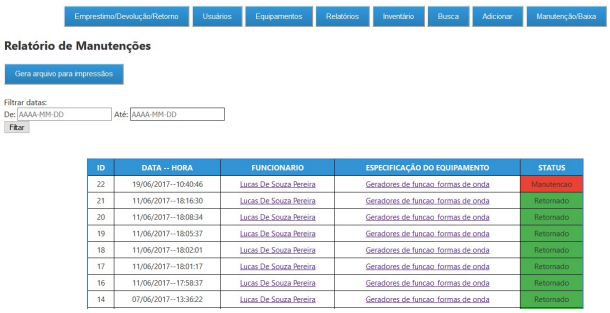
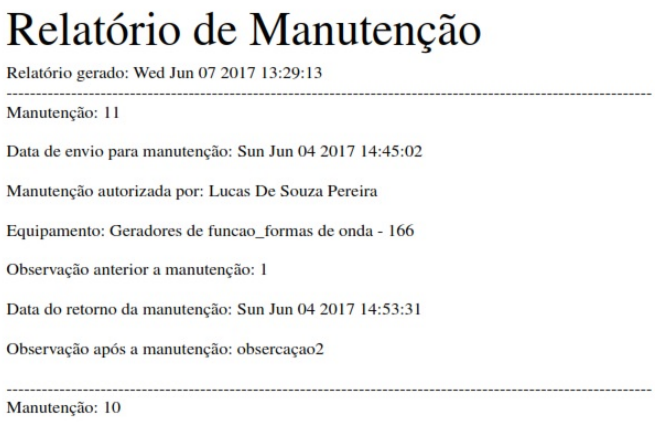


Figura 3.19: Página de relatório de manutenção

Após ser selecionada a opção de gerar arquivo para impressão, o sistema cria um arquivo PDF, que será armazenado na pasta Relatórios, a figura 3.20 mostra o relatório de manutenção.



Relatório de Manutenção

Relatório gerado: Wed Jun 07 2017 13:29:13

Manutenção: 11

Data de envio para manutenção: Sun Jun 04 2017 14:45:02

Manutenção autorizada por: Lucas De Souza Pereira

Equipamento: Geradores de funcao_formas de onda - 166

Observação anterior a manutenção: 1

Data do retorno da manutenção: Sun Jun 04 2017 14:53:31

Observação após a manutenção: obsercaçao2

Manutenção: 10

Figura 3.20: Relatório de manutenção

3.10.7 Visualização do inventário

Selecionando a opção Inventário no menu principal, o utilizador do sistema é redirecionado a uma página (figura 3.21) onde é possível visualizar uma tabela com o status de todos os equipamentos registados no sistema. Nesta tabela, clicando-se sobre a descrição relativa a cada equipamento, o utilizador é direcionado a uma página que contém uma tabela com informações específicas de equipamentos com a descrição selecionada.

Figura 3.21: Página inventário

3.10.8 Realização de pesquisas no sistema

Selecionando a opção Busca no menu principal, o utilizador do sistema é redirecionado a uma página (figura 3.22) onde é possível filtrar informações do histórico de empréstimo. Para realização da pesquisa pode-se utilizar tanto o código de identificação do aluno, professor ou funcionário como também a descrição do equipamento, ou ambos. Após preenchidas as opções, e confirmando no botão buscar, o utilizador visualizará uma tabela com o histórico dos empréstimos referentes ao filtro aplicado.

Figura 3.22: Página de pesquisa

3.10.9 Enviando equipamentos para manutenção ou desativando

Selecionando a opção Manutenção/Baixa no menu principal, o utilizador do sistema é redirecionado a uma página onde é possível iniciar o processo de envio para assistência técnica os equipamentos (com o status disponível) que

necessitem de manutenção. O processo é iniciado com o preenchimento do campo equipamento, realizado através da leitura do cartão de identificação do equipamento a ser enviado.

Após a identificação do equipamento o sistema abre três opções: Manutenção Preventiva, Manutenção Reparativa e Baixa do equipamento.

Capítulo 4

Conclusões

Neste capítulo serão apresentadas as conclusões do projeto, serão também apresentados os objetivos atingidos, contribuições do projeto, análise das limitações e trabalhos futuros. Por fim, a apreciação final do projeto cita as capacidades pessoais desenvolvidas e adquiridas durante a execução dos trabalhos.

4.1 Resumo

O projeto tem como principal objetivo apresentar ao armazém do DAELN da UTFPR um sistema de gestão e controlo, mais eficiente, seguro, versátil e dinâmico. Utilizando ferramentas modernas e tecnologias avançada, facilitando assim a realização de empréstimos, devolução de equipamentos, geração de relatórios, controlo do inventário e das manutenções necessárias.

Durante o desenvolvimento deste sistema foi necessário, uma prévia análise das atuais condições de empréstimos praticadas no armazém, posteriormente identificadas as necessidades e dificuldades enfrentadas, foram criadas adaptações para modernizar a execução das atividades.

Para a implementação do sistema foi necessário a criação de uma base de dados, programação da página *web*, do servidor e integração do *hardware*

com a base de dados.

4.2 Objetivos realizados

A modernização do sistema foi desenvolvida satisfatoriamente, com funções de leitura dos cartões, autopreenchimento dos campos, realização de empréstimos, devoluções, envios para manutenções, baixas, registos e geração de relatórios.

Ao longo do desenvolvimento do projeto o formato das páginas, suas funcionalidade e a maneira de trabalhar com a base de dados sofreram inúmeras alterações, sempre com o objetivo de deixar o aspecto do sistema mais simples e facilitando sua utilização. Com o intuito de facilitar pesquisas foram inseridos filtros e buscas. Finalizou-se com ajustes, verificações, e testes com pleno sucesso.

4.3 Contribuições do projeto

O projeto contribui para a UTFPR no âmbito de aprimorar e modernizar o processo de como os empréstimos são realizados, foi desenvolvido como estudo inicial que poderá ser adaptado e expandido para uso coletivo.

4.4 Limitações e trabalho futuro

Uma das limitações para implementação do sistema deve-se a necessidade da utilização de cartões RFID, pois, atualmente a UTFPR Curitiba-Centro não dispõe deste método de identificação. E será necessário a substituição dos cartões atuais.

Outra limitação é a falta de um estudo da viabilidade econômica para a implantação de *tags* em todos os equipamentos do armazém bem como definir qual sistema de antenas para leitura dos cartões, uma vez que as

antenas com maior alcance seriam as mais indicadas, por proporcionarem um maior controlo, possibilitando a emissão de alertas quando um equipamento é retirado dos limites da área do departamento.

Com a implantação do sistema se faz necessário disponibilizar treinamento aos funcionários, encarregados de orientar alunos e professores, para uma correta utilização e adaptação ao novo sistema.

Para tornar o sistema mais seguro, futuramente poderá ser implantado acessos através de *login* e senha, com esse recurso também poderão ser feitos agendamentos *on-line*.

A implantação de um sistema RFID no armazém poderá motivar a universidade a ampliar esta ideia a outros departamentos, para facilitar controlos e registo de movimentações.

4.5 Apreciação final

O desenvolvimento do sistema ocorreu de forma esperada, apresar das diversas alterações e adaptações no decorrer do projeto. Todos os objetivos propostos foram cumpridos, ademais algumas ferramentas inicialmente não previstas foram incluídas tais como buscas e filtros. Entretanto, durante a implantação e a utilização poderão surgir novos fatos, necessitando de ajustes conforme as finalidade de uso.

A necessidade para resolução dos problemas e desenvolvimento das ideias contribuirão muito para ampliação dos horizontes do autor em diversos aspectos, tais como na área de desenvolvimento web, programação, base de dados, métodos e planeamento organizacional.

A pouca experiencia do autor em desenvolvimento web, ocasionou um trabalho mais demorado, em decorrência da necessidade de adquirir conhecimento em novas linguagens e maneiras para criação de determinadas tarefas.

A expectativa do autor com este projeto é contribuir com a universidade, onde atualmente estuda, proporcionando facilidades para empréstimos dos

materiais, maior segurança e controlo do patrimônio, eliminação dos formulários em papel e também redução dos custos com impressões. Enfim fornecer não somente um sistema, mas também um novo conceito que poderá ser ampliado para as mais diversas áreas dentro da universidade.

Referências Documentais

[1] *De Escola de Aprendizes à Universidade Tecnológica*, URL: `http://www.utfpr.edu.br/a-instituicao/historico`, consultado em Junho de 2017

[2] *UTFPR: inovação e geração de tecnologias*, URL: `http://www.utfpr.edu.br/a-instituicao`, consultado em Junho de 2017

[3] D. F. B. A. d. Castro, *Sistema de Informação para Gestão de Equipamentos* (2016)

[4] R. L. Alcami and C. D. Caranana, *Introduction to Management Information Systems*, http://repositori.uji.es/xmlui/handle/10234/46625?locale-attribute=en (2012), consultado em Junho de 2017

[5] U. Mishra, *Introduction to Management Information system*, CoRR **abs/1308.1797** (2013), URL: `http://arxiv.org/abs/1308.1797`

[6] L. S. Rodrigues, *Arquitecturas dos sistemas de informação* (FCA - Editora de Informática, 2002)

[7] K. I. Satoto, R. R. Isnanto, R. Kridalukmana and K. T. Martono, *Optimizing MySQL database system on information systems research, pu-*

blications and community service, in *2016 3rd International Conference on Information Technology, Computer, and Electrical Engineering (ICITACEE)* (2016), pp. 1–5

[8] T. Quatrani, *Introduction to UML 2.0*,
URL: `http://www.omg.org/news/meetings/workshops/MDA-SOA-WS_Manual/00-T4_Matthews.pdf`, consultado em Junho de 2017

[9] M. Nunes and H. O'Neill, *Fundamental de UML* (FCA (BRASIL)),
URL: `https://books.google.com.br/books?id=ezhcXwAACAAJ`

[10] F. Lopes, M. Morais and A. Carvalho, *Desenvolvimento de sistemas de informação* (FCA (BRASIL), 2015),
URL: `https://books.google.com.br/books?id=pRZkbwAACAAJ`

[11] R. Ramakrishnan and J. Gehrke, *DataBase management systems*, 3 ed. (McGraw-Hill Companies, Inc, Nova York, Estados Unidos, 2008)

[12] C. J. Date, *An introduction to database systems*, 8 ed. (Pearson Education, Inc, Califórnia, Estados Unidos, 2003)

[13] S. Vaze, *Computer Fundamentals and RDBMS* (Himalaya Pub. House, Estados Unidos, 2010)

[14] T. F. Aydos and J. C. E. Ferreira, *RFID-based system for Lean Manufacturing in the context of Internet of Things*, in *2016 IEEE International Conference on Automation Science and Engineering (CASE)* (2016), pp. 1140–1145

[15] D. M. Dobkin, *The RF in RFID: passive UHF RFID in practice* (Elsevier Inc, Massachusetts,Estados Unidos, 2008)

[16] O. Hongzhi, W. Xinlin, Z. Weihua and L. Yuehua, *Design of auto-guard system based on RFID and network*, in *2011 International Conference on Electric Information and Control Engineering* (2011), pp. 1292–1295

[17] *DigiKey - MFRC522*,
URL: `https://goo.gl/9Qy02e`, consultado em Maio de 2017

[18] *MIFARE Classic Family*,
URL: `https://www.mifare.net/en/products/chip-card-ics/mifare-classic/`, consultado em Maio de 2017

[19] *ISO/IEC 14443-1:2016* (2016),
URL: `https://www.iso.org/standard/70170.html`, consultado em Maio de 2017

[20] J. Nussey, *Arduino For Dummies*, –For dummies (Wiley, 2013),
URL: `https://books.google.com.br/books?id=W14KwR0irpcC`

[21] *Arduino.cc*,
URL: `https://www.arduino.cc/`, consultado em Maio de 2017

[22] *Robotizando - O Ambiente de Desenvolvimento Integrado*,
URL: `https://goo.gl/mPu00r`, consultado em Maio de 2017

[23] B. Craft, *Arduino Projects For Dummies* (John Wiley & Sons, Ltd, West Sussex, Inglaterra, 2013)

[24] *ArduinoBoardUno*,
URL: `https://www.arduino.cc/en/Main/ArduinoBoardUno`, consultado em Maio de 2017

[25] S. Monk, *Arduino Android Projects for the Evil Genius: Control Arduino with Your Smartphone or Tablet* (McGraw-Hill Companies, Inc, Nova York, Estados Unidos, 2011)

[26] *Arduino Software (IDE)*,
URL: `https://www.arduino.cc/en/Guide/Environment`, consultado em Maio de 2017

[27] *Hardware Livre USP - O Ambiente de Desenvolvimento*,
URL: https://goo.gl/mPu00r, consultado em Maio de 2017

[28] R. J. Niederst, *Learning Web Design - A Beginner's Guide to (X)HTML, Style Sheets, and Web Graphics*, 3 ed. (O'Reilly Media, Inc, Califórnia, Estados Unidos, 2007)

[29] D. Raggett, J. Lam, I. Alexander and M. Kmiec, *Raggett on HTML 4* (Addison-Wesley, 1998),
URL: https://books.google.com.br/books?id=tE4PAQAAMAAJ

[30] M. S. Silva, *Criando sites com HTML: sites de alta qualidade com HTML e CSS* (Novatec Editora Ltda, São Paulo, Brasil, 2008)

[31] D. Flanagan, *JavaScript: The Definitive Guide: Activate Your Web Pages* (O'Reilly Media, 2011),
URL: https://books.google.com.br/books?id=6TAODdEIxrgC

[32] W. Soares, *AJAX (Asynchronous JavaScript And XML): guia prático* (Ed. Érica, 2006),
URL: https://books.google.com.br/books?id=Dx2KPgAACAAJ

[33] *Node.js*,
URL: https://nodejs.org/en/, consultado em Maio de 2017

[34] I. K. Chaniotis, K.-I. D. Kyriakou and N. D. Tselikas, *Is Node.js a viable option for building modern web applications? A performance evaluation study*, Computing **97** (2015) (10), pp. 1023–1044,
URL: http://dx.doi.org/10.1007/s00607-014-0394-9, consultado em Maio de 2017

[35] D. Drohan, *Dynamic Web Development*,
URL: https://ddrohan.github.io/wit-wad/topic02-node/talk-1-node-1/node.1.pdf, consultado em Maio de 2017

[36] T. Croucher, *Node : up and running* (O'Reilly Media, Inc, Califórnia, Estados Unidos, 2012)

[37] *NPM*,
URL: `https://www.npmjs.com/`, consultado em Maio de 2017

[38] V. H. de Paiva Gonçales, *Introdução ao Node.js*,
URL: `http://victorvhpg.github.io/minicurso-introducao-NodeJS/slides/index.html#/`, consultado em Junho de 2017

[39] F. Lardinois, *GitHub Open Sources Its Atom Text Editor*,
URL: `https://techcrunch.com/2014/05/06/github-open-sources-its-atom-text-editor/`, consultado em Maio de 2017

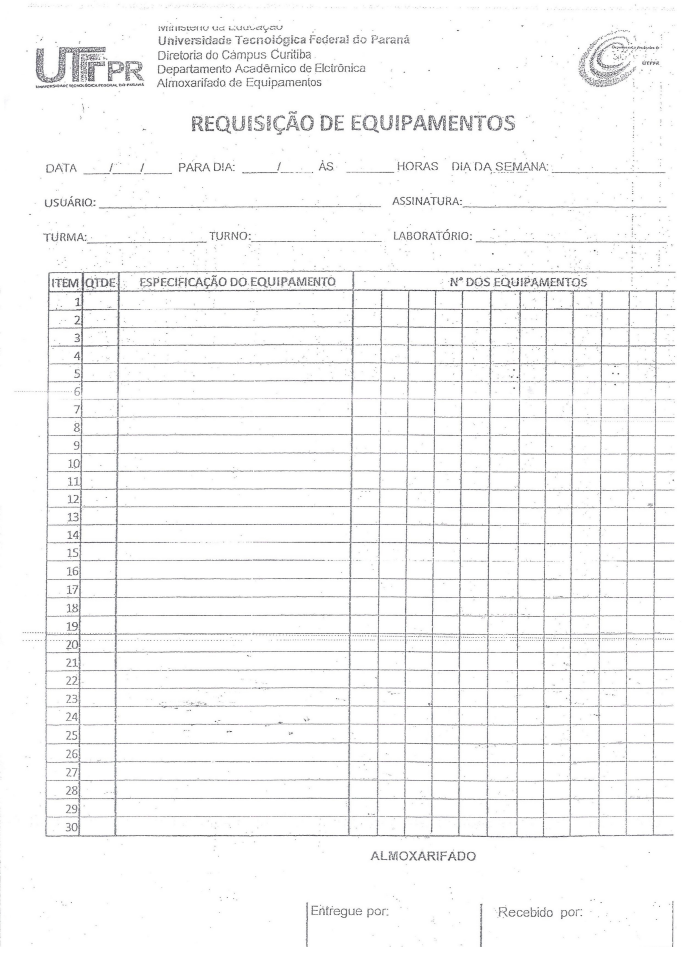
[40] *Atom Flight Manual*,
URL: `http://flight-manual.atom.io/#the-native-web`, consultado em Maio de 2017

[41] S. A. Gabarro, *Using PhpMyAdmin* (Wiley-IEEE Press, 2007), pp. 151–158,
URL: `http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=5989449`

[42] *Express - Basic routing*,
URL: `http://expressjs.com/en/starter/basic-routing.html`, consultado em Maio de 2017

[43] S. B. O. Blog, *Express.js Middleware Demystified*,
URL: `https://goo.gl/RihnAs`, consultado em Maio de 2017

Anexo A. Ficha de requisição de equipamentos



Ministério da Educação
Universidade Tecnológica Federal do Paraná
Diretoria do Câmpus Curitiba
Departamento Acadêmico de Eletrônica
Almoxarifado de Equipamentos

REQUISIÇÃO DE EQUIPAMENTOS

DATA ___/___/___ PARA DIA: ____/____ ÀS _____HORAS DIA DA SEMANA: _____

USUÁRIO: _____ ASSINATURA: _____

TURMA: _____ TURNO: _____ LABORATÓRIO: _____

ITEM	QTDE	ESPECIFICAÇÃO DO EQUIPAMENTO	Nº DOS EQUIPAMENTOS
1			
2			
3			
4			
5			
6			
7			
8			
9			
10			
11			
12			
13			
14			
15			
16			
17			
18			
19			
20			
21			
22			
23			
24			
25			
26			
27			
28			
29			
30			

ALMOXARIFADO

Entregue por:

Recebido por:

Figura 4.1: Ficha para requisição de equipamentos no almoxarifado do departamento acadêmico de eletrônica da UTFPR

Anexo B. MIFARE Classic

MIFARE Classic EV1

	MIFARE Classic
	MIFARE Classic EV1
RF Interface	Contactless only, ISO/IEC 14443-2
Protocol	ISO/IEC 14443-3
UID	7-byte UID, RID, 4-byte NUID
Comm, Speed	106 kbps
EEPROM	1KB, 4KB
Memory Model	Compact, Sectors & 16-byte block
Crypto	Crypto-1
Key Length	48-bit
Authentication	3-pass mutual
Comm, Security	Encrypted
MIsmartApp	N/A
Transaction MAC	N/A
Multi Key Sets	N/A
Proximity Check	No
Virtual Card Select	No
CC Certification	No
ISO 7816-4 APDU	No

Figura 4.2: MIFARE-Classic

Anexo C - Descrição dos casos de uso

Tabela 4.1: Casos de uso - Empréstimo de equipamentos

	Empréstimo de equipamentos
Descrição	Tanto alunos quanto os professores podem realizar empréstimo de equipamentos no armazém do DAELN (UTFPR). Este processo necessita do auxílio de um funcionário.
Atores	Aluno/Professor e Funcionário
Pré-condições	O aluno/professor, funcionário e equipamento devem estar previamente registados no sistema. O empréstimo ocorre apenas com a autorização de um funcionário.
Pós-condições	Apenas equipamentos com status "Disponível"podem ser emprestados. Após realizado o empréstimo o equipamento passa a apresentar o status de "Pendente". Nesta condição não será possível realizar outro empréstimo ou manutenção (do equipamento pendente). A pendencia encerra-se com a devolução do equipamento ao armazém.

Anexo C.

Tabela 4.2: Fluxo básico,de eventos para o empréstimo de equipamentos

Fluxo básico, de eventos para o empréstimo de equipamentos		
	Ações dos atores	**Ações do sistema**
1	Selecionar a opção empréstimos no menu principal.	O sistema irá redirecionar o utilizador para a página de empréstimos.
2	Realizar o procedimento de leitura do cartão RFID.	Feita a leitura do código de identificação registado no cartão (UID), esta informação é enviada ao sistema, que irá confrontar com a base de dados e realizará o correto preenchimento dos campos.
3	Um funcionário é responsável pela seleção dos equipamentos desejados e autorização do empréstimo.	Apenas com o aval do funcionário responsável será possível realizar o empréstimo.
4	Confirmação do empréstimo através do botão confirmar.	Inclui na tabela empréstimos da base de dados os detalhes referentes cada empréstimo. Altera o status do equipamento para "Pendente". Apresenta uma mensagem informando que o empréstimo foi realizado com sucesso.

Anexo C.

Tabela 4.3: Casos de uso - Devolução de equipamentos

	Devolução de equipamentos
Descrição	Devolução de um equipamento emprestado.
Atores	Aluno/Professor ou Funcionário.
Pré-condições	Equipamento ter sido emprestado.
Pós-condições	Atualiza a tabela empréstimo, com o status "Retornado". Regista a devolução e disponibiliza o equipamento para novos empréstimos ou manutenção.

Tabela 4.4: Fluxo básico,de eventos para a devolução de equipamentos

Fluxo básico de eventos para a devolução de equipamentos		
	Ações dos atores	**Ações do sistema**
1	Selecionar a opção devolução no menu principal.	O sistema irá redirecionar o utilizador para a página onde irá registar a devolução.
2	Realizar o procedimento de leitura do cartão RFID referente ao equipamento a ser devolvido.	O sistema irá atualizar na base de dados à tabela empréstimo, alterando o status de "Empréstimo"para "Retornado". Na sequência regista a devolução do equipamento e altera o status do emprestado para "Retornado".

Anexo C.

Tabela 4.5: Casos de uso - Envio dos equipamentos para a manutenção

Envio dos equipamentos para a manutenção	
Descrição	Inicializa o processo para envio de equipamentos para manutenção.
Atores	Funcionário.
Pré-condições	É necessário que o equipamento esteja com o status "Disponível".
Pós-condições	Estando em manutenção o equipamento não pode ser emprestado.

Tabela 4.6: Fluxo básico,de eventos para envio dos equipamentos para a manutenção

Fluxo básico, de eventos para envio dos equipamentos para a manutenção		
	Ações do ator	**Ações do sistema**
1	Selecionar a opção manutenção no menu principal.	O sistema irá redirecionar o utilizador para a página onde poderá registar os dados do equipamento a ser enviado para a manutenção.
2	Deverá selecionar o tipo de manutenção (preventiva ou reparativa).	O sistema irá processar o tipo de manutenção e fornece as opções para o funcionário se identificar, e anotar observações referentes ao problema do equipamento.
3	Confirmação do envio para manutenção através do botão confirmar.	Inclui na tabela manutenção da base de dados os detalhes referentes cada manutenção. Altera o status do equipamento para "Manutenção". Apresenta uma mensagem informando que o envio para a manutenção foi realizado com sucesso.

Anexo C.

Tabela 4.7: Casos de uso - Consultar empréstimos e manutenções

Consultar empréstimos e manutenções	
Descrição	Permite ao utilizador do sistema verificar todos empréstimos, manutenções e devoluções realizadas.
Atores	Aluno/Professor ou Funcionário
Pré-condições	Nenhuma
Pós-condições	Nenhuma

Tabela 4.8: Fluxo básico de eventos para consulta de empréstimos e manutenções

Fluxo básico de eventos para consulta de empréstimos e manutenções		
	Ações do ator	**Ações do sistema**
1	Selecionar a opção relatório no menu principal	O sistema irá redirecionar o utilizador para a página de relatórios
2	Na página relatório, a opção Relatórios de manutenção ou empréstimo deverá ser selecionada	O sistema irá redirecionar o utilizador a uma página com o histórico de manutenções ou empréstimos.

Anexo C.

Tabela 4.9: Casos de uso - Geração de relatórios para impressão

	Geração de relatórios para impressão
Descrição	Gerar arquivos no formato PDF contendo as informações dos empréstimos, manutenções ou baixas, que poderão ser enviadas para impressora.
Atores	Aluno/Professor ou Funcionário
Pré-condições	Nenhuma
Pós-condições	Nenhuma

Tabela 4.10: Fluxo básico de eventos para geração de relatórios para impressão

Fluxo básico de eventos para geração de relatórios para impressão		
	Ações do ator	**Ações do sistema**
1	Selecionar a opção relatório no menu principal.	O sistema irá redirecionar o utilizador para a página de relatórios.
2	Na página relatório, o utilizador tem a opção para gerar relatórios de empréstimos, manutenção ou baixa de equipamentos.	O sistema irá redirecionar o utilizar para a página de opção desejada.
3	Clicar na opção gerar relatório.	O sistema irá gerar o relatório desejado, as informações contidas neste relatório poderão ser previamente visualizadas na tela. Os arquivos PDF serão automaticamente armazenados na pasta relatórios do sistema.

Anexo C.

Tabela 4.11: Casos de uso - Registo de utilizadores e equipamentos

	Registo de utilizadores e equipamentos
Descrição	Realizar o registo de um novo cartão RFID, registando os dados de um aluno, professor, funcionário ou equipamento.
Atores	Funcionário.
Pré-condições	Possuir um cartão RFID cujo código de identificação registado no cartão (UID), não encontre-se no banco de dados.
Pós-condições	Nenhuma

Tabela 4.12: Fluxo básico,de eventos para registo de utilizadores e equipamentos

Fluxo básico, de eventos para registo de utilizadores e equipamentos		
	Ações do ator	**Ações do sistema**
1	Selecionar a opção adicionar no menu principal.	O sistema irá redirecionar o utilizador para a página onde poderá realizar o processo de registo de um novo código de identificação.
2	Realizar o procedimento de leitura do cartão RFID a ser registado.	O sistema recebe o código de identificação registado no cartão (UID) e libera os campos de registo para preenchimento. Sendo que irá recusar o registo de UID anteriormente registadas.
3	Deverá preencher os dados de registo de acordo.	Processa e regista na base de dados a nova UID, com os respectivos dados.

Anexo C.

Tabela 4.13: Retorno de equipamentos da manutenção

Retorno de equipamentos da manutenção	
Descrição	Retorno de um equipamento enviado para a manutenção.
Atores	Funcionário.
Pré-condições	Equipamento ter sido enviado para a manutenção.
Pós-condições	Atualiza a tabela Manutenção, com o status "Retornado". Regista o retorno ou baixa do equipamento.

Tabela 4.14: Fluxo básico de eventos para a retorno equipamento da manutenção

Fluxo básico de eventos para a retorno equipamento da manutenção		
	Ações do ator	**Ações do sistema**
1	Selecionar a opção retorno no menu principal.	O sistema irá redirecionar o utilizador para a página onde irá efetuar o retorno do equipamento que estava em manutenção.
2	Realizar o procedimento de leitura do cartão RFID do equipamento que está retornando da manutenção.	O sistema ira identificar o equipamento. E retornar as informações que constam no formulário de envio do equipamento para a manutenção.
3	O ator deverá selecionar se deseja retornar ou dar baixa no equipamento.	O sistema identifica a opção realizada e libera o campo para identificação do funcionário responsável e observação de retorno.

Anexo D.

Tabela 4.15: Casos de uso - Baixa de equipamentos

	Baixa de equipamentos
Descrição	Baixa de equipamentos,do patrimônio.
Atores	Funcionário.
Pré-condições	Equipamentos obsoletos, avariados ou sem condições para reparação.
Pós-condições	Baixa do equipamento e exclusão do armazém.

Tabela 4.16: Fluxo básico de eventos para a baixa de equipamentos

Fluxo básico, de eventos para baixa de equipamentos		
	Ações do ator	**Ações do sistema**
1	Selecionar a opção baixa no menu principal	O sistema irá redirecionar o utilizador para a página onde irá efetuar a baixa do equipamento.
2	Realizar o procedimento de leitura do cartão RFID do equipamento a ser baixado.	O sistema ira identificar o equipamento
3	O ator deverá selecionar se deseja baixar	Irá deixa-lo inativo na,base de dados. E registar informações,do destino dado ao equipamento.

Anexo D - Programa - Arduino

```
/*
 * ----------------------------------
 *             MFRC522      Arduino
 *             Reader/PCD   UNO
 * Signal      Pin          Pin
 * ----------------------------------
 * RST/Reset   RST          09
 * SPI SS      SS           10
 * SPI MOSI    MOSI         11
 * SPI MISO    MISO         12
 * SPI SCK     SCK          13
 */
#include <MFRC522.h>
#include <SPI.h>
#define SS_PIN 10
#define RST_PIN 9
MFRC522 mfrc522(SS_PIN, RST_PIN);   // INSTANCIA MFRC522

void setup()
{
    Serial.begin(9600);   // Inicializa a comunicacao seria a ...
    uma taxa de 9600 bits por segundo
    SPI.begin();
    pinMode(3, OUTPUT);
    pinMode(4, OUTPUT);
```

```
25      //INICIALIZA O MODULO MFRC522
        mfrc522.PCD_Init();   // Initiate MFRC522
27  }
    void loop()
29  {
        // Procura por novos Cartoes
31      if ( ! mfrc522.PICC_IsNewCardPresent())
        {
33          return;
        }
35      // Seleciona um Cartao
        if ( ! mfrc522.PICC_ReadCardSerial())
37      {
            return;
39      }
        //Mostra o UID no serial monitor
41      String content = "";

43      for (byte i = 0; i < mfrc522.uid.size; i++)
        {
45          content.concat(String(mfrc522.uid.uidByte[i] < 0x10 ? ...
       "0" : ""));
            content.concat(String(mfrc522.uid.uidByte[i], HEX));
47      }
        content.toUpperCase();
49  }
```

Código 4.1: Código para o Arduino UNO realizar a comunicação com o leitor de RFID MF-RC522