UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO
CENTRO TECNOLÓGICO
DEPARTAMENTO DE ENGENHARIA ELÉTRICA
PROJETO DE GRADUAÇÃO

LUCAS SOARES PESSINI

SISTEMA ELETRÔNICO DE CONTROLE DE EMPRÉSTIMOS PARA LABORATÓRIOS DO DEL/UFES

VITÓRIA
2019

LUCAS SOARES PESSINI

SISTEMA ELETRÔNICO DE CONTROLE DE EMPRÉSTIMOS PARA LABORATÓRIOS DO DEL/UFES

Parte manuscrita do Projeto de Graduação do aluno **Lucas Soares Pessini**, apresentado ao Departamento de Engenharia Elétrica do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do grau de Engenheiro Eletricista.

Orientador: Prof. Dr. André Ferreira

VITÓRIA
2019

LUCAS SOARES PESSINI

SISTEMA ELETRÔNICO DE CONTROLE DE EMPRÉSTIMOS PARA LABORATÓRIOS DO DEL/UFES

Parte manuscrita do Projeto de Graduação do aluno **Lucas Soares Pessini**, apresentado ao Departamento de Engenharia Elétrica do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do grau de Engenheiro Eletricista.

Aprovada em 16, de dezembro de 2019.

COMISSÃO EXAMINADORA:

Prof. Dr. André Ferreira
Universidade Federal do Espírito Santo
Orientador

Profa. Dra. Eliete Maria de Oliveira Caldeira
Universidade Federal do Espírito Santo
Examinadora

Prof. Dr. Walbermark Marques dos Santos
Universidade Federal do Espírito Santo
Examinador

.

Aos alunos e professores do Curso de Engenharia Elétrica da UFES.

Agradeço ao professor orientador Dr. André Ferreira por seu apoio, dedicação e estímulo para a produção deste projeto. Agradeço aos meus pais e meu irmão, por sempre estarem ao meu lado apoiando e me dando toda a ajuda para concluir minha graduação, sou imensamente grato por eles. Aos meus amigos de classe que estiveram comigo todos esses anos, enfrentando as mesmas dificuldades e ajudando direta ou indiretamente a superá-las.

RESUMO

Aqui se apresenta um projeto para o controle eletrônico de empréstimos de *kits* de apoio ao ensino e de equipamentos disponíveis no Laboratório de Eletrônica do Departamento de Engenharia Elétrica da UFES, com os procedimentos e os embasamentos teóricos seguidos. Foi realizado um estudo sobre o monitoramento da localização de tais empréstimos. O registro de empréstimo armazena em um banco de dados e há uma interface da *web* para registro e consulta dos empréstimos. O sistema também gera alguns relatórios contendo informações pertinentes, além de manter um histórico dos empréstimos. Foi desenvolvido um sistema eletrônico para registro e controle dos empréstimos de *kits* e equipamentos dos laboratórios do DEL/UFES. Também foram realizados um estudo e simulações sobre o monitoramento da localização dos equipamentos emprestados. Além disso, foi desenvolvido um módulo de 'autoatendimento', onde simula-se retirada/entrega de um equipamento ou *kit* pelo próprio usuário, sem a necessidade de um outro operador.

Palavras chave: Banco de dados. Sistema *web*. Automatização de processos.

ABSTRACT

Here is a project for the electronic control of loans for teaching support kits and equipment available at the Electronics Laboratory of the Department of Electrical Engineering at UFES, with the procedures and theoretical bases followed. A study was carried out on monitoring the location of such loans. The loan record stores in a database and there is a web interface for registering and consulting the loans. The system also generates some reports containing pertinent information, in addition to maintaining a history of loans. An electronic system was developed to record and control the loans for kits and equipment from the DEL / UFES laboratories. A study and simulations were also carried out on monitoring the location of the borrowed equipment. In addition, a 'self-service' module was developed, in which the removal / delivery of equipment or kit by the user is simulated, without the need for another operator.

Keywords: Database. Web system. Process automation.

LISTA DE FIGURAS

Figura 1 – Esquemático geral do projeto 16
Figura 2 – Esquemático de reconhecimento do usuário, *kits* ou equipamento e inserindo no banco de dados 16
Figura 3 – Código de barras do tipo *Code 39* 25
Figura 4 – Código de barras do tipo *Code* 128 26
Figura 5 – Código de barras do tipo *Code 2 of 5* 26
Figura 6 – Código de barras do tipo UPC 27
Figura 7 – Código de barras do tipo IAN 27
Figura 8 – Código de barras do tipo PDF417 28
Figura 9 – Código de barras do tipo *Data Matrix* 28
Figura 10 – Código de barras do tipo *Quick Response* 29
Figura 11 – Esquemático das tabelas do banco de dados e como elas se relacionam 41
Figura 12 – Página inicial do aplicativo *web* 45
Figura 13 – Página inicial do aplicativo *web* quando logado em uma conta administrativa - Parte 1/2 46
Figura 14 – Página inicial do aplicativo *web* quando logado em uma conta administrativa – Parte 2/2 46
Figura 15 – Página do aplicativo *web* para adicionar um novo usuário 47
Figura 16 – Página do aplicativo *web* para *login* 48
Figura 17 – Página do aplicativo *web* para visualizar todos os setores 49
Figura 18 – Arquivo Excel com todas os setores cadastrados 49
Figura 19 – Página do aplicativo *web* para adicionar um novo setor 50
Figura 20 – Página do aplicativo *web* mostrando a lista de usuários 51
Figura 21 – Arquivo Excel com todos os usuários 51
Figura 22 – Página *web* que mostra todos os empréstimos 52
Figura 23 – Arquivo Excel com todos os empréstimos 53
Figura 24 – Página para adicionar um novo empréstimo 53
Figura 25 – Página para visualizar todos os estudantes 54
Figura 26 – Arquivo Excel com todos os estudantes 55
Figura 27 – Página *web* para adicionar um novo estudante 55
Figura 28 – Página *web* para visualizar todos os equipamentos e *kits* 56
Figura 29 – Arquivo Excel com todos os *kits* e equipamentos 57

Figura 30 – Página *web* para adicionar um novo *kit* ou equipamento 57
Figura 31 – Página *web* para criar vínculo entre o estudante e o equipamento ou *kit* 58
Figura 32 – Fazendo login em uma conta administrativa 59
Figura 33 – 'Ler Código de Barras' no menu de navegação 60
Figura 34 – Procedimentos para ler a matrícula do estudante através do código de barras: (a)Apertar no botão 'Start' no título '*Scan Barcode From Student ID*'; (b) Posicionar a câmera no código de barras do aluno; e (c) Apertar '*Ok*' na janela de *Pop-up* que aparece caso o código for correto. 61
Figura 35 – Procedimentos para ler a matrícula do estudante através do código de barras: (a)Aperta no Botão 'Start' no título '*Scan Barcode From Equipament*'; (b) Posicionar a câmera no código de barras do equipamento ou *kit*; e (c) Apertar 'Ok' na janela de Pop-up que apareceu caso for correto 62
Figura 36 – Confirmar o empréstimo 62
Figura 37 – Pesquisando o nome do aluno no banco de dados para fazer o empréstimo 63
Figura 38 – Circuito que estará situado nos equipamentos com o módulo GPS 64
Figura 39 – Circuito que ficará ligado a um computador recebendo os dados de localização e enviando para o banco de dados 64
Figura 40 – Sensor GPS utilizado 65
Figura 41 – Conectando COM3 e COM4 virtualmente com *Virtual Serial Port Driver* 66
Figura 42 – Conectando COM1 e COM2 virtualmente com *Virtual Serial Port Driver* 67
Figura 43 – Interface do *Prompt* quando executa o *script* enquanto a simula o circuito 68
Figura 44 – Dados inseridos no banco 68
Figura 45 – Esquemático do circuito eletrônico do autoatendimento 69
Figura 46 – Página para adicionar um novo empréstimo para ser utilizado no 'autoatendimento' 70
Figura 47 – Página inicial com próximos empréstimos daqui 2 horas 71
Figura 48 – Página direcionada logo depois que é apertado em '*Get*' 71

LISTA DE ABREVIATURAS E SIGLAS

1D	Unidimensional
2D	Bidimensional
ASCII	*American Standard Code for Information Interchange*
BD	Banco de Dados
COM	*Component Object Model*
CPF	Cadastro de Pessoa Física
CRUD	*Create, Read, Update and Delete*
CSS	*Cascading Style Sheets*
CTII	Prédio II do Centro Tecnológico da UFES
DEL	Departamento de Engenharia Elétrica
DOS	*Disk Operating System*
FTP	*File Transfer Protocol*
FTPS	*File Transfer Protocol Secure*
GIS	*Geographic Information System*
GPS	*Global Positioning System*
HTML	*HyperText Markup Language*
IAN	*International Article Number*
IDE	*Integrated Development Environment*
LCEE	Laboratório de Computação da Engenharia Elétrica
LED	*Light-Emitting Diode*
MVC	*Model, View and Controller*
PCB	*Printed Circuit Board*
PHP	*Personal Home Page*
QR	*Quick Response*
RTLS	*Real-Time Locating Systems*
SGBD	Sistemas de Gestão de Base de Dados
SQL	*Structured Query Language*
TCC	Trabalho de Conclusão de Curso
UART	*Universal Asynchronous Receiver/Transmitter*
UFES	Universidade Federal do Espírito Santo
UPC	*Universal Product Codes*

USB	*Universal Serial Bus*
USPS	*United States Postal Service*
WMI	*Windows Management Instrumentation*

SUMÁRIO

1 INTRODUÇÃO 14

1.1 Justificativa 17

1.2 Objetivos 18

1.2.1 Objetivo Geral 18

1.2.2 Objetivos Específicos 18

2 EMBASAMENTO TEÓRICO 20

2.1 Automatização de Processos 20

2.2 Aplicação *Web* 20

2.2.1 HTML 21

2.2.2 PHP 21

2.2.3 JavaScript 22

2.3 Banco de Dados 22

2.4 Sistema Embarcado 23

2.5 Código de Barras 24

2.5.1 *Code* 39 25

2.5.2 *Code* 128 25

2.5.3 *Interleaved* 2 *of* 5 26

2.5.4 *Universal Product Codes* (*UPC*) 26

2.5.5 *International Article Number* (*IAN*) 27

2.5.6 PDF417 27

2.5.7 Data Matrix 28

2.5.8 Quick Response (QR) Codes 29

3 ALOCAÇÃO DE RECURSOS 30

3.1 XAMPP 30

3.2 MySQL Workbench 30

3.3 CakePHP 31

3.4 Github 32

3.5 000webhost 33

3.6 FileZilla 33

3.7 Google Chrome *DevTools* 33

3.8 ZXing 34

3.9 Simular Câmera Para Leitura de Código de Barras **34**
3.10 Proteus **35**
3.11 Arduino **35**
3.12 PowerShell **35**
3.13 Arquivos '.bat' **36**
3.14 Virtual Serial Port Driver por Eltima Software **36**
3.15 XBee **37**
4 METODOLOGIA E ETAPAS DE DESENVOLVIMENTO **39**
5 APRESENTAÇÃO DO SISTEMA **41**
5.1 Banco de Dados **41**
5.1.1 Página Inicial 44
5.1.2 *New User* ou Add um Usuário 47
5.1.3 Login 48
5.1.4 Todos os Setores 48
5.1.5 *Download* Todos os Setores 49
5.1.6 Adicionar um Setor 50
5.1.7 Todos os Usuários 50
5.1.8 *Download* Todos os Usuários 51
5.1.9 Todos os Empréstimos 52
5.1.10 *Download* Todos os Empréstimos 52
5.1.11 Adicionar um Empréstimo 53
5.1.12 Todos os Estudantes 54
5.1.13 *Download* Todos os Estudantes 54
5.1.14 Adicionar um Estudante 55
5.1.15 Todos os *Kits* e Equipamentos 56
5.1.16 *Download* Todos os *Kits* e Equipamentos 56
5.1.17 Adicionar um *Kit* ou Equipamento 57
5.2 Interface de Aquisição/Vínculo **58**
5.3 Simulação do Módulo RTLS **64**
5.4 Módulo de 'Autoatendimento' **69**
6 CONCLUSÃO **72**
REFERÊNCIAS BIBLIOGRÁFICAS **73**
APÊNDICE A – CÓDIGO FEITO NA IDE DO ARDUINO PARA O MÓDULO RTLS **76**

APÊNDICE B – CÓDIGO EXECUTADO NO POWERSHELL PARA ENVIAR DADOS PARA O BANCO DE DADOS 79
APÊNDICE C – CÓDIGO DO ARQUIVO '.BAT' PARA O MÓDULO DE 'AUTOATENDIMENTO' 80
APÊNDICE D – CÓDIGO FEITO NA IDE DO ARDUINO PARA O MÓDULO DE 'AUTOATENDIMENTO' 81

1 INTRODUÇÃO

O Laboratório de Eletrônica do Departamento de Engenharia Elétrica da UFES possui diversos equipamentos, tais como osciloscópios, geradores de sinais, multímetros, fontes de alimentação, dentre vários outros, que são utilizados diariamente (das 7h às 19h) durante as aulas práticas, em projetos dos alunos (laboratório de portas abertas) e também atendendo a situações específicas de empréstimos no Centro Tecnológico, como o exemplo de laboratórios de pesquisa e de projetos de extensão. Além dos referidos equipamentos, o laboratório disponibiliza aos alunos *kits* de apoio ao ensino que incluem pontas de prova de osciloscópio, cabos para o gerador de sinais e *protoboard*.

Todo o processo de empréstimo atual, seja de *kits* e/ou equipamentos, é controlado manualmente, onde um responsável pelo setor realiza o vínculo de um número de identificação do usuário ao registro do equipamento/*kit*. Para casos específicos, também são anotadas, manualmente, informações adicionais como o local para onde o equipamento será levado, data e hora do empréstimo, previsão de retorno e contato do responsável.

Da forma como está implementado hoje, tal sistema de empréstimos não é eficiente, gera atrasos, erros, e também não permite geração de relatórios úteis para o gestor do laboratório, como por exemplo o histórico de utilização, dados para manutenção programada, dados para aquisições futuras e análise de demanda por horários e dias da semana.

Tendo em vista a importância de tais equipamentos no desenvolvimento acadêmico dos alunos, além do custo elevado dos mesmos, torna-se imprescindível um controle mais eficiente de alocação e movimentação desses equipamentos.

Isto posto, será desenvolvido um sistema eletrônico para registro e controle dos empréstimos de *kits* e equipamentos dos laboratórios do DEL/UFES. Também serão realizados um estudo e simulações sobre o monitoramento da localização dos equipamentos emprestados. Além disso, será desenvolvido um módulo de ‘autoatendimento’, onde simula-se retirada/entrega de um equipamento ou *kit* pelo próprio usuário, sem a necessidade de um outro operador.

O sistema de controle eletrônico de empréstimos será rápido, de operação simples e intuitiva, além de fornecer relatórios em uma interface através de consulta a um banco de dados.

Ele vinculará (ou desvinculará, dependendo do contexto) informações únicas do usuário (matrícula, por exemplo) ao objeto. Para obter agilidade na captação dos dados do empréstimo e maior rapidez nas transações serão utilizados código de barras (padrão disponível no documento de identificação do aluno) e uma interface *web* para realizar tais atividades.

A interface poderá ser acessada por um aparelho celular que permite realizar as funções de leitura dos códigos, (des)vinculação usuário-objeto e consulta ao banco de dados remotamente.

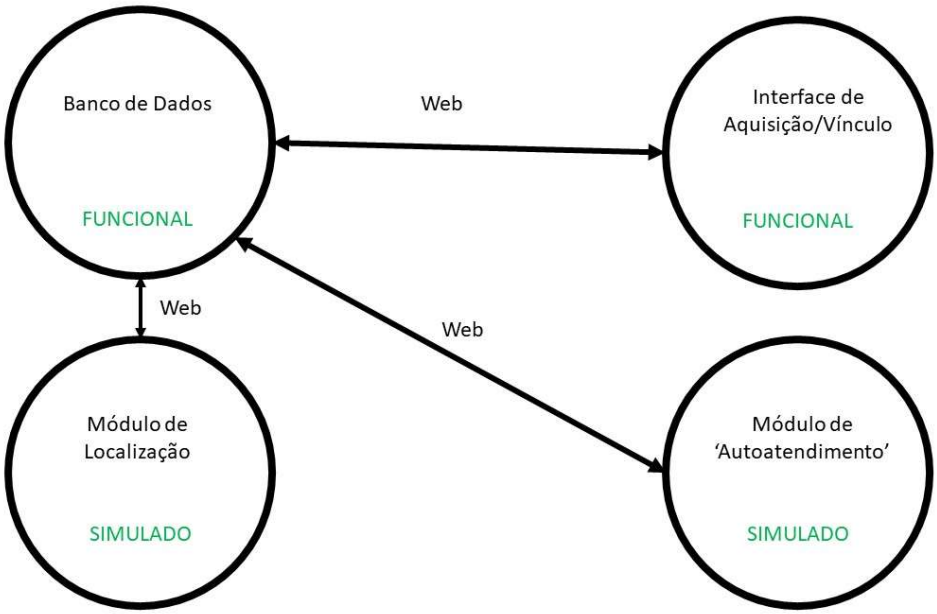
Módulos RTLS (*Real-Time Locating Systems*) baseados em XBee poderão fornecer a localização em tempo real com boa precisão dos equipamentos emprestados, permitindo ao coordenador do laboratório um monitoramento via *web* desses recursos. Módulos GPS (*Global Positioning System*) também são utilizados neste sentido, mas sua maior contribuição se dá em ambientes exteriores. Essas tecnologias apresentam características (alcance, custo, robustez, precisão) distintas.

Pode-se dizer, então, que o projeto tem os seguintes focos:

- **Desenvolvimento de interface *web*:** integrar *software* para leitura de código de barras com aplicativo que será desenvolvido para vínculo usuário-objeto e acesso ao banco de dados;
- **Banco de dados:** acessos (leitura e escrita) remotos a um banco de dados responsável pelo armazenamento dos registros. Os dados referentes aos vínculos usuário-objeto serão gravados para relatórios futuros;
- **Interligação e configuração de rede de sensores RTLS:** uma rede de sensores e transmissores deve ser estabelecida visando a correta localização dos objetos e esta informação a ser disponibilizada para visualização remota via *web*;
- **Módulo de autoatendimento:** Um estudo será realizado, com vistas a permitir que os próprios usuários, desde que habilitados, realizem empréstimos e devoluções no laboratório, sem a necessidade de um operador local do sistema.

Assim, tem-se o esquemático geral dos tópicos ditos anteriormente na Figura 1.

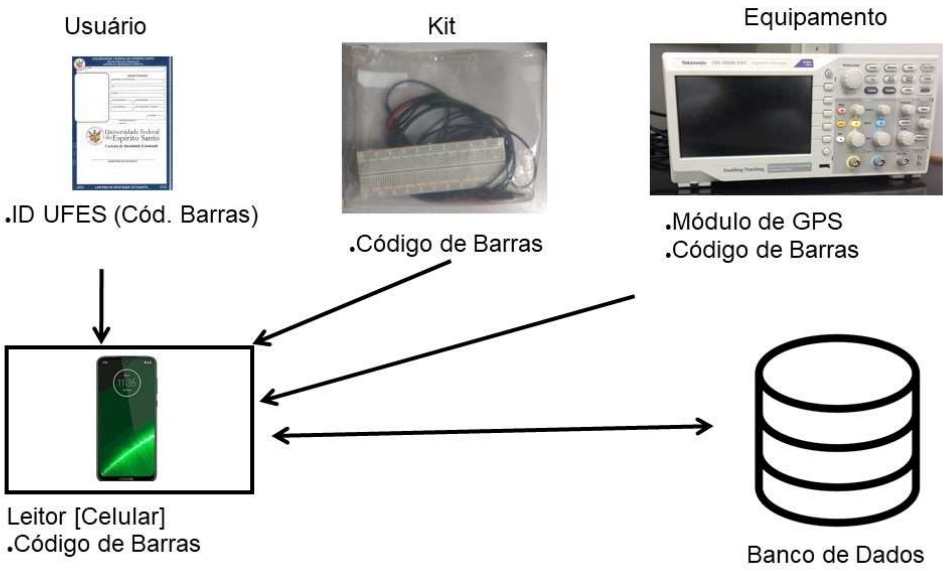
Figura 1 – Esquemático geral do projeto



Fonte: Produção do próprio autor.

Apresenta-se na Figura 2 o esquemático em relação à interface de aquisição e vínculo do usuário com o *kit* ou equipamento e como ele se relaciona com o banco de dados.

Figura 2 – Esquemático de reconhecimento do usuário, *kits* ou equipamento e inserindo no banco de dados



Fonte: Produção do próprio autor.

Como mostrado na Figura 2, a interface *web* será capaz de criar o vínculo entre o aluno e o equipamento ou *kit* utilizando o leitor de código de barras e inserirá nos bancos de dados o empréstimo realizado.

1.1 Justificativa

O sistema manual de empréstimos, atualmente em uso no laboratório de eletrônica do DEL/UFES, não é eficiente, gerando erros e atrasos, além da impossibilidade de registro do histórico e da produção de relatórios. Assim, é necessário que seja feito um controle mais aprimorado deste processo.

Tais equipamentos dos laboratórios são constantemente utilizados e, dessa forma, foi proposto desenvolver uma solução e executá-la de forma que auxilie o processo de gerenciamento destes itens, ajudando tanto alunos e professores como os próprios funcionários do local.

Os referidos equipamentos são de alto custo e são utilizados nas aulas práticas, o que torna necessário ter um controle minimamente estruturado dos empréstimos para que não hajam perdas, prejuízos e aulas práticas com alunos sem equipamentos. Para este controle, os dados do equipamento e do usuário que fez a solicitação devem ser armazenados corretamente para que, caso ocorra algum problema com o equipamento, sejam tomadas atitudes necessárias para resolvê-lo, ou caso seja utilizado em alguma outra aula, não possa ser emprestado.

Com um sistema automatizado, o processo ganha agilidade, maior segurança e ainda mantém um histórico atualizado de cada um dos equipamentos utilizados. Uma implementação de um sistema eletrônico para controlar os empréstimos é necessário para que os dados sejam registrados corretamente. Assim, é proposto desenvolver um projeto que apresente melhorias em economia de tempo e trabalho humano dos laboratórios de eletrônica do prédio do CT II.

Manter um histórico de empréstimos também pode ser útil para evitar o uso excessivo de um mesmo equipamento, podendo-se planejar um revezamento daqueles disponíveis.

O sistema proposto propiciará o uso dos equipamentos fora do laboratório, pois haverá um monitoramento de suas localizações através de módulos baseados em RTLS, que registrarão continuamente as posições geográficas dos equipamentos no BD (Banco de Dados).

Será pesquisada a forma de como realizará o acesso à interface *web* e ao banco de dados. Também será apresentada uma solução RTLS tendo em vista os custos envolvidos, autonomia, além da adequação e posicionamento desta eletrônica embarcada nos equipamentos do laboratório.

Além disso, um módulo de 'autoatendimento' será apresentado. Ele permitirá que o próprio usuário, desde que com acesso autorizado, retire ou entregue os equipamentos e *kits* sem intermédio de um funcionário.

1.2 Objetivos

1.2.1 Objetivo Geral

O projeto teve como objetivo geral a simplificação e automatização do gerenciamento de empréstimos de equipamentos e *kits* no Laboratório de Eletrônica do Departamento de Engenharia Elétrica da UFES. Este sistema obtém de forma automatizada o código do equipamento e matrícula do estudante através do código de barras, data de empréstimo e entrega e além de informações adicionais para geração de relatórios.

1.2.2 Objetivos Específicos

Os objetivos foram listados em tópicos:

- Desenvolver um sistema para fazer a leitura do código de barra presente na carteira de estudante e no *kit*/ equipamento;
- Desenvolver um banco de dados que irá armazenar os dados dos empréstimos;
- Estudar um sistema de baixo custo que insere no BD a localização do equipamento (RTLS);

- Desenvolver uma interface *web* que realize o controle dos empréstimos de maneira automatizada e que seja integrada ao sistema de leitura do código de barras;
- Desenvolver um estudo de como poderia ser feito um sistema de 'autoatendimento'.

Após a Introdução, será apresentada no Capítulo 2 uma revisão teórica dos tópicos relacionados ao tema em estudo. Na sequência, o Capítulo 3 descreverá os recursos necessários para a execução do projeto. No Capítulo 4 serão apresentados os procedimentos metodológicos utilizados e quais as etapas do processo de pesquisa a serem desenvolvidas. A seguir, todo o sistema implementado será descrito no Capítulo 5 e, por fim, as conclusões e observações para trabalhos futuros serão apresentadas no Capítulo 6.

2 EMBASAMENTO TEÓRICO

2.1 Automatização de Processos

Automatizar processos significa passar as tarefas realizadas de maneira manual pelas pessoas para equipamentos, máquinas, instrumentos e outros (GESTÃOCLICK, 2012). Para que a automatização de processos ofereça os resultados esperados, é muito importante garantir que sua implantação seja feita de maneira estruturada e de acordo com as diretrizes de onde está sendo aplicado (MERCADO ELETRÔNICO, 2016). No meio industrial, a preocupação com produtividade, redução do risco operacional e qualidade, leva à implantação de sistemas de automatização.

A parte operacional na automação industrial é uma parte do sistema que atua diretamente no processo e é um conjunto de elementos que fazem com que a máquina se mova e realize a operação desejada (CITISYSTEMS, 2016), aperfeiçoamento constante das atividades dos processos.

A automação em processos industriais foi abordada nas disciplinas de Controle Inteligente e Sistemas Realimentados durante o curso, nas quais foram apresentados diversos meios de controlar o processo. Neste projeto será desenvolvido, principalmente, um sistema de *software* e *hardware* para automatizar o processo de empréstimos de equipamentos do laboratório.

2.2 Aplicação *Web*

Uma aplicação *web* é um *software* que é instalado em um servidor *web* e é projetado para responder a solicitações, processar informações, armazenar informações e dimensionar as respostas de acordo com a demanda e, em muitos casos, é distribuído em vários sistemas ou servidores (MACÊDO, 2017). Essas aplicações apresentam várias linguagens de programação (PHP, Javascript, etc) e elementos de interface gráfica (HTML, CSS).

As aplicações *web* se diferenciam das aplicações '*desktop*', pois não precisam de instalação no computador, são acessíveis de qualquer lugar com *internet*, não dependem do sistema operacional (todo o processamento de funções e instruções é feito no servidor *web*) e o

navegador funciona apenas como uma interface da aplicação (ACADEMIA DO CÓDIGO, 2016). Essas vantagens de aplicação *web* foram vistas principalmente na disciplina Redes de Computadores e de Automação.

Os *frameworks* em desenvolvimento de *software* são uma abstração, unindo códigos comuns entre vários projetos de *software* e provendo uma funcionalidade genérica (WIKIPÉDIA, 2013). Para o desenvolvimento desse *software* será utilizada a *framework* CakePHP (CAKEPHP, 2017), que torna a construção de aplicativos da *web* mais simples, mais rápida e com menos código.

2.2.1 HTML

HTML é uma linguagem de computador desenvolvida para permitir a criação de *sites*. Esses *sites* podem ser visualizados por qualquer pessoa conectada à *internet* (YOUR HTML SOURCE, 2014).

O HTML consiste em uma série de códigos de acesso digitados em um arquivo de texto pelo autor do *site* - essas são as *tags*. O texto é salvo como um arquivo HTML e visualizado através de um navegador, como o *Internet Explorer* ou o *Google Chrome*. Este navegador lê o arquivo e converte o texto em um formulário visível, esperando renderizar a página como o autor pretendia. Escrever seu próprio HTML implica usar *tags* corretamente para criar sua visão (YOUR HTML SOURCE, 2014).

2.2.2 PHP

PHP significa *Hypertext Preprocessor*. É uma linguagem de *script* de código aberto, do lado do servidor, usada para o desenvolvimento de aplicativos da *web*. Por linguagem de *script*, entende-se um programa baseado em *script* (linhas de código) escrito para a automação de tarefas (STUDY.COM, 2015).

As páginas da *web* podem ser projetadas usando HTML. Com o HTML, a execução do código é feita no navegador do usuário (lado do cliente). Por outro lado, com a linguagem de *script* do

lado do servidor PHP, ela é executada no servidor antes de chegar ao navegador da *web* do usuário (STUDY.COM, 2015).

O PHP pode ser incorporado em HTML e é adequado para o desenvolvimento da *web* e a criação de páginas da *web* dinâmicas para aplicativos da *web*, aplicativos de comércio eletrônico e aplicativos de banco de dados. É considerada uma linguagem amigável, com capacidade de se conectar facilmente ao MySQL, Oracle e outros bancos de dados (STUDY.COM, 2015).

2.2.3 JavaScript

JavaScript é uma das linguagens de programação mais populares do mundo. Geralmente, o JavaScript é mencionado como uma linguagem projetada principalmente para o desenvolvimento *web front-end* (QUORA, 2018).

Existem vários lugares diferentes onde o JavaScript pode ser usado, mas o local mais comum para usá-lo é em uma página da *web*. De fato, para a maioria das pessoas que usa JavaScript, em uma página da *web* é o único local onde elas o usam (QUORA, 2018).

JavaScript é uma linguagem de *script* de cliente usada para criar páginas da *web*. É usado quando uma página da *web* deve ser dinâmica e adicionar efeitos especiais em páginas como *rollover*, *rollout* e muitos tipos de gráficos. É usado principalmente por todos os *sites* para fins de validação (QUORA, 2018).

2.3 Banco de Dados

Um banco de dados é uma coleção de dados inter-relacionados, representando informações sobre um domínio específico, ou seja, sempre que for possível agrupar informações que se relacionam e tratam de um mesmo assunto, pode-se dizer que se tem um banco de dados (DEVMEDIA, 2014). Como exemplo de banco de dados pode-se citar um sistema de bibliotecas, uma agenda telefônica, um cadastro de clientes, dentre outros.

Um sistema de gerenciamento de banco de dados (SGBD) é um *software* que possui recursos capazes de manipular as informações do banco de dados e interagir com o usuário. Existem

vários SGBDs no mercado, como Oracle, SQL Server, DB2, PostgreSQL, MySQL, o próprio Access ou Paradox, entre outros (DEVMEDIA, 2014).

Os sistemas de banco de dados têm certas vantagens em relação ao sistema tradicional de armazenamento de arquivos. Eles são implementados com a função de isolar os detalhes internos do banco de dados do usuário, ou seja, promover a abstração desses dados e também permitir a relativa dependência dos dados e aplicativos que acessam (DEVMEDIA, 2014).

Outro fator importante é a questão da segurança e integridade dos dados, pois estes são geralmente criptografados e não são acessados tão facilmente. No entanto, a implantação de um sistema de banco de dados é mais cara e nem sempre é necessário usá-lo (DEVMEDIA, 2014).

Para realizar consultas, inserir, editar e vincular dados armazenados no banco de dados, é usada uma linguagem baseada em consultas estruturadas chamada SQL (*Structured Query Language*) (DEVMEDIA, 2014).

A importância em banco de dados foi abordada principalmente em disciplinas como Controle Inteligente. O banco de dados será utilizado para armazenar os dados dos empréstimos de equipamentos. SGBD utilizado será o MySQL, devido ao fato de estar presente no XAMPP, que é um pacote com os principais servidores de código aberto do mercado, utilizado para o desenvolvimento da interface *web* (TECHTUDO, 2018).

2.4 Sistema Embarcado

O sistema embarcado, também chamado de sistema embutido, é um sistema microprocessado em que um computador está anexado ao sistema que ele controla. Um sistema embarcado pode realizar um conjunto de tarefas que foram pré-definidas. O sistema é usado para tarefas específicas e, assim, por meio de engenharia é possível otimizar um determinado produto e diminuir o tamanho, bem como os recursos computacionais e o seu valor final (OFICINA DA NET, 2017).

Os sistemas embarcados estão por todo lado, e por essa razão, não se dá conta de sua capacidade computacional, já que se está envolvido com tais mecanismos (OFICINA DA NET, 2017). Há

uma grande variedade de processadores disponíveis no mercado, o que leva ao desenvolvimento de vários sistemas.

Há muitas restrições em sistemas embarcados comparando com os computadores convencionais. Entre elas, as restrições dimensionais, que envolvem tamanho e peso, são extremamente importantes em equipamentos pequenos, como telefones celulares. Outra restrição é o consumo de energia, que é extremamente importante em equipamentos móveis e é alimentado por baterias, como no caso dos módulos. Restrições de recursos, como memória e processamento, afetam o *design* do *software*. Outra restrição que pode ser citada é a da execução. Isso é relevante porque vários aplicativos devem ser executados em um momento muito específico.

O sistema embarcado é dedicado a uma única finalidade, ou a um pequeno conjunto de propósitos (COMPUTADOR PORTUGUÊS, 2017). Ele é dependente da sua aplicação.

Sistemas Embarcados foi abordado nas disciplinas como Sistemas Embarcados, Eletrônica Básica 1 e 2. Será utilizado este conceito para desenvolver o sistema que informará a localização do equipamento (RTLS) e também uma proposta que permitirá que o usuário retire/devolva o equipamento sem intermédio de um funcionário.

2.5 Código de Barras

Um código de barras é um método de representação de dados em um formato visual legível por máquina. Inicialmente, códigos de barras eram representados pela variação das larguras e espaçamentos das linhas paralelas. Esses códigos de barras, agora conhecidos como lineares ou unidimensionais (1D), podem ser digitalizados por *scanners* ópticos especiais, chamados leitores de código de barras. Posteriormente, variantes bidimensionais (2D) foram desenvolvidas, usando retângulos, pontos, hexágonos e outros padrões geométricos, chamados códigos de matriz ou códigos de barras 2D, embora não usem barras como tais. Os códigos de barras 2D podem ser lidos ou desconstruídos usando o *software* aplicativo em dispositivos móveis com câmeras embutidas, como *smartphones* (ELECTRONIC IMAGING, 2016).

A seguir, serão apresentados os tipos de códigos de barras atualmente disponíveis:

2.5.1 *Code* 39

Esse é um dos códigos de barras mais antigos do mercado e é uma simbologia comum encontrada em eletrônicos, serviços de saúde e governo (ELECTRONIC IMAGING, 2016). É um código alfanumérico linear, 1D, com a capacidade de incluir todo o conjunto de 128 caracteres ASCII e se estender a qualquer comprimento, limitado apenas pelo tamanho do rótulo. Esse é código padrão utilizado nas carteiras de estudante da UFES. Se o espaço é uma preocupação, o Código 128 seria uma escolha melhor a considerar (ELECTRONIC IMAGING, 2016).

A Figura 3 mostra um exemplo de código de barras do tipo *Code 39*.

Figura 3 – Código de barras do tipo *Code 39*

Fonte: Eletronic Imaging (2016).

2.5.2 *Code* 128

Derivado do conjunto de caracteres ASCII 128 (0-9, a-z, A-Z e alguns caracteres especiais), esse código de barras compacto é amplamente utilizado em aplicativos de embalagem e transporte em todo o mundo (ELECTRONIC IMAGING, 2016). O *Code 128* possui uma configuração de comutação automática que permite aos usuários otimizar o tamanho do código de barras (ELECTRONIC IMAGING, 2016).

A Figura 4 mostra um exemplo de código de barras do tipo *Code* 128.

Figura 4 – Código de barras do tipo *Code* 128

Fonte: Eletronic Imaging (2016).

2.5.3 *Interleaved* 2 *of* 5

Comumente encontrado em armazém, distribuição e fabricação, o *Interleaved 2 of 5* é um código de barras somente numérico usado para codificar pares de números. A cada dois dígitos são emparelhados para criar um símbolo (ELECTRONIC IMAGING, 2016). O número de dígitos usados deve ser par para que esse formato funcione; portanto, normalmente é adicionado um zero no final de um conjunto ímpar de números (ELECTRONIC IMAGING, 2016).

A Figura 5 mostra um exemplo de Código de barras do tipo *Code 2 of 5*.

Figura 5 – Código de barras do tipo *Code 2 of 5*

Fonte: Eletronic Imaging (2016).

2.5.4 *Universal Product Codes* (*UPC*)

Encontrados em quase todos os produtos de varejo, esses códigos de barras foram criados originalmente para supermercados para fornecer impressão rápida de recibos e rastreamento de estoque (ELECTRONIC IMAGING, 2016). Depois de garantir um número UPC, o fabricante receberá um número exclusivo da empresa para combinar com os números de produtos individuais (ELECTRONIC IMAGING, 2016).

A Figura 6 mostra um exemplo de código de barras do tipo UPC.

Figura 6 – Código de barras do tipo UPC

Fonte: Eletronic Imaging (2016).

2.5.5 *International Article Number* (*IAN*)

Considerados um superconjunto da UPC, esses códigos de barras são usados especificamente por livreiros, bibliotecas, universidades e atacadistas para rastreabilidade de livros. Esses códigos de 13 dígitos são criados a partir do ISBN (*International Standard Book Numbers*) para cada livro respectivo rastreado (ELECTRONIC IMAGING, 2016). Como os UPCs, eles são padronizados para a identificação exclusiva dos editores (ELECTRONIC IMAGING, 2016).

A Figura 7 mostra um exemplo de código de barras do tipo IAN.

Figura 7 – Código de barras do tipo IAN

Fonte: Eletronic Imaging (2016).

2.5.6 PDF417

Esse código de barras 2D linear e empilhado pode ser encontrado em muitos tipos de identificação, como na carteira de motorista. É também o padrão escolhido pelo USPS (Serviço Postal dos Estados Unidos) e pelo Departamento de Segurança Interna devido a seus recursos avançados, como codificação de *links* para mais de um arquivo de dados (ELECTRONIC

IMAGING, 2016). No entanto, ele pode ser expansivo em tamanho - quatro vezes maior que outros códigos de barras 2D, como *Datamatrix* e *QR Codes* (ELECTRONIC IMAGING, 2016).

A Figura 8 mostra um exemplo de código de barras do tipo PDF417.

Figura 8 – Código de barras do tipo PDF417

Fonte: Eletronic Imaging (2016).

2.5.7 Data Matrix

Este se tornou um dos códigos de barras 2D mais comuns. É um código de forma quadrada e pode codificar grandes quantidades de informações em um espaço muito pequeno; é muito popular na fabricação de eletrônicos e assistência médica por esse motivo (ELECTRONIC IMAGING, 2016). Os códigos 2D requerem *scanners* sofisticados, como *smartphones*, para basicamente digitalizar o código e traduzir a imagem inteira de uma só vez. Quando as empresas precisam de mais capacidade de armazenamento de códigos de barras, os códigos de barras 2D reinam supremos sobre seus equivalentes 1D (ELECTRONIC IMAGING, 2016).

A Figura 9 mostra um exemplo de Código de barras do tipo *Data Matrix*.

Figura 9 – Código de barras do tipo *Data Matrix*

Fonte: Eletronic Imaging (2016).

2.5.8 Quick Response (QR) Codes

Os *QR Codes* são uma versão moderna deste tipo de código e estão ganhando popularidade como ferramentas de marketing para vincular as informações baseadas na *web* (ELECTRONIC IMAGING, 2016). Não é tão compacto quanto o Data Matrix e você os encontrará frequentemente usados na publicidade de materiais e fachadas de lojas, com *links* para promoções ou detalhes especiais sobre um determinado produto (ELECTRONIC IMAGING, 2016).

A Figura 10 mostra um exemplo de Código de barras do tipo *Quick Response*.

Figura 10 – Código de barras do tipo *Quick Response*

Fonte: Eletronic Imaging (2016).

3 ALOCAÇÃO DE RECURSOS

No projeto foram elaboradas a interface *web* e simulações de circuitos elétricos no computador. A seguir, serão apresentados alguns recursos aplicados ao desenvolvimento do projeto.

3.1 XAMPP

O XAMPP é um pacote de soluções para servidores *web* de plataforma cruzada, gratuito e de código aberto, desenvolvido pela Apache Friends, consistindo principalmente no servidor HTTP Apache, banco de dados MariaDB e intérpretes para *scripts* escritos nas linguagens de programação PHP e Perl (QUORA, 2016).

XAMPP significa plataforma cruzada (X), Apache (A), MariaDB (M), PHP (P) e Perl (P) (QUORA, 2016). Esta plataforma apresenta uma interface leve e de simples implementação, diminuindo o tempo de desenvolvimento de um servidor *web* local para fins de teste, antes de hospedá-lo no servidor remoto, facilitando a transição para um servidor ativo.

O servidor *web* remoto que se usou neste projeto tinha os mesmos componentes que o XAMPP, o que tornava extremamente fácil a transição de um servidor de teste local para um servidor ativo.

3.2 MySQL Workbench

O MySQL Workbench é uma ferramenta de gerenciamento de banco de dados fácil de usar. Consegue-se realizar modelagem de dados, administração do sistema e uma variedade de outras tarefas necessárias para manter um aplicativo orientado a dados. Esse programa foi bem útil para modelar o banco de dados do sistema.

3.3 CakePHP

Hoje, existem muitas alternativas de código aberto acessíveis para as organizações desenvolverem seus aplicativos da *web*. Escolheu-se para este projeto o CakePHP que é um sistema em PHP que permite desenvolver aplicativos de *site* e *web* rapidamente.

Há muitas razões pelas quais foi utilizado o CakePHP para desenvolvimento do aplicativo:

- **Nenhuma configuração especial necessária:** A maior razão para o desenvolvimento do CakePHP ser popular é que, essa plataforma de código aberto não requer nenhum tipo de configuração antes dos programadores começarem a usá-la. No desenvolvimento do CakePHP, a maioria das configurações e recursos são detectados automaticamente (BRAINVIRE, 2018). Isso economiza muito tempo, pois os programadores não precisam mais estudar as definições de configuração.
- **Suporta arquitetura MVC:** Como o CakePHP é baseado em uma estrutura MVC, permite acessar, inserir, excluir e modificar dados diretamente do banco de dados. Um processo do controlador sempre responde rapidamente quando erros são identificados. Isso garante uma modificação rápida antes mesmo de interagir com o banco de dados (BRAINVIRE, 2018). Assim, o padrão MVC fornece uma plataforma de desenvolvimento segura e confiável.
- **Método Fácil para Gerenciamento de Banco de Dados:** O gerenciamento de banco de dados é bastante simplificado com a ajuda do recurso de integração CRUD (acrónimo do inglês *Create, Read, Update and Delete*) (BRAINVIRE, 2018). Esse recurso fornece funcionalidades adicionais para o trabalho rápido e eficaz de gerenciamento de banco de dados.
- **Capacidade de criar teste:** O CakePHP fornece excelente suporte para criar testes que ajudarão os desenvolvedores a verificar rapidamente problemas críticos em qualquer aplicativo e resolver os problemas presentes no sistema.
- **Grande suporte da comunidade *online*:** O CakePHP é suportado por uma grande comunidade, sempre pronta para compartilhar informações. Com tantos colaboradores, é sempre útil para os programadores aprenderem rapidamente novas ferramentas e recursos do CakePHP (BRAINVIRE, 2018). Além disso, o fórum bem estabelecido

também facilita o conhecimento sobre novos recursos e funcionalidades de desenvolvimento para proprietários e desenvolvedores de *sites*.

Com todas essas vantagens foi escolhido o CakePHP para o desenvolvimento do aplicativo *web*.

3.4 Github

O *GitHub* é um *site* e um serviço baseado em nuvem que ajuda o desenvolvedor a armazenar e gerenciar seu código, além de rastrear e controlar alterações em seu código (KINSTA, 2016). Para entender exatamente o que é o *GitHub*, precisa-se conhecer dois princípios:

- Controle de versão
- Git

O controle de versão ajuda o desenvolvedor a rastrear e gerenciar alterações no código de um projeto de *software* (KINSTA, 2016). À medida que um projeto de *software* cresce, o controle de versão se torna essencial.

O Git é um sistema de controle de versão distribuído, o que significa que toda a base de código e o histórico estão disponíveis no computador de todos os desenvolvedores, o que permite fácil ramificação e fusão (KINSTA, 2016).

O *GitHub* é uma empresa que oferece um serviço de hospedagem de repositório Git baseado em nuvem. Essencialmente, torna muito mais fácil para indivíduos e equipes usar o Git para controle de versão e colaboração (KINSTA, 2016).

Qualquer pessoa pode se inscrever e hospedar um repositório de código público gratuitamente, o que torna o *GitHub* especialmente popular em projetos de código aberto.

Durante esse projeto foi utilizado este serviço, o que permite compartilhar o código em diversos computadores e desenvolver o projeto sem precisar de uma máquina específica. Também se controlava as suas versões ao longo do tempo.

3.5 000webhost

A hospedagem na *web* é um serviço que permite que organizações e indivíduos publiquem um *site* ou página da *web* na *internet*. Um *host* ou provedor de serviços de hospedagem é uma empresa que fornece as tecnologias e serviços necessários para que o *site* ou a página da *web* seja visualizada na *internet*.

Os *site*s são hospedados ou armazenados em computadores especiais chamados servidores. Quando os usuários da *internet* desejam visualizar seu *site*, tudo o que precisam fazer é digitar o endereço ou domínio do *site* no navegador. O computador se conectará ao seu servidor e suas páginas serão entregues através do navegador (MAQUINA DE RESULTADOS, 2016).

A empresa *000webhost* oferece uma quantidade considerável de espaço em disco e largura de banda que são mais que suficientes para etapas de teste do projeto. Foi utilizado desta empresa uma hospedagem de domínio grátis que suporta as últimas versões do PHP e MySQL para o desenvolvimento desse projeto.

3.6 FileZilla

O *FileZilla* é uma ferramenta de *software* de protocolo de transferência de arquivos de código aberto (FTP) gratuita que permite que os usuários configurem servidores FTP ou se conectem a outros servidores FTP para trocar arquivos. O *FileZilla* tradicionalmente suporta o *File Transfer Protocol* sobre *Transport Layer Security* (FTPS) (TACHTARGET, 2017).

Para hospedar o *site* no servidor *online* da *000webhost*, foi utilizado FileZilla para transferir os arquivos necessários para o aplicativo funcionar.

3.7 Google Chrome *DevTools*

Com a explosão das estruturas Javascript, é muito importante que seja possível trabalhar com seu código *online* no navegador.

O painel do *DevTools* permite interação com o HTML, CSS e Javascript em execução em praticamente qualquer página da *internet*. O seu objetivo real é economizar tempo e angústia mental para os desenvolvedores. O *DevTools* economiza o tempo do desenvolvedor, permitindo que ele manipule seu código no navegador e que resolva rapidamente problemas de estilo e explore ideias diferentes (GALVANIZE, 2017).

Assim, ele foi essencial para o desenvolvimento do aplicativo *web*.

3.8 ZXing

O ZXing ("*zebra crossing*") é uma biblioteca de processamento de imagem de código de barras 1D / 2D de código aberto e multi-formato implementada em Java, com portas para outros idiomas. Uma biblioteca de processamento de imagem de código de barras 1D / 2D em vários formatos, utilizável no sistema JavaScript (ZXING, 2017).

Para o desenvolvimento desse projeto foi utilizada essa biblioteca para que seja possível a leitura do código de barras através da interface *web*.

3.9 Simular Câmera Para Leitura de Código de Barras

Foi necessário simular a câmera do celular durante o desenvolvimento do aplicativo *web* e os seguintes programas foram utilizados:

- ***DroidCam Source 3*****:** *DroidCam* transforma seu dispositivo Android em uma *webcam* sem fio para o seu PC. Ele pode ser usado com programas de bate-papo como o Skype, com programas de transmissão ao vivo como OBS ou XSplit ou em qualquer outro lugar em que você usaria normalmente uma *webcam*. Também pode acessar a *DroidCam* como uma câmera IP através de um navegador da *internet*, sem precisar instalar *software* adicional (TECHTUDO, 2014).

- ***SparkoCam Virtual webCam*****:** O *SparkoCam* é um *software* de *webcam* e efeitos de vídeo para transmitir e aplicar efeitos de *webcam* ao vivo em suas conversas e gravações de vídeo. Neste programa é possível mostrar a tela da sua área de trabalho e transmitir

o que está acontecendo na área de trabalho pela *webcam* (SPAKOCAM, 2015). Assim, era possível simular o aplicativo *web* sem utilizar uma *webcam* no computador.

3.10 Proteus

O Proteus é um *software* de *design* de circuitos eletrônicos que inclui módulos de captura esquemática, simulação e *layout* de PCB (placa de circuito impresso). Proteus está à frente na simulação dos circuitos que contêm os microcontroladores, onde se pode simular o circuito carregando o código hexadecimal no microcontrolador (QUORA, 2015).

Assim, ele foi essencial para simular os circuitos do projeto, onde foram utilizados microcontroladores e módulos de GPS e de transmissão sem fio presentes em suas bibliotecas.

3.11 Arduino

O Arduino é uma plataforma de código aberto usada na construção de projetos eletrônicos. O Arduino consiste em uma placa de circuito programável física e em um *software*, ou IDE (*Integrated Development Environment*) executado no computador, usado para escrever e fazer *upload* de código de computador na placa física.

Diferentemente da maioria das placas de circuito programáveis anteriores, o Arduino não precisa de uma peça de *hardware* separada (chamada de programador) para carregar um novo código na placa - você pode simplesmente usar um cabo USB (SPARKFUN, 2015). Finalmente, o Arduino fornece um fator de forma padrão que interrompe as funções do microcontrolador em um pacote mais acessível.

O Arduino foi utilizado para os módulos baseados em RTLS e nos módulos de ‘autoatendimento’.

3.12 PowerShell

O *PowerShell* é uma estrutura de tarefas automatizada da Microsoft (TECHOPEDIA, 2017). Ele automatiza o processamento em lote e cria ferramentas de gerenciamento do sistema. Ele

inclui mais de 130 ferramentas de linha de comando padrão para funções e permite que os administradores executem tarefas em sistemas Windows locais e remotos através do acesso ao COM (*Component Object Model*) e WMI (Instrumentação de Gerenciamento do Windows) (TECHOPEDIA, 2017).

Com o *PowerShell* foi possível criar um *script* que envia para o banco de dados os valores da posição do equipamento obtidos pelos módulos baseados em RTLS.

3.13 Arquivos '.bat'

Os arquivos .bat são chamados de Arquivos em lote, que são o conjunto de comandos do Windows que você pode executar clicando duas vezes no ícone do arquivo e executa os comandos no DOS (QUORA, 2010).

Os comandos do menu Executar do Windows também podem ser executados usando arquivos em lote. Se for preciso fazer algo repetidamente, pode-se criar um arquivo em lotes e, quando precisar executar esse comando, basta executar o arquivo em lotes. Não precisa digitar o comando toda vez.

Para o módulo de 'autoatendimento' foi utilizado um arquivo em lote para acionar a entregar o equipamento ou o *kit* pois é preciso acessar o banco de dados remoto.

3.14 Virtual Serial Port Driver por Eltima Software

O *Virtual Serial Port Driver* é um aplicativo simples que permite criar pares de portas COM virtuais. As portas seriais virtuais criadas no *Virtual Serial Port Driver* se comportam como se fossem portas reais, para que os aplicativos possam se comunicar e transferir dados via cabo de modem nulo virtual (FILEHIPPO, 2018).

O *Virtual Serial Port Driver* possui uma interface amigável que permite criar rapidamente portas seriais com qualquer nome que você precisar (não se limita aos nomes COMx) (FILEHIPPO, 2018).

Com o *Virtual Serial Port Driver* foi possível integrar os circuitos simulados pelo Proteus com os *scripts* desenvolvidos no *PowerShell* e nos arquivos em lote.

3.15 XBee

Os módulos XBee são pequenos dispositivos de radiofrequência (RF) que transmitem e recebem dados pelo ar usando sinais de rádio. A capacidade sem fio é essencial sempre que se deseja colocar sensores onde nenhum cabo consiga ser instalado ou onde tal amarração seja indesejável (DIGI INTERNATIONAL, 2019).

Os dispositivos XBee são altamente configuráveis e suportam vários protocolos, o que permite escolher a tecnologia para sua aplicação – quando se deseja configurar um par de rádios para trocar dados ou projetar uma rede de malha grande com vários dispositivos (DIGI INTERNATIONAL, 2019).

Os dispositivos XBee se comunicam pelo ar, enviando e recebendo mensagens sem fio. Os dispositivos transferem apenas essas mensagens sem fio; eles não podem gerenciar os dados recebidos ou enviados. No entanto, eles podem se comunicar com dispositivos inteligentes através da interface serial (DIGI INTERNATIONAL, 2019).

Os dispositivos XBee transmitem dados provenientes da entrada serial pelo ar e enviam os dados recebidos à saída serial. Seja para fins de comunicação ou simplesmente para configurar o dispositivo, uma combinação dos dois processos possibilita a comunicação com o XBee (DIGI INTERNATIONAL, 2019). Dessa maneira, dispositivos inteligentes, como microcontroladores ou PCs, podem controlar o que o dispositivo XBee envia e gerenciar as mensagens sem fio recebidas (DIGI INTERNATIONAL, 2019).

Com essas informações, pode-se identificar os dois tipos de transmissão de dados sem fio em um processo de comunicação XBee:

- Comunicação sem fio: essa comunicação ocorre entre os módulos XBee. Os módulos que devem trabalhar juntos precisam fazer parte da mesma rede e devem usar a mesma

frequência de rádio. Todos os módulos que atendem a esses requisitos podem se comunicar sem fio (DIGI INTERNATIONAL, 2019).

- Comunicação serial: essa comunicação ocorre entre o módulo XBee e o dispositivo inteligente conectado a ele através da interface serial (DIGI INTERNATIONAL, 2019).

4 METODOLOGIA E ETAPAS DE DESENVOLVIMENTO

Cada etapa de desenvolvimento é caracterizada por um conjunto distinto de atividades que levam o projeto desde a primeira ideia até sua conclusão. Cada estágio é de igual importância e contribui para o sucesso geral do projeto.

As etapas de desenvolvimento serão abordadas nas seções a seguir.

a) **Estabelecer e Revisar as Propriedades do Sistema:** Alguns requisitos e características do sistema foram definidos, para que as atividades que compõem o projeto sejam melhores direcionadas;
b) **Instalação dos pacotes, programas e sistema operacional para o desenvolvimento do projeto:** Antes de iniciar o projeto, foi necessário instalar os programas como citados anteriormente para a simulação do *hardware* (programação dos microcontroladores, placas de circuito impresso, simulações dos circuitos) e para o desenvolvimento da interface *web*, banco de dados e simulações;
c) **Modelagem e implementação do banco de dados do sistema:** Foi modelado um banco de dados com todas as tabelas com colunas e relações necessárias para atender o objetivo do projeto. Para esta tarefa foi utilizado o programa MySQL *Workbench* devido à sua facilidade para gerenciamento de banco de dados;
d) **Desenvolvimento da interface *web*:** Com o banco de dados modelado, foi iniciado o desenvolvimento da interface *web* do sistema. Para a elaboração dessa interface foram utilizadas as seguintes ferramentas:
    - **XAMPP:** Usava-se como um servidor da *web* local para fins de teste e implantação, antes de colocá-lo no servidor *online*;
    - **CakePHP:** Permitiu desenvolver o aplicativo *web* rapidamente;
    - ***Github:*** Ajudou armazenar e gerenciar seu código, além de rastrear e controlar alterações do código do aplicativo;
    - ***000webhost:*** Utilizou-se para hospedar o projeto, é o servidor *online*;
    - ***FileZilla:*** Utilizado para transferir os arquivos para o servidor *online*;
    - **Google Chrome *DevTolls*:** Ajudou no desenvolvimento da parte de HTML, CSS e Javascript da página *web*.

e) **Leitor de código de barras:** Foi desenvolvida uma página *web* responsável pela leitura do código de barras do equipamento e da carteira do estudante que identifica a sua matrícula. Nesta interface, foi utilizada a biblioteca de processamento de imagem de código de barras ZXing e os programas *DroidCam Source* e *SparkoCam Virtual WebCam* para realizar as simulações e testes do *site*;

f) **Simulação do Módulo RTLS:** Foi simulada a eletrônica referente ao módulo que proverá informações de localização dos equipamentos ao banco de dados. Para esta etapa foi utilizado o programa Proteus e suas bibliotecas que contém Arduino e XBee. Também foi utilizado *PowerShell* para executar um *script* que permita atualizar o banco de dados do servidor *online*. Além disso foi utilizado o programa *Virtual Serial Port Driver* por *Eltima Software* para criar portas COM virtuais, integrando o *script* do *Powershell* e o projeto simulado no Proteus;

g) **Simulação do Módulo de 'Autoatendimento':** Foi simulada a eletrônica referente ao módulo que permitirá ao usuário retirar/ entregar o equipamento ou *Kit* sem intermédio de uma outra pessoa. Para esta etapa foi utilizado o programa Proteus e suas bibliotecas que contém Arduino. Também foram utilizados arquivos '.bat' para executar um *script* que permita acessar o banco de dados do servidor *online*. Além disso, foi utilizado o *programa Virtual Serial Port Driver* por *Eltima Software* para criar portas COM virtuais, integrando o *script* do *Powershell* e o projeto simulado no Proteus;

h) **Testes e validações:** Com o sistema em operação, foram realizados testes e verificações para validar sua operação de maneira estável. Com isso, o sistema foi validado e dado como pronto para ser utilizado no controle de empréstimos de equipamentos;

i) **Lições Aprendidas:** Registro de tudo que foi aprendido durante o desenvolvimento do projeto.

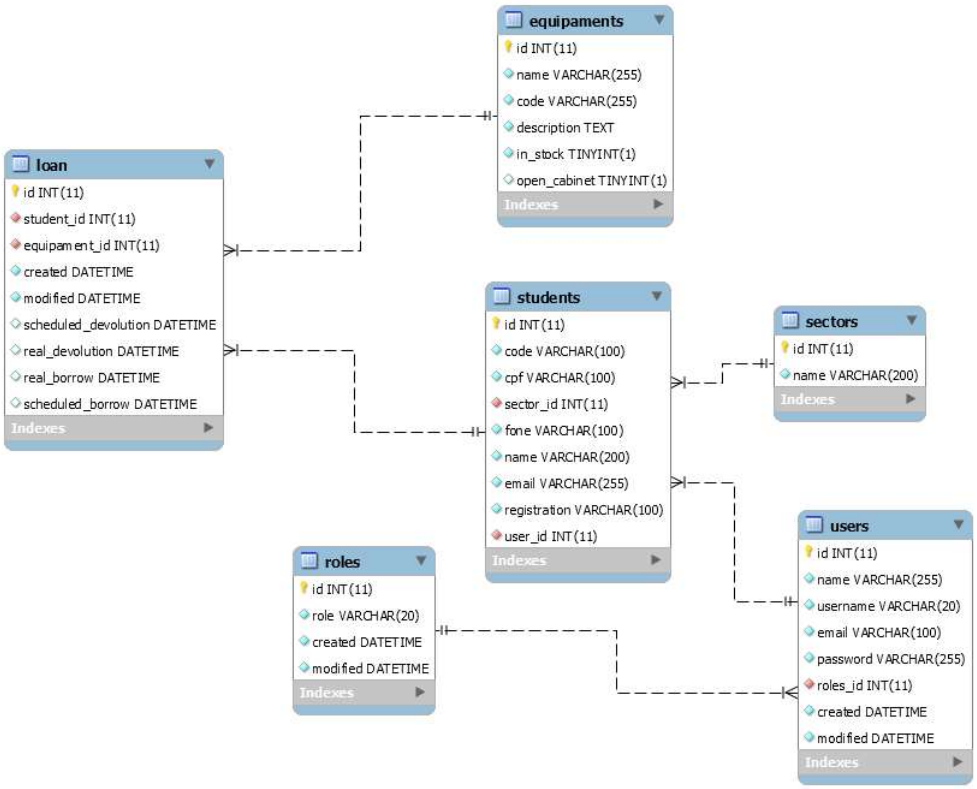
5 APRESENTAÇÃO DO SISTEMA

O sistema final foi dividido em quatro partes: Banco de Dados (funcional), Interface de Aquisição/ Vínculo (funcional), Módulo de Localização (Simulado) e Módulo de 'Autoatendimento'.

5.1 Banco de Dados

A Figura 11 apresenta a configuração do banco de dados utilizado.

Figura 11 – Esquemático das tabelas do banco de dados e como elas se relacionam



Fonte: Produção do próprio autor.

Como pode observar na Figura 11, o sistema apresenta as seguintes tabelas com as suas colunas:

- *Loan:* Ela armazena os empréstimos.
    - *id:* Esta coluna recebe o valor inteiro auto incrementado que identifica o empréstimo na tabela;

- *student_id*: Recebe valor inteiro referente à coluna *id* da tabela dos estudantes. Assim, guarda o estudante que pegou o equipamento;
- *equipament_id*: Recebe valores de números referentes ao *id* da tabela dos equipamentos. Assim, guarda o equipamento do empréstimo;
- *created*: Recebe o valor da data e do horário que o dado do empréstimo foi inserido;
- *modified*: Recebe o valor da data e do horário que os dados do empréstimo foram modificados;
- *scheduled_devolution*: Recebe o valor da data e do horário que o equipamento ou *kit* deverá ser devolvido;
- *real_devolution*: Recebe o valor da data e do horário que o equipamento ou *kit* foi realmente devolvido;
- *real_borrow*: Recebe o valor da data e do horário que o equipamento ou *kit* foi realmente emprestado;
- *scheduled_borrow*: Recebe o valor da data e do horário que o equipamento ou *kit* deverá ser emprestado.

- *Equipments*: Ela guarda todos os equipamentos ou *kits* presentes no almoxarifado.
  - *id*: Esta coluna recebe o valor inteiro auto incrementado que identifica o equipamento ou *kit* na tabela;
  - *name*: Recebe valor em *string* de caracteres que se refere ao nome do equipamento ou *kit*;
  - *code*: Recebe o código em *string* de caracteres do código de barras que identifica o equipamento ou *kit*;
  - *description*: Recebe o valor em texto que se refere a descrição do equipamento ou *kit*;
  - *in_stock*: Esta coluna indica se o equipamento ou *kit* está em estoque ou não, ou seja, se ele está emprestado ou não;
  - *open_cabinet*: Refere-se permissão ou não para o usuário abrir o armário para pegar o equipamento no módulo de autoatendimento.

- *Students*: Ela guarda todos os estudantes que usam os laboratórios e os equipamentos.

- *id*: Esta coluna recebe o valor inteiro auto incrementado que identifica o estudante na tabela;
- *code*: Recebe valor em *string* de caracteres referente ao código de barras que identifica a matrícula do estudante;
- *cpf*: Recebe *string* de caracteres que é o CPF do estudante;
- *sector_id*: Recebe valores de números referentes ao *id* da tabela de Setores (Engenharia Mecânica, Engenharia Elétrica...). Assim, guarda qual o curso superior que o estudante faz;
- *fone*: Recebe valor em *string* de caracteres referente ao telefone do estudante;
- *name*: Recebe valor em *string* de caracteres referente ao nome do estudante;
- *email*: Recebe valor em *string* de caracteres referente ao e-mail do estudante;
- *registration*: Recebe um valor em *string* de caracteres referente a algum outro registro do estudante;
- *user_id*: Recebe valores de números referentes ao *id* da tabela de Usuários. Assim, guarda qual usuário da interface *web* que o estudante é. Essa coluna não precisa ser preenchida, ou seja, um aluno não precisa ser um usuário.

- *Sectors*: Esta tabela guarda quais são os cursos superiores dos alunos que são atendidos no laboratório;
  - *id*: Esta coluna recebe o valor inteiro auto incrementado que identifica o curso na tabela;
  - *name*: Recebe valor em *string* de caracteres que se refere ao nome do curso;

- *Users*: Ela guarda todos os usuários que têm acesso à interface *web*. Ele pode ser aluno, funcionário do laboratório ou professor.
  - *id*: Esta coluna recebe o valor inteiro auto incrementado que identifica o usuário na tabela;
  - *name*: Recebe valor em *string* de caracteres referente ao nome do usuário;
  - *username*: Recebe valor em *string* de caracteres referente ao nome do usuário para fazer o login;
  - *email*: Recebe valor em *string* de caracteres referente ao e-mail do usuário;
  - *password*: Recebe valor em *string* de caracteres referente à senha criptografada do usuário;

- *roles_id*: Recebe valores de números referentes ao *id* da tabela de tipos de usuários. Assim, guarda qual o tipo o usuário é;
- *created*: Recebe o valor da data e do horário que o usuário foi criado;
- *modified*: Recebe o valor da data e do horário que os dados do usuário foram modificados;

- *Roles*: Ela guarda quais são os tipos dos usuários. Ele pode ser administrador ou moderador.
  - *id*: Esta coluna recebe o valor inteiro auto incrementado que identifica o tipo do usuário na tabela;
  - *role*: Recebe valor em *string* de caracteres referente ao nome do tipo de usuário;
  - *created*: Recebe o valor da data e do horário que o usuário foi criado;
  - *modified*: Recebe o valor da data e do horário que os dados do usuário foram modificados;

A seguir serão apresentadas as interações que são feitas diretamente ao Banco de Dados através da interface *web*.

5.1.1 Página Inicial

Acessando o *site* http://automatik.000webhostapp.com/AutomaTIK/, o usuário tem acesso à página inicial do aplicativo. Nele são apresentados os próximos empréstimos até 2 horas e *kits* e equipamentos não emprestados. Esses relatórios são obtidos através do banco de dados como mostrado anteriormente. Essa funcionalidade pode ser visualizada na Figura 12.

Figura 12 – Página inicial do aplicativo *web*

Fonte: Produção do próprio autor.

Além disso, na página inicial, pode-se acessar as seguintes funções do aplicativo *web* através do menu lateral:

- *New User*
- *Login*
- Novo Empréstimo
- Devolução

Cada funcionalidade será explicada na sequência.

Se o usuário estiver logado como uma conta administrativa, as Figuras 13 e 14 mostram as telas iniciais relativas a este acesso.

Figura 13 – Página inicial do aplicativo *web* quando logado em uma conta administrativa - Parte 1/2

Fonte: Produção do próprio autor.

Figura 14 – Página inicial do aplicativo *web* quando logado em uma conta administrativa – Parte 2/2

Fonte: Produção do próprio autor.

Além dos próximos empréstimos até 2 horas e *kits* e equipamentos não emprestados, a página inicial quando logado mostra os empréstimos atrasados e os empréstimos atuais. Esses relatórios são obtidos através dos bancos de dados como mostrado anteriormente.

Cabe ressaltar que o menu lateral disponibiliza funcionalidades adicionais:

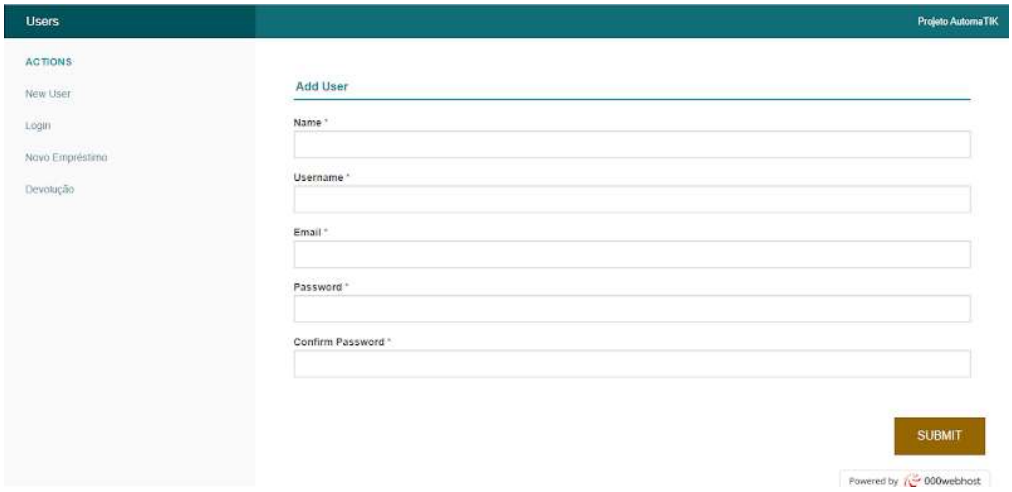
- Todos os Setores
- *Download* Todos os Setores

- Adicionar um Setor
- Todos os Usuários
- *Download* Todos os Usuários
- Add um Usuário
- Todos os Empréstimos
- *Download* Todos os Empréstimos
- Adicionar um Empréstimo
- Todos os Estudantes
- *Download* Todos os Estudantes
- Adicionar um Estudante
- Todos os *Kits* e Equipamentos
- *Download* Todos os *Kits* e Equipamentos
- Adicionar *Kits* e Equipamento
- Ler o Código de Barras

5.1.2 *New User* ou Add um Usuário

Essa página se encontra no seguinte *site*: http://automatik.000webhostapp.com/AutomaTIK/users/add, ou clicando em 'New User', na barra lateral de navegação quando não está logado ou em 'Add um Usuário' quando estiver logado em uma conta administrador. A página pode ser visualizada na Figura 15.

Figura 15 – Página do aplicativo *web* para adicionar um novo usuário



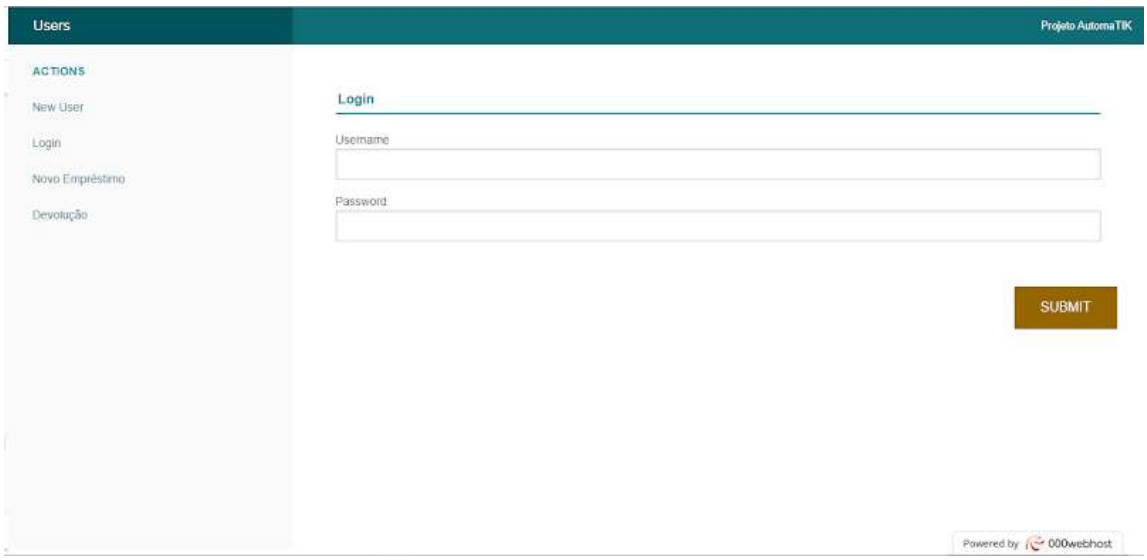
Fonte: Produção do próprio autor.

Para criar o usuário, devem ser inseridas informações de nome, nome de usuário, e-mail, senha e a confirmação da senha. Cabe destacar que quando é criado um usuário para esse *site*, o usuário não é um usuário administrador. É preciso acessar o banco de dados para torná-lo um usuário administrador.

5.1.3 Login

A página de *login* pode ser acessada no seguinte *site* http://automatik.000webhostapp.com/AutomaTIK/users/login, ou apertando em '*Login*' na barra lateral de navegação. O usuário é direcionado a uma página conforme apresentado na Figura 16.

Figura 16 – Página do aplicativo *web* para *login*



Fonte: Produção do próprio autor.

Para se logar no sistema, o usuário deve inserir o nome da conta e sua senha.

5.1.4 Todos os Setores

Para acessar essa página, é preciso estar logado e acessar a opção 'Todos os Setores' na barra de navegação lateral e terá acesso à página apresentada na Figura 17.

Figura 17 – Página do aplicativo *web* para visualizar todos os setores

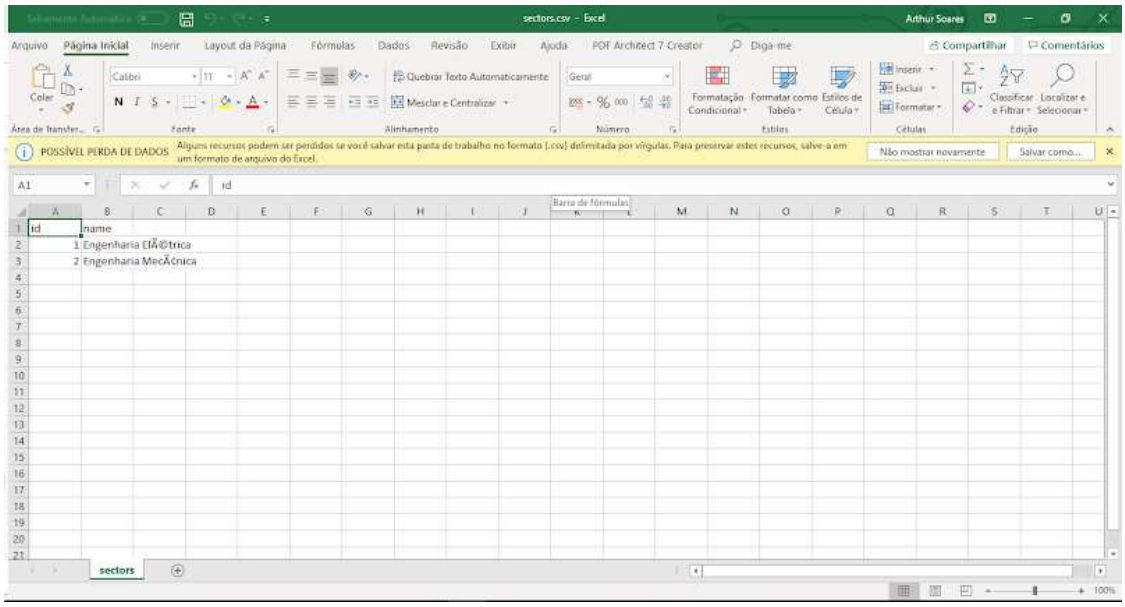
Fonte: Produção do próprio autor.

Essa página mostra todos os cursos superiores que foram registrados no banco de dados.

5.1.5 *Download* Todos os Setores

Esta opção, disponível na barra lateral, gera um arquivo Excel com todos os setores como mostrado a Figura 18.

Figura 18 – Arquivo Excel com todas os setores cadastrados



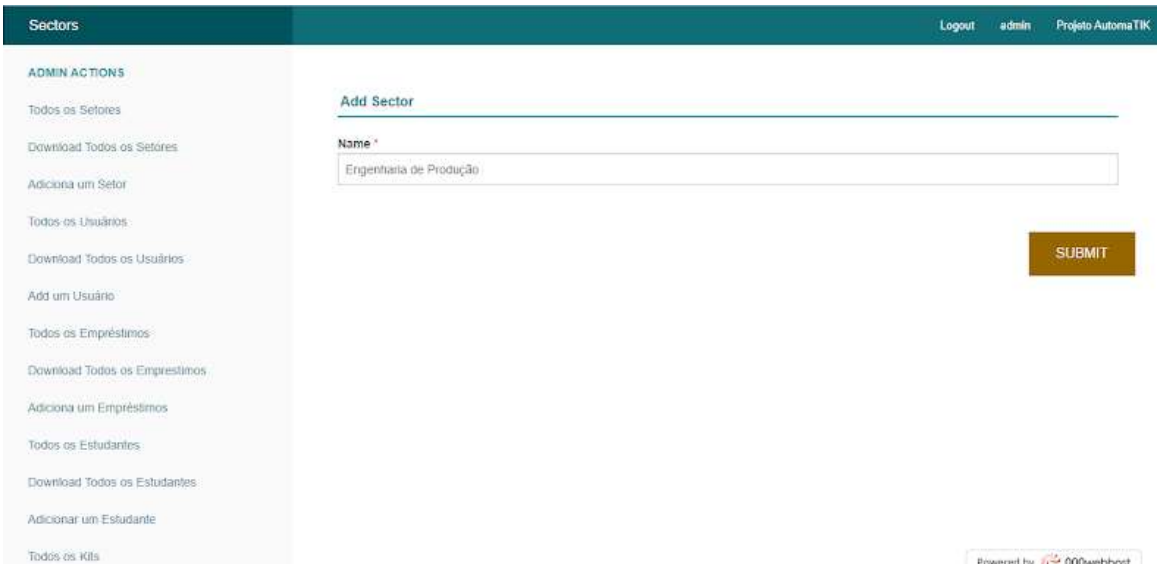
Fonte: Produção do próprio autor.

Para ter acesso a essa função, você terá que estar logado em uma conta do tipo administrador.

5.1.6 Adicionar um Setor

Para acessar essa página, é preciso estar logado como administrador. A página é apresentada na Figura 19.

Figura 19 – Página do aplicativo *web* para adicionar um novo setor



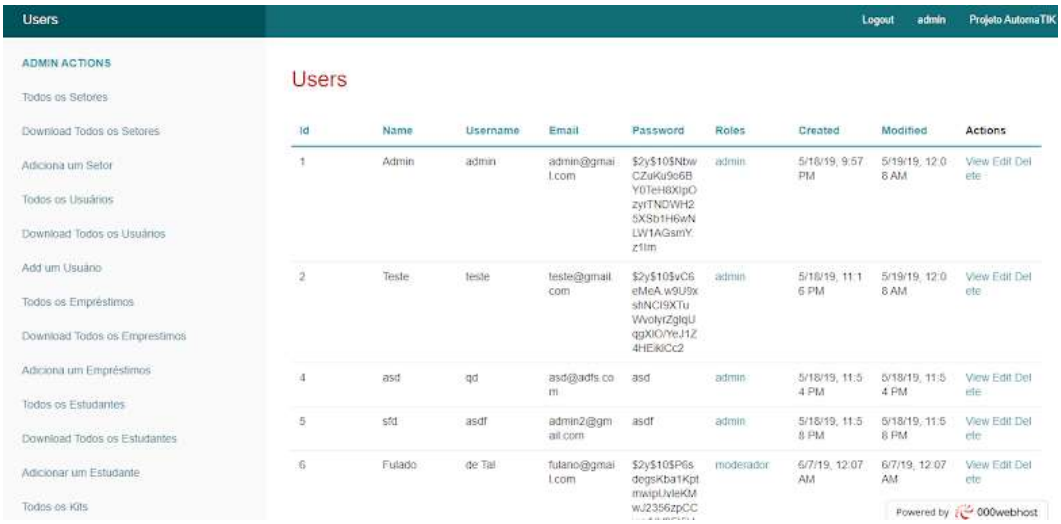
Fonte: Produção do próprio autor.

Essa página serve para adicionar um novo curso com o nome referido.

5.1.7 Todos os Usuários

Para acessar essa página, é preciso estar logado em uma conta administrador. A página apresentada na Figura 20.

Figura 20 – Página do aplicativo *web* mostrando a lista de usuários



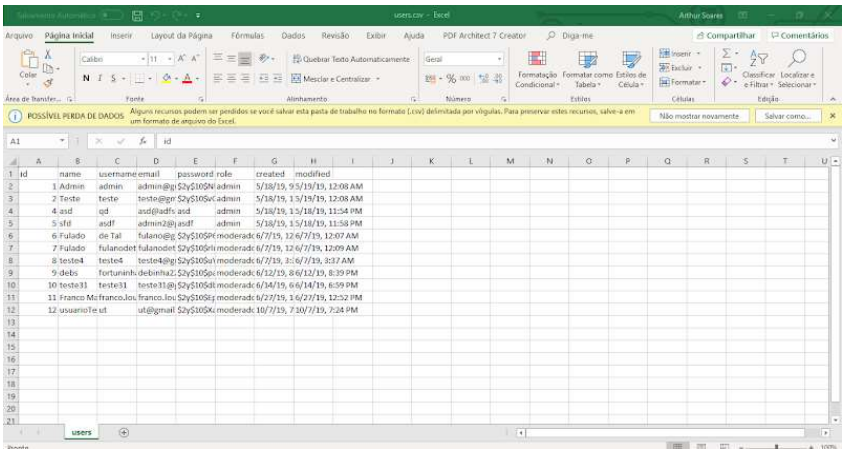
Fonte: Produção do próprio autor.

Nessa página podemos ver todos os usuários. Pode-se perceber que a senha fica criptografada. Ele apresenta o nome, *username*, e-mail, senha, tipo de usuário, a data e hora que foi criado e a data e hora que foi modificado.

5.1.8 *Download* Todos os Usuários

Esta função, disponível na barra de navegação lateral, gera um arquivo Excel com todos usuários como mostrados na Figura 21.

Figura 21 – Arquivo Excel com todos os usuários



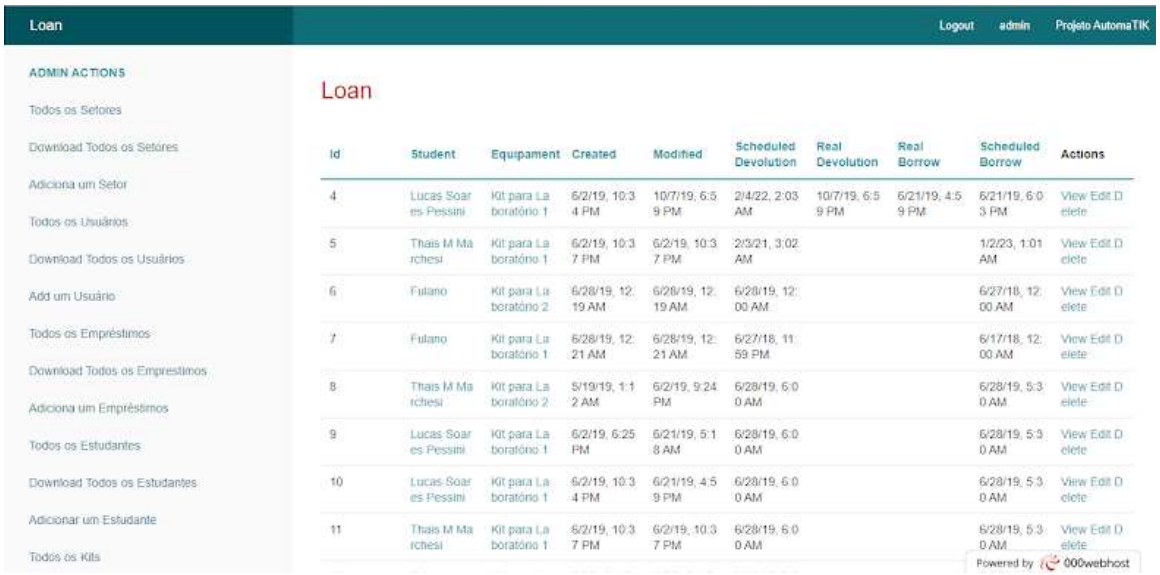
Fonte: Produção do próprio autor.

Para ter acesso a essa função, o usuário deve estar logado em uma conta do tipo administrador.

5.1.9 Todos os Empréstimos

Para acessar essa função, é preciso estar logado em uma conta administrador. A página apresentada na Figura 22.

Figura 22 – Página *web* que mostra todos os empréstimos



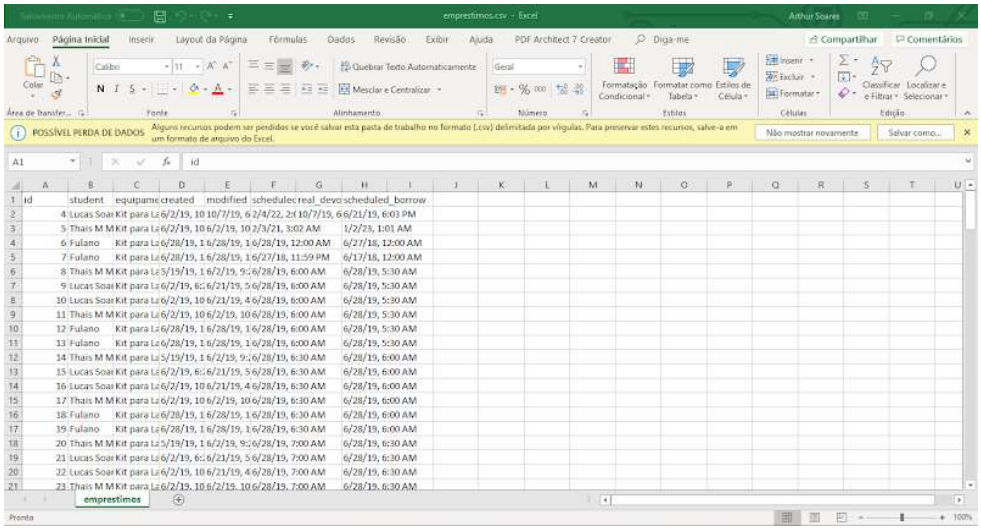
Fonte: Produção do próprio autor.

Nessa página podemos ver todos os empréstimos presentes na tabela onde mostra qual foi a pessoa que fez o empréstimo, o equipamento emprestado, quando foi criado, quando foi modificado, a data e hora que é real e programado para devolver e retirar o equipamento.

5.1.10 *Download* Todos os Empréstimos

Esta função gera um arquivo Excel com o nome ‘emprestimos.csv’, contendo todos os usuários como mostrado a Figura 23.

Figura 23 – Arquivo Excel com todos os empréstimos



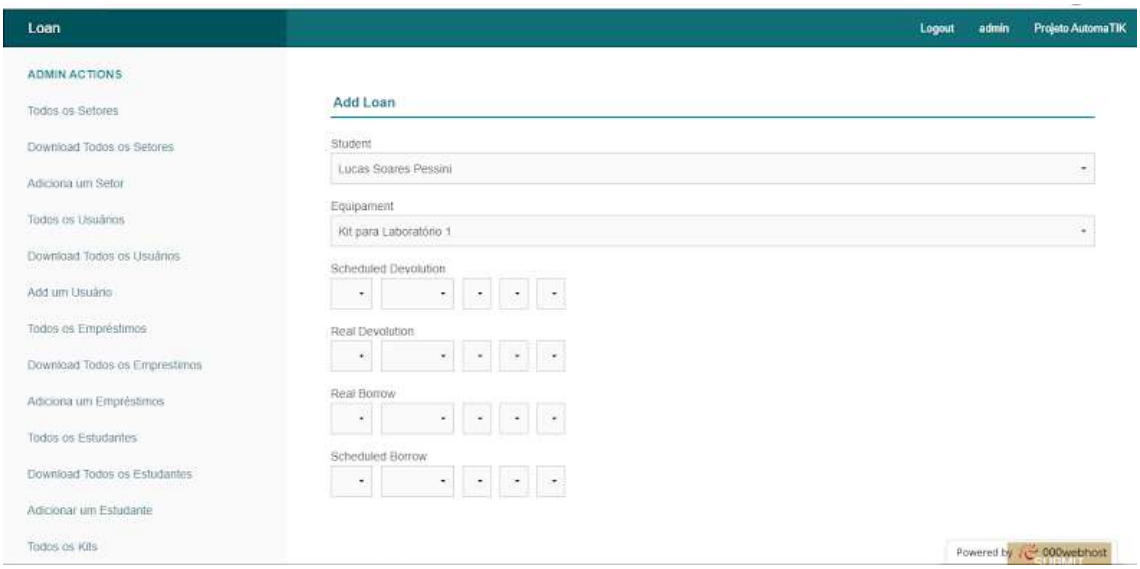
Fonte: Produção do próprio autor.

Para ter acesso a essa função, você terá que estar logado em uma conta do tipo administrador.

5.1.11 Adicionar um Empréstimo

Para acessar essa página, é preciso estar logado. A página é apresentada na Figura 24.

Figura 24 – Página para adicionar um novo empréstimo



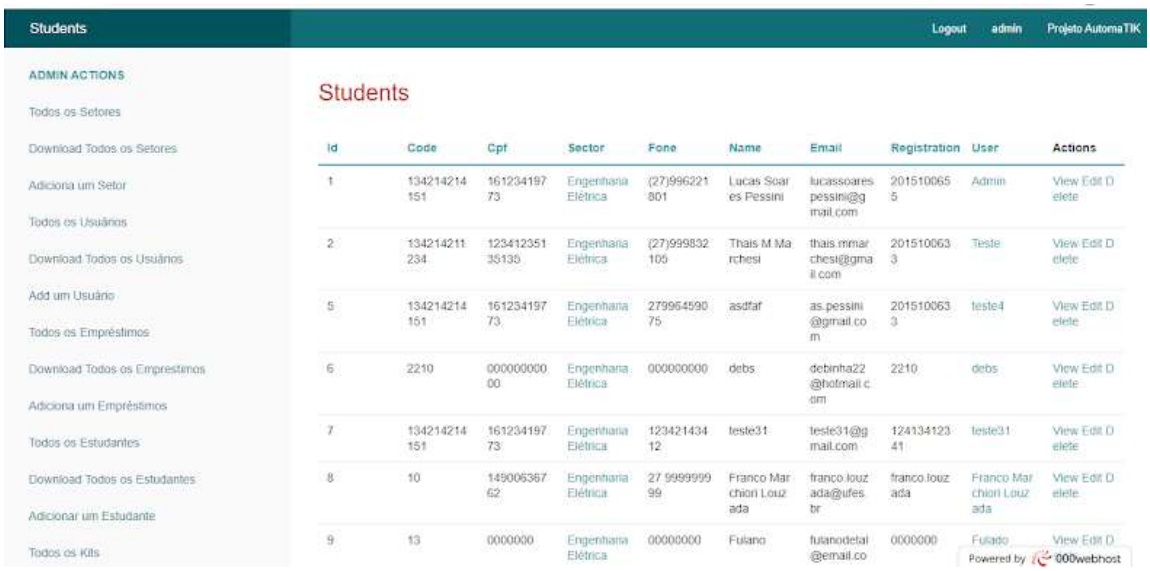
Fonte: Produção do próprio autor.

Essa página serve para adicionar um novo empréstimo. Mas há um outro método de adicionar os empréstimos que é através dos códigos de barras que será abordado posteriormente.

5.1.12 Todos os Estudantes

Para acessar essa página, é preciso estar logado em uma conta administrador. A página é apresentada na Figura 25.

Figura 25 – Página para visualizar todos os estudantes



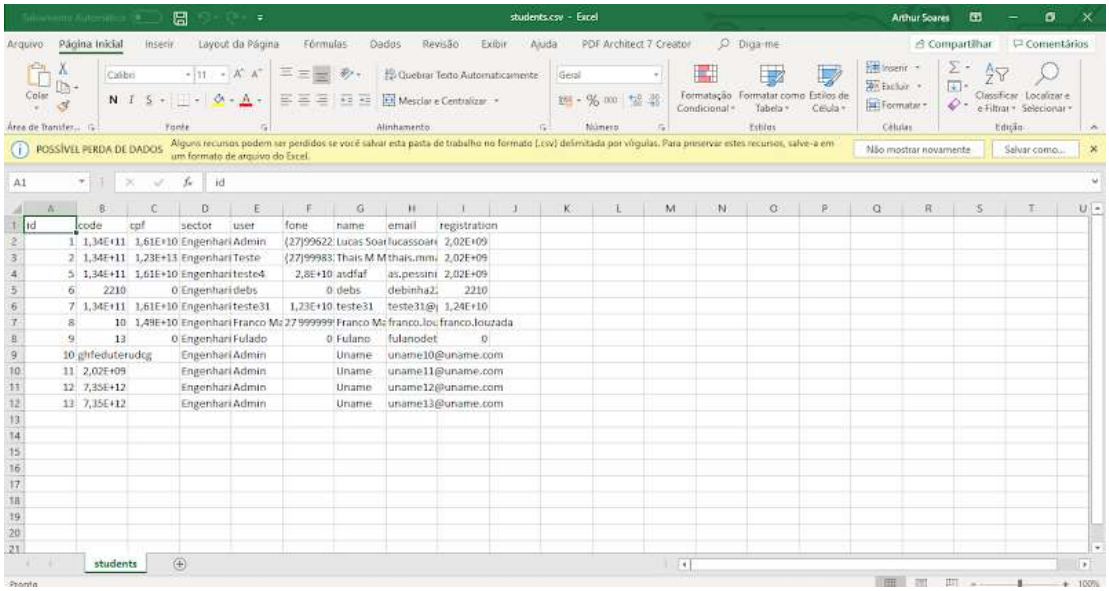
Fonte: Produção do próprio autor.

Nessa página podemos ver todos os estudantes registrados e seus dados que estão presentes na tabela do banco de dados. Também nesta página é possível editar e deletar um estudante.

5.1.13 *Download* Todos os Estudantes

Esta função gera um arquivo Excel com o nome ‘students.csv’, contendo todos os usuários como mostrado na Figura 26.

Figura 26 – Arquivo Excel com todos os estudantes



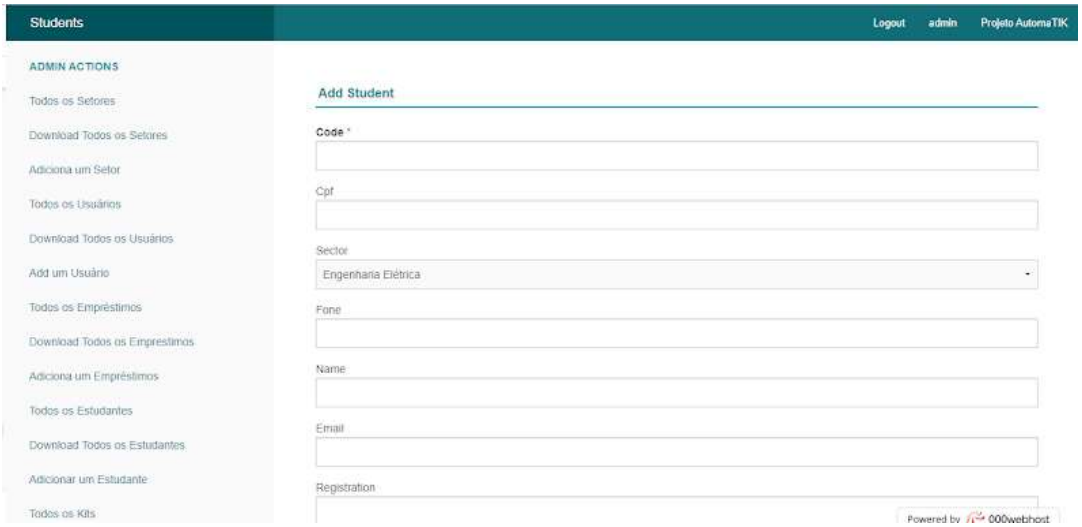
Fonte: Produção do próprio autor.

Para ter acesso a essa função, você terá que estar logado em uma conta do tipo administrador.

5.1.14 Adicionar um Estudante

Para acessar essa página, é preciso estar logado. A página apresentada na Figura 27.

Figura 27 – Página *web* para adicionar um novo estudante



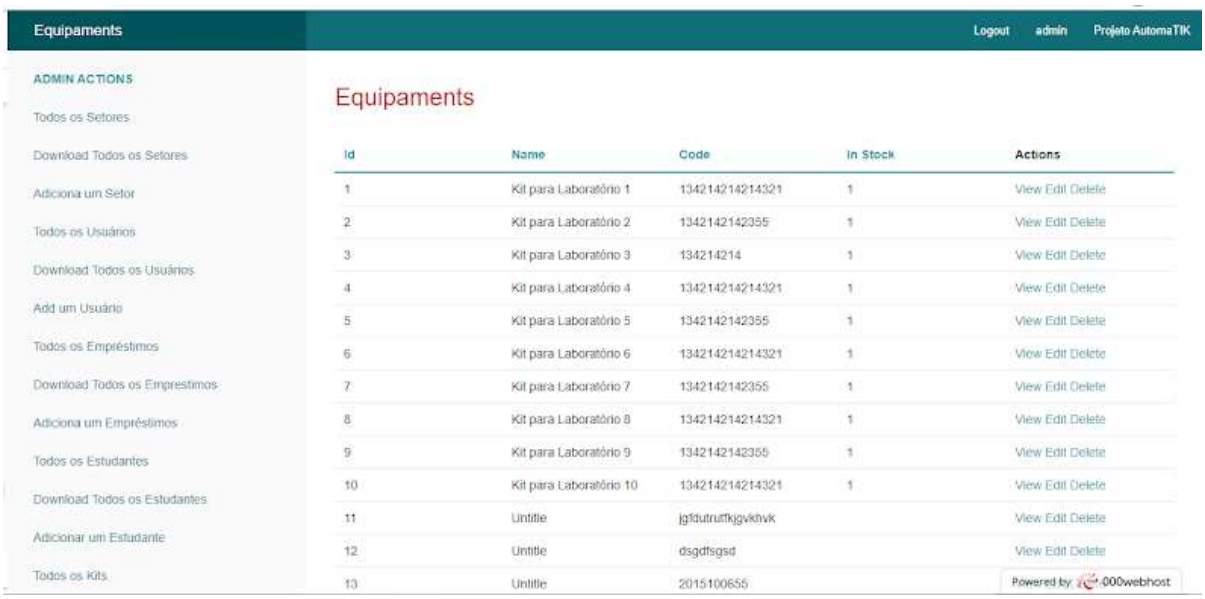
Fonte: Produção do próprio autor.

Essa página serve para adicionar um novo estudante. Devem ser inseridas informações de sua matrícula que está no código de barras, CPF, escolher o curso, o telefone, o nome e o e-mail.

5.1.15 Todos os *Kits* e Equipamentos

Para acessar essa página, é preciso estar logado em uma conta administrador. A página apresentada na Figura 28.

Figura 28 – Página *web* para visualizar todos os equipamentos e *kits*



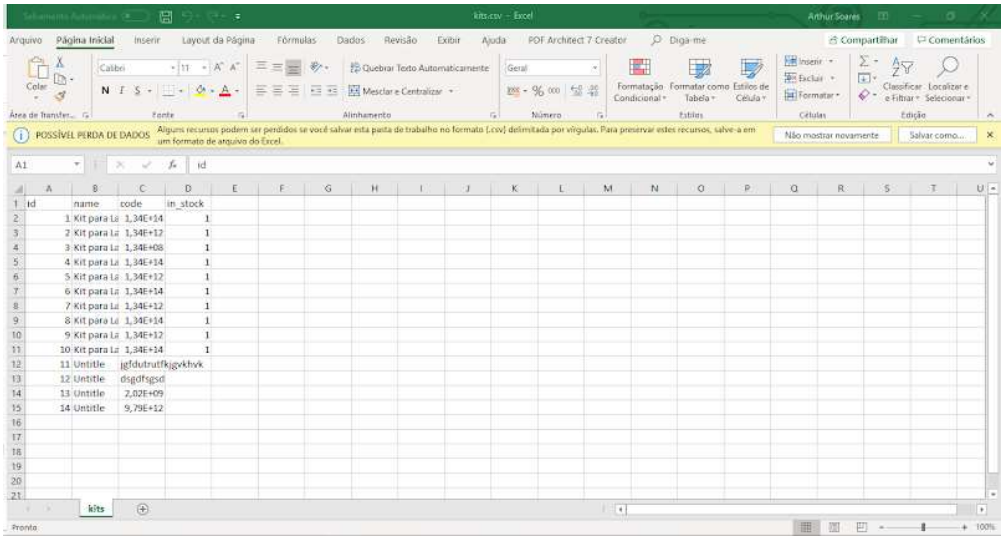
Fonte: Produção do próprio autor.

Nessa página podemos ver todos os *kits* e equipamentos e podemos ver também os seus dados: nome, código de barras e se ele está em estoque ou não. Consegue-se também editar e deletar.

5.1.16 *Download* Todos os *Kits* e Equipamentos

Esta função gera um arquivo Excel com o nome ‘kits.csv’ contendo todos os usuários como mostrado na Figura 29.

Figura 29 – Arquivo Excel com todos os *kits* e equipamentos



id	name	code	in_stock
1	Kit para La	1,34E+14	1
2	Kit para La	1,34E+12	1
3	Kit para La	1,34E+08	1
4	Kit para La	1,34E+14	1
5	Kit para La	1,34E+12	1
6	Kit para La	1,34E+14	1
7	Kit para La	1,34E+12	1
8	Kit para La	1,34E+14	1
9	Kit para La	1,34E+12	1
10	Kit para La	1,34E+14	1
11	Untitle	jgfdutrutfkjgvkhvk	
12	Untitle	dsgdfsgsd	
13	Untitle	2,02E+09	
14	Untitle	9,79E+12	

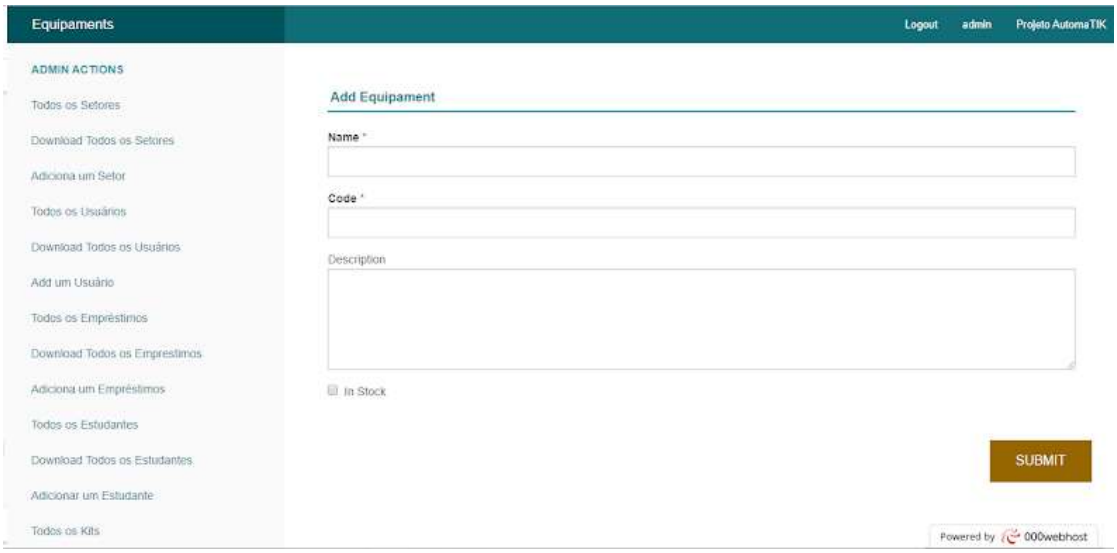
Fonte: Produção do próprio autor.

Para ter acesso a essa função, você terá que estar logado em uma conta do tipo administrador.

5.1.17 Adicionar um *Kit* ou Equipamento

Para acessar essa página, é preciso estar logado. A página apresentada na Figura 30.

Figura 30 – Página *web* para adicionar um novo *kit* ou equipamento



Fonte: Produção do próprio autor.

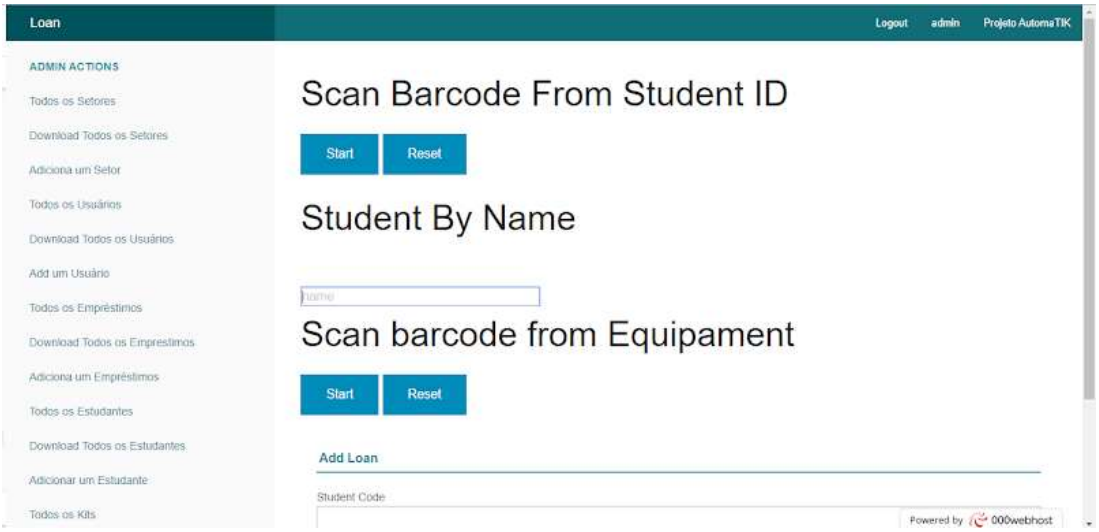
O usuário insere o nome do Equipamento ou *Kit*, o código de barras, descrição e se ele está em estoque.

5.2 Interface de Aquisição/Vínculo

Para acessar essa página, é preciso estar logado. Na função “Ler o código de barras”, o usuário tem acesso à página apresentada na Figura 31.

Para esse método funcionar não é preciso cadastrar previamente o aluno, equipamento ou *kit*. Caso não haja cadastro do estudante ou do equipamento na hora do empréstimo, ele insere automaticamente no banco de dados o novo aluno, equipamento ou *kit*, mas somente com o dado do código de barras. Isso permite que não ocorram filas e congestionamentos na hora de fazer os empréstimos.

Figura 31 – Página *web* para criar vínculo entre o estudante e o equipamento ou *kit*



Fonte: Produção do próprio autor.

Essa página foi feita para ser utilizada por um celular, pois essa página é capaz de ler o código de barras disponível na imagem da câmera. O procedimento é o seguinte:

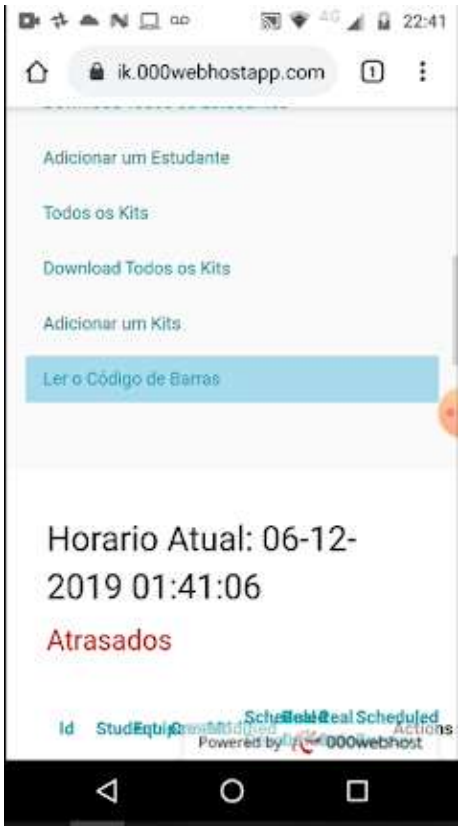
a) ***Logar* em uma conta administrativa:** Primeiramente deve *logar* na conta administrativa como está mostrado na Figura 32;

Figura 32 – Fazendo login em uma conta administrativa

Fonte: Produção do próprio autor.

b) **No Menu Navegação 'Ler o Código de Barras':** Ir ao menu de acesso e apertar em 'Ler o Código de Barras' como mostrado na Figura 33;

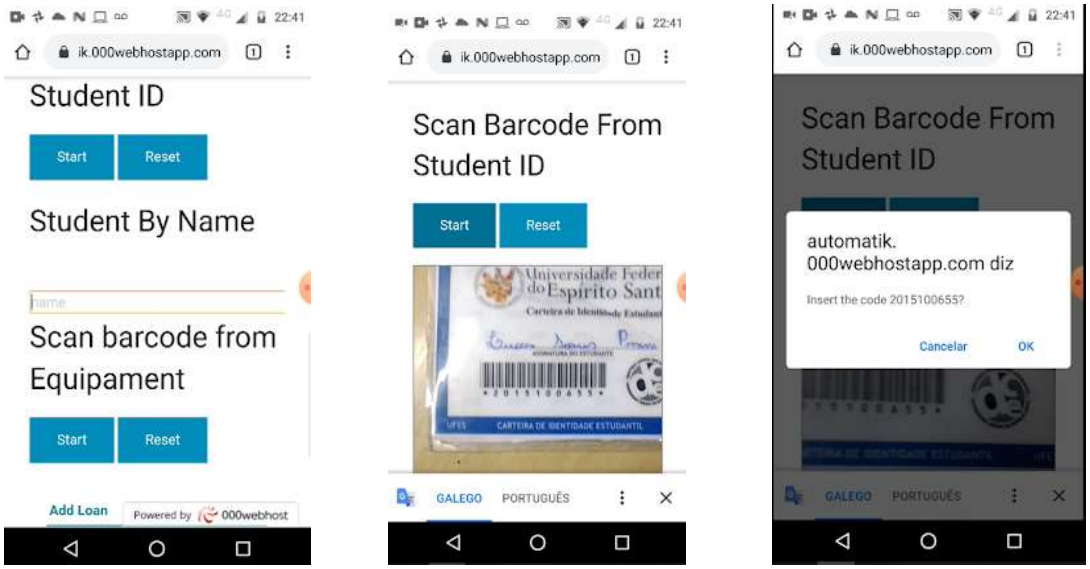
Figura 33 – ‘Ler Código de Barras’ no menu de navegação



Fonte: Produção do próprio autor.

c) **Lendo a Matrícula do Estudante:** Em ‘*Scan Barcode From Student ID*’ e selecione ‘*Start’* e posicione a câmera para capturar o código de barras da carteirinha do estudante. Quando é reconhecido o código de barras, uma janela *Pop-up* apresenta o código de barras lido e se for o correto selecione ‘*Ok*’. Esse procedimento é mostrado na Figura 34;

Figura 34 – Procedimentos para ler a matrícula do estudante através do código de barras: (a)Apertar no botão 'Start' no título '*Scan Barcode From Student ID*'; (b) Posicionar a câmera no código de barras do aluno; e (c) Apertar '*Ok*' na janela de *Pop-up* que aparece caso o código for correto.

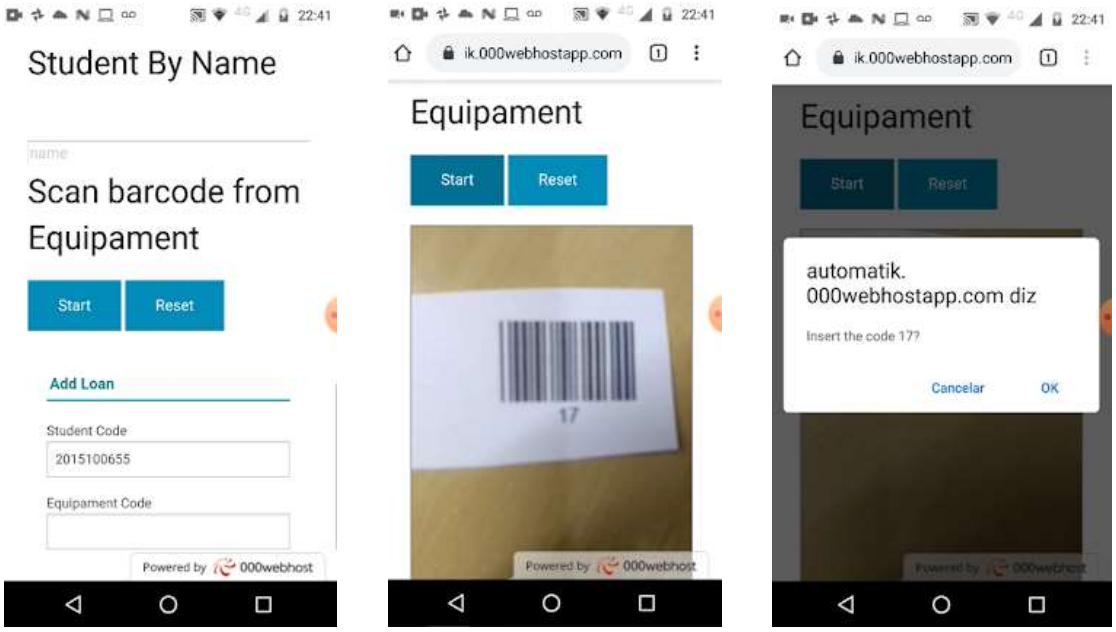


(a) (b) (c)

Fonte: Produção do próprio autor.

d) **Lendo o Código do Equipamento ou *Kit*:** No título '*Scan Barcode From Equipament*' e selecione '*Start*' e posicione a câmera para capturar o código de barras do equipamento ou *kit*. Quando é reconhecido o código de barras, uma janela '*Pop-up*' apresenta o código de barras lido e se for o correto você confirma. Esse procedimento é mostrado na Figura 35;

Figura 35 – Procedimentos para ler a matrícula do estudante através do código de barras: (a)Aperta no Botão 'Start' no título '*Scan Barcode From Equipament*'; (b) Posicionar a câmera no código de barras do equipamento ou *kit*; e (c) Apertar 'Ok' na janela de Pop-up que apareceu caso for correto

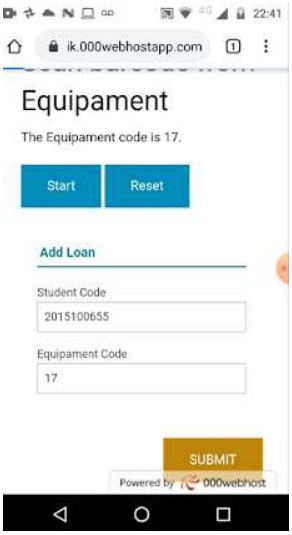


(a) (b) (c)

Fonte: Produção do próprio autor.

e) **Confirmar o Empréstimo:** Selecionar '*Submit*' para confirmar o empréstimo como mostrado na Figura 36.

Figura 36 – Confirmar o empréstimo

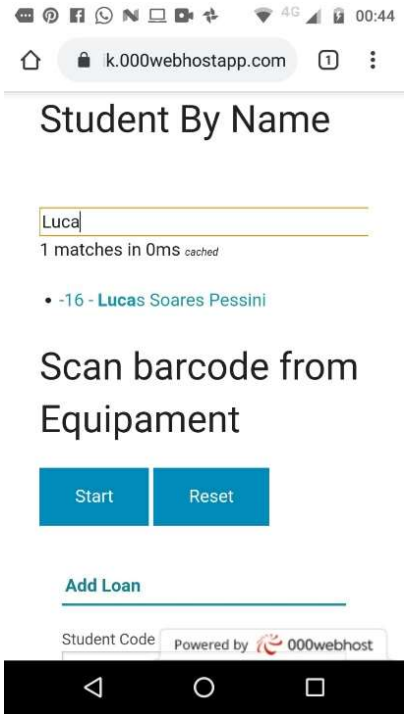


Fonte: Produção do próprio autor.

Depois do passo ‘e’, o sistema volta para página para registrar o próximo empréstimo, tornando o procedimento de vinculação serial e diminui o tempo para fazer uma nova vinculação.

Caso o aluno não tenha carteira de estudante ou caso a leitura do código de barras não seja possível, pode pesquisar se o nome do estudante está no banco de dados através da caixa de texto como mostrado na Figura 37.

Figura 37 – Pesquisando o nome do aluno no banco de dados para fazer o empréstimo



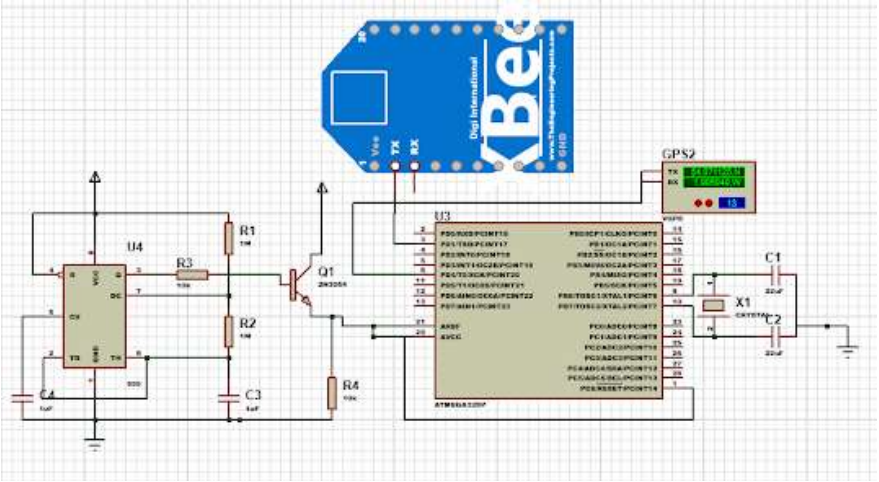
Fonte: Produção do próprio autor.

Para o procedimento de devolução, na página inicial, logado em uma conta administrativa, ir à lista de equipamentos e *kits* emprestados no momento como mostrado na Figura 15. Selecionar ‘*Devolution*’ no empréstimo referente ao equipamento que está sendo entregue pelo aluno, e inserir em *Real Devolution* a data e a hora que foi entregue o equipamento.

5.3 Simulação do Módulo RTLS

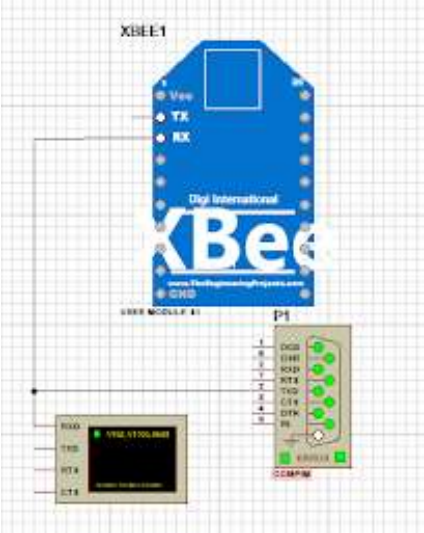
Será abordada neste item a simulação do módulo RTLS que permite inserir no banco de dados a localização do equipamento. Foi utilizado o programa Proteus para montar os circuitos que são mostrados nas Figuras 38 e 39.

Figura 38 – Circuito que estará situado nos equipamentos com o módulo GPS



Fonte: Produção do próprio autor.

Figura 39 – Circuito que ficará ligado a um computador recebendo os dados de localização e enviando para o banco de dados



Fonte: Produção do próprio autor.

Na Figura 38 pode-se perceber que não foi utilizado Arduino. Ao invés disso, utilizou-se o Atmega328p, que é o mesmo microcontrolador utilizado no Arduino Uno. Tal adequação visa redução do consumo e do tamanho do módulo.

O CI 555 em modo astável permite energizar o microcontrolador com a frequência desejada, reduzindo o consumo de energia.

No Atmega328p foi utilizado o mesmo código para uma plataforma Arduino Uno, o que permitiu sua implementação na Arduino IDE e o uso de suas bibliotecas.
Como observado na Figura 38, o Atmega328p é conectado ao módulo GPS para receber dados de satélites. O módulo GPS usado neste projeto é o NEO-6M que permite medir a posição (latitude, longitude e altitude), hora, data e velocidade.

O módulo GPS NEO-6M utilizado é semelhante ao mostrado na Figura 40.

Figura 40 – Sensor GPS utilizado

Fonte: Simple Projects (2016).

Geralmente o módulo GPS possui 4 pinos: VCC, RX, TX e GND. Ele usa comunicação serial (protocolo UART) para se comunicar com o microcontrolador onde os pinos RX / TX são para receber / transmitir dados de / para o microcontrolador (SIMPLE PROJECTS, 2016).

Observe que o pino RX do módulo GPS não está conectado porque não há necessidade de enviar dados do Atmega328p para ele. Neste projeto, o Atmega328p recebe apenas dados do módulo GPS (o módulo GPS recebe dados apenas dos satélites).

O código desenvolvido no Arduino IDE e que foi utilizado no Atmega328p está presente no apêndice A. Os dados do módulo GPS NEO-6M obtidos são: latitude, longitude, altitude, hora,

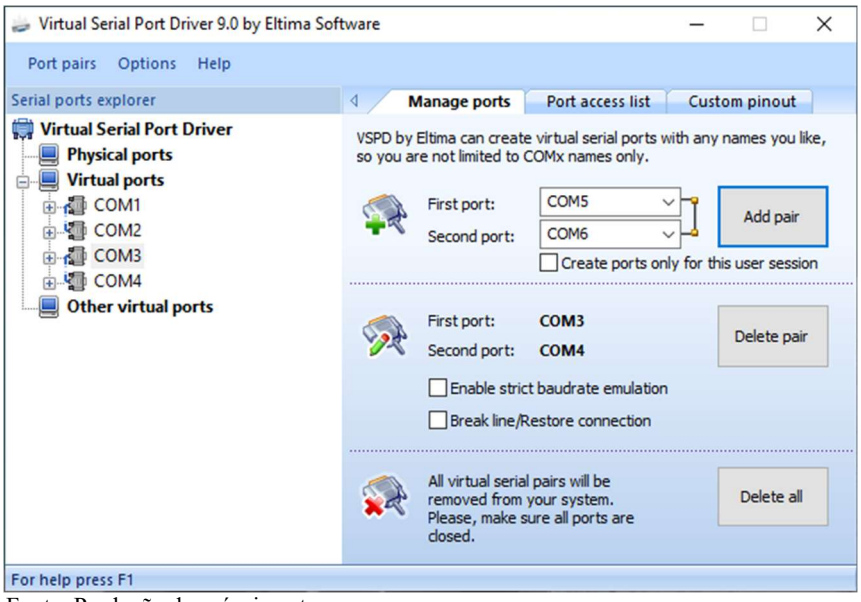
data e número de satélites em uso. Esses dados são enviados para o XBee por meio da porta serial.

Para simular o módulo XBee no Proteus foi necessário utilizar uma biblioteca que permite a interação do XBee com o microcontrolador.

É preciso alterar as propriedades dos módulos XBee no Proteus para que os dois se comuniquem, pois, cada módulo será ligado a uma porta física do computador. O módulo XBee da Figura 38 está configurada ao COM3 enquanto o da Figura 39 está configurado ao COM4.

Para realizar a comunicação entre os dois XBee, é necessário combinar virtualmente essas duas portas e, para isso, usou-se um *software* chamado *Driver de Software Virtual* da *Eltima*. Na Figura 41 pode-se observar essa combinação virtual feita pelo programa.

Figura 41 – Conectando COM3 e COM4 virtualmente com *Virtual Serial Port Driver*



Fonte: Produção do próprio autor.

Executando a simulação, o XBee do circuito da Figura 39 começará a receber dados de localização do GPS. Esses dados são enviados por comunicação serial para porta COMPIM, que é uma porta serial do computador.

O COMPIM modela uma porta serial física. Ele armazena em *buffer* a comunicação serial e a apresenta como sinais digitais para o circuito. Quaisquer dados seriais transmitidos a partir do modelo UART ou da CPU também passarão pela porta serial do computador (ELTIMA, 2017).

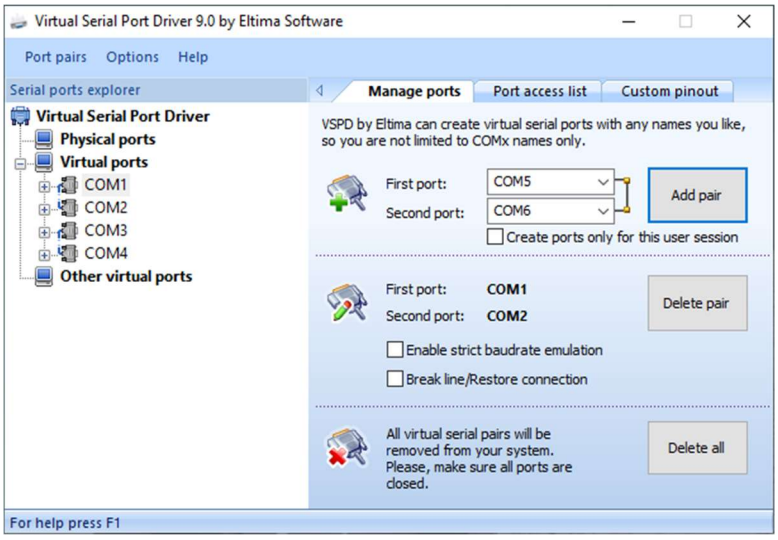
Esta simulação utiliza o protocolo UART. Após o XBee receber os dados, ele os envia para o computador. Uma maneira eficaz de fazer isso em circuito real é usar um conversor UART-COM, como o chip MAX232, para conectar o computador ao XBee.

O Proteus não cria portas seriais virtuais com as quais pode-se simular a comunicação com portas físicas. Assim, foi necessário empregar *software* adicional para criar portas seriais virtuais no Proteus.

A resolução desse problema baseou-se no *Virtual Serial Port Driver*.

Foi criado um par de portas nomeadas COM1 e COM2 como mostrado na Figura 42.

Figura 42 – Conectando COM1 e COM2 virtualmente com *Virtual Serial Port Driver*



Fonte: Produção do próprio autor.

Resumindo, conectou-se COMPIM ao COM2 através das configurações do Proteus e o COM2 foi conectado ao COM1 através do *Virtual Serial Port Driver*.

Quando foi simulado o circuito dado pelas Figuras 38 e 39, também foi executado o *script* presente no apêndice B. Esse *script* em *PowerShell* praticamente lê a porta COM1 e envia o que foi lido para o banco de dados através do método *Get* do protocolo HTTP.

Simulando o projeto do Proteus e executando o código no *prompt* de comando conseguiu-se a seguinte resposta mostrada na Figura 43.

Figura 43 – Interface do *Prompt* quando executa o *script* enquanto a simula o circuito

```
Prompt de Comando - powershell
Satellites = 10
Latitude   = 54.071125
Longitude  = -1.995948
Altitude   = 100.00 meters
Speed      = 0.00 kmph
Time (GMT) : 00:39:11
Date       : 26/11/2019
Satellites = 10
Latitude   = 54.071125
Longitude  = -1.995948
Altitude   = 100.00 meters
Speed      = 0.00 kmph
Time (GMT) : 00:39:11
Date       : 26/11/2019
Satellites = 10
Latitude   = 54.071125
Longitude  = -1.995948
Altitude   = 100.00 meters
Speed      = 0.00 kmph
Time (GMT) : 00:39:12
Date       : 26/11/2019
Satellites = 10
Latitude   = 54.071125
Longitude  = -1.995948
Altitude   = 100.00 meters
Speed      = 0.00 kmph
Time (GMT) : 00:39:13
Date       : 26/11/2019
Satellites = 10
```

Fonte: Produção do próprio autor.

A Figura 44 mostra os dados do Módulo RTLS inseridos no banco de dados. Para acessar essa página, só é preciso acessar o seguinte *link*: https://automatik.000webhostapp.com/AutomaTIK/localizations.

Figura 44 – Dados inseridos no banco

Localizations | Logout | admin | Projeto AutomaTIK

ACTIONS

New Localization

Localizations

Id	Text	Actions
21	Speed Invalid	View Edit Delete
22	Time (GMT) : 03:13:36	View Edit Delete
23	Date : 02/12/2019	View Edit Delete
24	Satellites = 13	View Edit Delete
25	Location Invalid	View Edit Delete
26	Altitude Invalid	View Edit Delete
27	Speed Invalid	View Edit Delete
28	Time (GMT) : 03:13:39	View Edit Delete
29	Date : 02/12/2019	View Edit Delete
30	Satellites Invalid	View Edit Delete
31	Location Invalid	View Edit Delete
32	Altitude Invalid	View Edit Delete
33	Speed Invalid	View Edit Delete

Powered by 000webhost

Fonte: Produção do próprio autor.

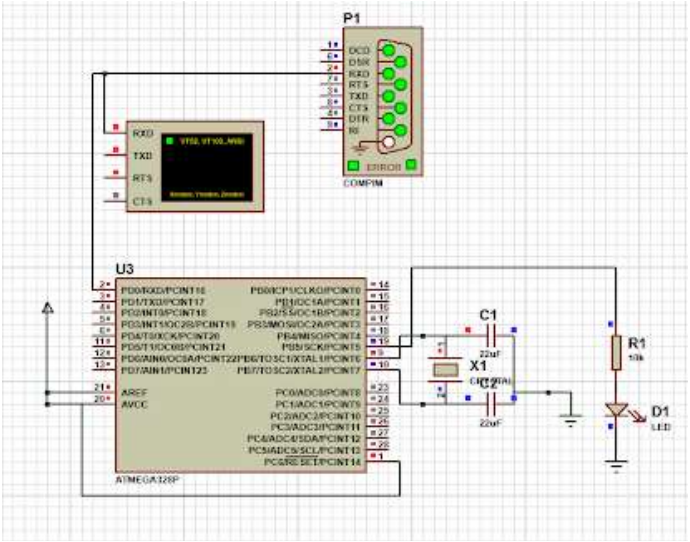
5.4 Módulo de ‘Autoatendimento’

A ideia desse módulo é permitir que o próprio usuário, desde que autorizado, retire ou entregue o *kit*/equipamento sem intermédio de uma outra pessoa.

O procedimento é o seguinte: o usuário deve realizar um agendamento prévio e quando for buscar o equipamento, precisa acessar o aplicativo *web* e selecionar o seu empréstimo, o que aciona o sistema de abertura do armário, neste momento, representado pelo LED aceso.

A Figura 45 mostra o circuito que foi simulado no Proteus que representa o módulo de ‘autoatendimento’.

Figura 45 – Esquemático do circuito eletrônico do autoatendimento



Fonte: Produção do próprio autor.

O circuito da Figura 45 apresenta o microcontrolador Atmega328p que substitui a plataforma Arduino Uno. O programa que está em execução no microcontrolador foi desenvolvido na IDE do Arduino e está disponível no apêndice D.

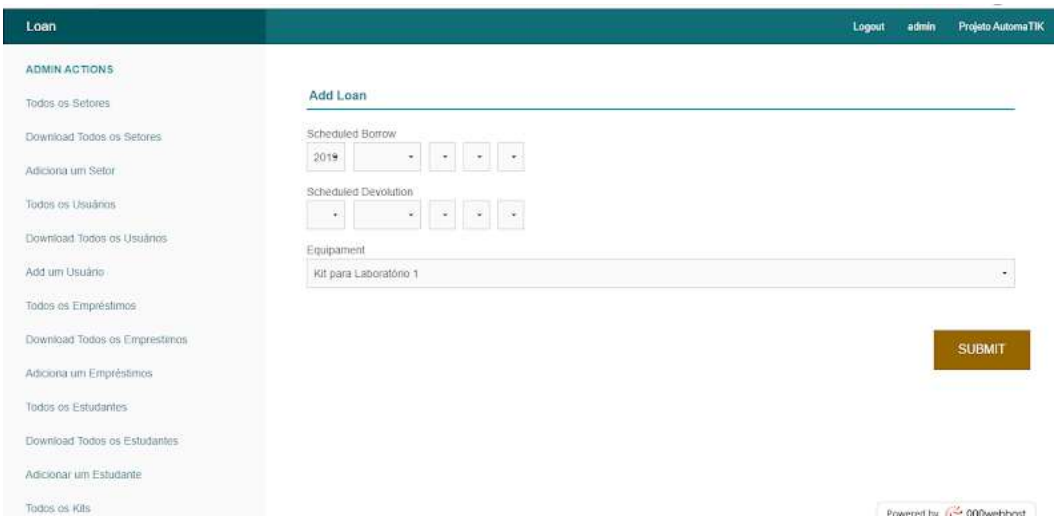
O controlador recebe informações do computador por meio de comunicação serial da porta COMPIM, que é uma porta serial do computador, que já foi mencionada anteriormente.

Dependendo do caractere que o controlador recebe ele vai acender ou apagar o LED, representando assim um armário abrindo ou fechando.

O que controla os caracteres que são transmitidos pela porta serial do computador é o *script* que está no apêndice C. Ele é um *script* que está em um arquivo '.bat'. Quando esse arquivo é executado, ele fica em *loop* infinito, onde acessa o banco de dados, e dependendo do seu estado, ele envia um caractere diferente na porta serial.

Para testar esse módulo de autoatendimento é necessário acessar a página inicial do aplicativo *web* e selecionar 'Novo Empréstimo' no menu Lateral, que solicitará o *login* de usuário e vai ser direcionado para a página representada na Figura 46.

Figura 46 – Página para adicionar um novo empréstimo para ser utilizado no 'autoatendimento'



Fonte: Produção do próprio autor.

Na página mostrada pela Figura 46 o usuário insere as informações de seu próximo empréstimo, ou seja, é realizado um agendamento do empréstimo de seu equipamento, informando a data e a hora que será retirado e devolvido o equipamento bem como a sua identificação.

De volta à página inicial, será apresentado algo parecido com a Figura 47.

Figura 47 – Página inicial com próximos empréstimos daqui 2 horas

Fonte: Produção do próprio autor.

Selecionando *Get* no empréstimo que foi agendado na tabela de 'Próximos Empréstimos Daqui 2 Horas'. Assim terá que *logar* novamente e será direcionado a uma página mostrada pela Figura 48, informando que 'Você pode pegar o seu equipamento. Você tem 5 minutos para pegar'.

Figura 48 – Página direcionada logo depois que é apertado em '*Get*'

Fonte: Produção do próprio autor.

Depois de ter aparecido a página da Figura 48, o LED do circuito da Figura 45 ficará aceso por 5 minutos.

6 CONCLUSÃO

O empréstimo de equipamentos é uma necessidade no meio acadêmico, seja para aulas práticas ou para o desenvolvimento de projetos fora do horário de aula. O controle dos empréstimos atualmente é feito de forma manual, que por vezes apresenta algumas falhas durante o processo, além de não manter um histórico de empréstimos.

Para o desenvolvimento do sistema foram realizadas pesquisas bibliográficas para definição da estrutura de *hardware* e *software*. Durante a pesquisa foram definidos os elementos que seriam utilizados para atender de maneira mais eficaz aos objetivos do projeto.

O sistema desenvolvido facilita o controle de saídas e entradas dos equipamentos e *kits* no Laboratório. Agiliza o atendimento aos alunos e melhora o controle das movimentações de empréstimo. Além disso, traz facilidades ao acesso das informações importantes para o auxílio na administração do laboratório. O *software* fornece relatórios gerenciais que são de extrema importância para uma eventual tomada de decisão, assim garantindo a qualidade de serviço prestado.

Para um trabalho futuro, tem-se as seguintes observações:

- O *host* utilizado, embora gratuito, permite poucos acessos ao banco de dados por hora, limitando a aplicação real do sistema;
- O *host* gratuito é lento;
- O almoxarifado possui características de uma gaiola de Faraday, o que pode dificultar alguns acessos à rede sem fio;
- Não foi desenvolvida uma interface com um mapa para indicar a localização dos equipamentos para o Módulo RTLS como o Sistema de Informações Geográficas, ou GIS; e
- Ainda não foi feito um método direto para registrar os alunos sem a carteirinha.

REFERÊNCIAS BIBLIOGRÁFICAS

ACADEMIA DO CÓDIGO. **O que é Programação *web* e Programação Desktop?** .2017. Disponível em: http://blog.academiadocodigo.com.br/2015/04/o-que-e-programacao-web-e-programacao-desktop/. Acesso em: 29 maio 2019.

BRAINVIRE. **Why CakePHP is a Worldwide Popular web Application Development Platform?**. 2018. Disponível em: https://www.brainvire.com/why-cakephp-is-a-worldwide-popular-web-application-development-platform/. Acesso em: 29 nov. 2019.

CAKEPHP. **New CakePHP 3.7 Red Velvet**. 2017. Disponível em: https://cakephp.org/. Acesso em: 29 maio 2019.

CITISYSTEMS. **O que é Automação Industrial?** 2016. Disponível em: https://www.citisystems.com.br/o-que-e-automacao-industrial/. Acesso em: 29 maio 2019.

COMPUTADOR PORTUGUÊS. **A diferença entre um sistema embarcado e um computador**. 2015. Disponível em: http://ptcomputador.com/Ferragens/computer-drives-storage/51420.html. Acesso em: 29 maio 2019.

DEVMEDIA. **O que é Banco de Dados?**.2014. Disponível em: https://www.devmedia.com.br/conceitos-fundamentais-de-banco-de-dados/1649. Acesso em: 29 maio 2019.

DIGI INTERNATIONAL. **How XBee devices work** . 2019. Disponível em: https://www.virtual-serial-port.org/article/com-port-in-proteus/. Acesso em: 29 nov. 2019.

ELECTRONIC IMAGING. **Types of Barcodes**. 2016. Disponível em: https://barcode-labels.com/getting-started/barcodes/types/. Acesso em: 29 nov. 2019.

ELTIMA. **Working with the Serial Port model in Proteus**. 2017. Disponível em: https://www.virtual-serial-port.org/article/com-port-in-proteus/. Acesso em: 29 nov. 2019.

EMBARCADOS. **Apresentando o módulo ESP8266**. 2015. Disponível em: https://www.embarcados.com.br/modulo-esp8266/. Acesso em: 29 maio 2019.

FERNANDES, B. F. **Sistema de Biblioteca USP**. 2013. Disponível em: http://biblioteca.fflch.usp.br/sites/biblioteca.fflch.usp.br/files/REGRAS_EMPR%C3%89STIMOS_.pdf. Acesso em: 29 maio 2019.

FILEHIPPO. **Virtual Serial Port Driver**. 2018. Disponível em: https://filehippo.com/download_virtual_serial_port_driver/. Acesso em: 29 nov. 2019.

GALVANIZE. **How to Use Chrome DevTools for the Absolute Beginner**. 2017. Disponível em: https://blog.galvanize.com/how-to-use-chrome-devtools-for-the-absolute-beginner/. Acesso em: 29 nov. 2019.

GESTÃOCLICK. **O que é automação de processos, quando e como iniciar**. 2012. Disponível em: https://gestaoclick.com.br/blog/automacao-de-processos. Acesso em: 29 maio 2019.

IKEDA, R. F. T. B. **Sistema de Gerenciamento de Empréstimos dos Equipamentos de Laboratório do DAELN-CM**. 2017. Trabalho de Conclusão de Curso (Bacharelado em Engenharia Eletrônica) - Universidade Tecnológica Federal do Paraná. Campo Mourão, 2017. Disponível em: http://repositorio.roca.utfpr.edu.br/jspui/handle/1/7256. Acesso em: 29 maio 2019.

KINSTA. **What is Github?**. 2016. Disponível em: https://kinsta.com/knowledgebase/what-is-github/. Acesso em: 29 nov. 2019.

LEND-ITEMS. **Manage Your Inventory Easily**. 2015. Disponível em: http://www.lend-items.com/. Acesso em: 29 maio 2019.

MACÊDO, D. **Entendendo as aplicações web**. 2017. Disponível em: https: //www.diegomacedo.com.br/entendendo-as-aplicacoes-web/. Acesso em: 29 maio 2019.

MAQUINA DE RESULTADOS. **O que é o nome de domínio e como ele funciona?**. 2016. Disponível em: https://www.maquinaderesultados.com.br/o-que-e-o-nome-de-dominio-e-como-ele-funciona/. Acesso em: 29 nov. 2019.

MARCEL, W. **VaiVem - Sistema de empréstimos**. 2013. Disponível em: http://wille.blog.br/2012/02/vaivem-sistema-de-emprestimos. Acesso em: 29 maio 2019.

MARTINS, G. M. **Sistema de Gerenciamento para Biblioteca**. 2015. Disponível em: https://cepein.femanet.com.br/BDigital/arqTccs/1311320122.pdf. Acesso em: 29 maio 2019.

MERCADO ELETRÔNICO. **Automatização de processos: O que é e quais são os benefícios?** 2016. Disponível em: https://blog.me.com.br/automatizacao-de-processos/. Acesso em: 29 maio 2019.

NOVIDÁ. **Rede LoRa: o que é e quais são as aplicações?**. 2018. Disponível em: https://novida.com.br/blog/rede-lora/. Acesso em: 29 maio 2019.

OFICINA DA NET. **O que são sistemas embarcados?**. 2017. Disponível em: https: //www.oficinadanet.com.br/post/13538-o-que-sao-sistemas-embarcados. Acesso em: 29 maio 2019.

QUORA. **What is .bat files?**. 2010. Disponível em: https://www.quora.com/What-are-bat-files. Acesso em: 29 nov. 2019.

_____. **What is JavaScript used for?**. 2018. Disponível em: https://www.quora.com/What-is-JavaScript-used-for-1. Acesso em: 29 nov. 2019.

_____. **What is proteus software?**. 2015. Disponível em: https://www.quora.com/What-is-proteus-software. Acesso em: 29 nov. 2019.

_____. **What is the use of XAMPP?**. 2016. Disponível em: https://www.quora.com/What-is-the-use-of-XAMPP. Acesso em: 29 nov. 2019.

_____. **Why CakePHP is popular for web App development?**. 2017. Disponível em: https://www.quora.com/Why-CakePHP-is-popular-for-web-App-development-1. Acesso em: 29 nov. 2019.

SIMPLE PROJECTS. **Interfacing Arduino with NEO-6M GPS module?**. 2016. Disponível em: https://simple-circuit.com/arduino-neo-6m-gps-module/. Acesso em: 29 nov. 2019.

SPAKOCAM. **SparkoCam Virtual webcam**. 2015. Disponível em: https://sparkosoft.com/sparkocam. Acesso em: 29 nov. 2019.

SPARKFUN. **What is an Arduino?**. 2015. Disponível em: https://learn.sparkfun.com/tutorials/what-is-an-arduino/all. Acesso em: 29 nov. 2019.

STUDY.COM. **What is PHP Used For? - Uses & Advantages**. 2015. Disponível em: https://study.com/academy/lesson/what-is-php-used-for-uses-advantages.html. Acesso em: 29 nov. 2019.

TACHTARGET. **FileZilla**. 2017. Disponível em: https://whatis.techtarget.com/definition/FileZilla. Acesso em: 29 nov. 2019.

TECHOPEDIA. **What is Powershell?**. 2017. Disponível em: https://www.techopedia.com/definition/25975/powershell. Acesso em: 29 nov. 2019.

TECHTUDO. **Droidcam Pc Client**. 2014. Disponível em: https://www.techtudo.com.br/tudo-sobre/droidcam-pc-client.html. Acesso em: 29 nov. 2019.

_____. **O que é XAMPP e para que serve**. 2018. Disponível em: https://www. techtudo.com.br/dicas-e-tutoriais/noticia/2012/02/o-que-e-xampp-e-para-que-serve.html. Acesso em: 29 maio 2019.

WIKIPÉDIA. **Framework**. 2013. Disponível em: https://pt.wikipedia.org/wiki/Framework. Acesso em: 29 maio 2019.

YOUR HTML SOURCE. **What is HTML?**. 2014. Disponível em: https://www.yourhtmlsource.com/starthere/whatishtml.html. Acesso em: 29 nov. 2019.

ZXING. **Library**. 2017. Disponível em: https://github.com/zxing-js/library. Acesso em: 29 nov. 2019.

APÊNDICE A – CÓDIGO FEITO NA IDE DO ARDUINO PARA O MÓDULO RTLS

```
// Interfacing Arduino with NEO-6M GPS module

#include <TinyGPS++.h>          // Include TinyGPS++ library
#include <SoftwareSerial.h>     // Include software serial library

TinyGPSPlus gps;

#define S_RX    4               // Define software serial RX pin
#define S_TX    3               // Define software serial TX pin

SoftwareSerial SoftSerial(S_RX, S_TX);    // Configure SoftSerial library

void setup(void) {
  Serial.begin(9600);
  SoftSerial.begin(9600);
}

void loop() {

  while (SoftSerial.available() > 0) {

    if (gps.encode(SoftSerial.read())) {

      if (gps.location.isValid()) {
        Serial.print("Latitude   = ");
        Serial.println(gps.location.lat(), 6);
        Serial.print("Longitude  = ");
        Serial.println(gps.location.lng(), 6);
      }
      else
```

```
  Serial.println("Location Invalid");

if (gps.altitude.isValid()) {
  Serial.print("Altitude   = ");
  Serial.print(gps.altitude.meters());
  Serial.println(" meters");
}
else
  Serial.println("Altitude Invalid");

if (gps.speed.isValid()) {
  Serial.print("Speed      = ");
  Serial.print(gps.speed.kmph());
  Serial.println(" kmph");
}
else
  Serial.println("Speed Invalid");

if (gps.time.isValid()) {
  Serial.print("Time (GMT) : ");
  if(gps.time.hour() < 10)     Serial.print("0");
  Serial.print(gps.time.hour());
  Serial.print(":");
  if(gps.time.minute() < 10)   Serial.print("0");
  Serial.print(gps.time.minute());
  Serial.print(":");
  if(gps.time.second() < 10)   Serial.print("0");
  Serial.println(gps.time.second());
}
else
  Serial.println("Time Invalid");

if (gps.date.isValid()) {
```

```
      Serial.print("Date       : ");
      if(gps.date.day() < 10)      Serial.print("0");
      Serial.print(gps.date.day());
      Serial.print("/");
      if(gps.date.month() < 10)    Serial.print("0");
      Serial.print(gps.date.month());
      Serial.print("/");
      Serial.println(gps.date.year());
    }
    else
      Serial.println("Date Invalid");

    if (gps.satellites.isValid()) {
      Serial.print("Satellites = ");
      Serial.println(gps.satellites.value());
    }
    else
      Serial.println("Satellites Invalid");

  }

 }

}
```

APÊNDICE B – CÓDIGO EXECUTADO NO POWERSHELL PARA ENVIAR DADOS PARA O BANCO DE DADOS

```
powershell
$COM = [System.IO.Ports.SerialPort]::getportnames()

function read-com {
   $port= new-Object System.IO.Ports.SerialPort COM1,9600,None,8,one

   $port.Open()
   do {
      $line = $port.ReadLine()
      Write-Host $line # Do stuff here
      $Parameters = @{text = $line}
                  Invoke-webRequest -Uri
'http://automatik.000webhostapp.com/AutomaTIK/localizations/add' -Body $Parameters -
Method Get
   }
   while ($port.IsOpen)
}
read-com
```

APÊNDICE C – CÓDIGO DO ARQUIVO ‘.BAT’ PARA O MÓDULO DE ‘AUTOATENDIMENTO’

```
ECHO .
Set /p NUM="Qual e o numero do COM?(SOMENTE O NUMERO):"

:loop

cd C:\xampp\htdocs\AutomaTIK
curl https://automatik.000webhostapp.com/AutomaTIK/loan/file > script.bat

call script.bat

echo Executou com o COM%NUM%

goto loop
```

APÊNDICE D – CÓDIGO FEITO NA IDE DO ARDUINO PARA O MÓDULO DE 'AUTOATENDIMENTO'

```
int time(long timeHigh, long timeLow, long atraso, long mref = 0) {
  long ajuste = mref % (timeHigh + timeLow);
  long resto  = (millis() + timeHigh + timeLow - ajuste - atraso) % (timeHigh + timeLow);
  return (resto < timeHigh ? HIGH : LOW);
}



void setup(){
  Serial.begin(9600);
  pinMode(LED_BUILTIN, OUTPUT);
}

boolean pisca = false;

void loop(){
  int c = Serial.read();

  if (c == 97) { digitalWrite(LED_BUILTIN, HIGH); pisca = false; } //a -> liga
  if (c == 98) { digitalWrite(LED_BUILTIN, LOW);  pisca = false; } //b -> desliga
  if (c == 99) { pisca = true;                    } //c -> pisca

  if (pisca)   { digitalWrite(LED_BUILTIN, time(400, 400, 0));   }
}
```