

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO AMAZONAS

GERÊNCIA EDUCACIONAL DA ÁREA DE SERVIÇOS (GEAS)

IFCONTROL: Sistema de automação predial

Manaus-AM

2015

GUILHERME SOUZA DA SILVA

JANSEN SAUNIER DE ALCANTARA JÚNIOR

PEDRO ALAN TAPIA RAMOS

IFCONTROL: Sistema de automação predial

Projeto Final apresentado como requisito final para
a obtenção da conclusão do Curso Técnico Nível
Médio em Informática

Orientador: Prof. Dr. Jucimar Brito de Souza

Manaus-AM

2015

FICHA CATALOGRÁFICA:

SILVA, Guilherme Souza; JÚNIOR, Jansen
Sau de Alcântara; RAMOS, Pedro Alan Tapia.

IFCONTROL, 2015

Trabalho Científico – Instituto Federal de
Educação, Ciência e Tecnologia do Amazonas,
2015.

Orientador: Prof. Dr. Jucimar Brito de Souza

FOLHA DE APROVAÇÃO

Guilherme Souza da Silva

Jansen Saunier de Alcantara Júnior

Pedro Alan Tapia Ramos

IFCONTROL - Sistema de automação predial

Prof. Dr. Jucimar Brito de Souza (Orientador)

Prof. Msc Marcia da Costa Pimenta Martins

Prof. Msc Emmerson Santa Rita da Silva

Manaus, Dezembro/2015

Resumo

A energia elétrica é um dos itens que mais pesa no orçamento de uma instituição de ensino. Economizá-la é uma tarefa difícil, devido ao tamanho e diversidade de horários e utilização das salas de aula e seus ambientes. No Instituto Federal do Amazonas (IFAM) observa-se algumas situações de desperdício de energia como: luzes acesas nas salas de aula e laboratórios quando não tem alunos nas mesmas, aparelhos de ar-condicionado ligados fora do horário de aula e a necessidade de ter um funcionário para percorrer todo o instituto para acioná-los. Observado estas situações, propõe-se neste projeto o desenvolvimento de um equipamento microcontrolado para automatizar tarefas relacionadas a sala de aula remotamente e assim evitar o desperdício de energia.

LISTA DE FIGURAS

- Figura 1: Ciclo de vida da Activity
- Figura 2: Diagrama de caso de uso
- Figura 3: Diagrama de Classes do Servidor
- Figura 4: Diagrama de Classes do Modelo de Dados
- Figura 5: Diagrama de Classes da Aplicação Android
- Figura 6: Tela de Login Desktop
- Figura 7: Tela de Login Mobile.
- Figura 8: Tela de Cadastro Desktop.
- Figura 9: Tela de Cadastro Mobile.
- Figura 10: Tela Home Desktop.
- Figura 11: Tela Home Mobile.
- Figura 12: Tela de Sala.
- Figura 13: Tela Horário.
- Figura 12: Tela Logs.

Sumário

CAPÍTULO I

1.1 OBJETIVOS

1.1.1 Objetivo Geral

1.2.2 Objetivo Específicos

CAPÍTULO II

1. Arduino

2. Shield Ethernet

3. ASTAH Community

4. Linguagem Java

5. NetBeans IDE

6. Socket

7. JSON/GSON

8. JDBC

9. MySQL

10. JavaFX

11. Scene Builder

12. Programação Android

13. Android Studio (IDE)

CAPÍTULO III

1.1. Diagrama de Classes: Servidor

1.2. Diagrama de Classes: Modelo de Dados

1.3. Diagrama de Classes: Aplicação Android

2 TELAS DO PROTÓTIPO

2.1 Login

2.2 Telas de Cadastro

2.3 Tela Home

2.6 Tela logs

CAPÍTULO IV

Conclusões

Referências Bibliográficas

CAPÍTULO I

INTRODUÇÃO

No Instituto Federal, observam-se algumas situações que podem ser automatizadas e com isso gerar economia e diminuir o trabalho de determinados servidores. São elas: luzes acesas nas salas de aula e laboratórios quando não tem alunos nas mesmas, aparelhos de ar-condicionado ligados fora do horário de aula e a necessidade de um servidor ir de sala em sala no início do turno e ligar todos os aparelhos de ar, sendo que este precisa fazer a mesma atividade a cada final e início de outro turno.

Observado estas situações, propõe-se neste projeto o desenvolvimento de um equipamento microcontrolado para automatizar tarefas sem a necessidade de intervenção humana.

1.1 OBJETIVOS

1.1.1 Objetivo Geral

Desenvolver um protótipo microcontrolado com o Arduino, além de uma aplicação móvel e desktop, para monitorar e controlar efetivamente luzes e aparelhos de ar-condicionado das salas de aula e laboratórios do IFAM- Campus Manaus-Centro

1.2.2 Objetivos Específicos

2.1.1 Montar um equipamento microcontrolado com o Arduino para gerenciar:

- a. A presença na sala, por meio do sensor de presença acoplado no Arduino
- b. As lâmpadas das salas de aula

- c. Os aparelhos de ar-condicionado da sala. Tanto sobre o seu funcionamento (se ligado ou desligado), quanto como o controle da temperatura.
 - d. A informação de temperatura e umidade da sala.
 - e. O acionamento automático dos aparelhos de ar-condicionado e lâmpadas em um horário específico determinado pelo usuário
- 2.1.2** Criar um modelo simulado de um controle de várias salas do Instituto.
- 2.1.3** Projetar e implementar uma aplicação mobile e desktop que gerencie e monitore as ações que o Arduino irá executar nas salas de aula.

O restante deste trabalho está organizado da seguinte forma: no capítulo 2 são apresentados os conceitos básicos para melhor entendimento dos recursos e tecnologia aqui utilizados para o desenvolvimento desse projeto. O capítulo 3 apresenta o protótipo do trabalho bem como os diagramas e recursos usados para o seu desenvolvimento. E, no capítulo 4 as conclusões e trabalhos futuros.

CAPÍTULO II

CONCEITOS BÁSICOS E TECNOLOGIAS UTILIZADAS

1. Arduino

Arduino¹ é uma plataforma de prototipagem eletrônica, projetada com um microcontrolador Atmel AVR, desenvolvida por Massimo Banzi, David Cuartielles, Tom Igoe e David Mellis em 2005 com o objetivo de facilitar a criação de projetos para incentivar o ensino de eletrônica e programação. (Arduino, 2015)

Placas Arduino são capazes de receber informações através de sensores, como de temperatura, luminosidade, presença, umidade, assim como de diversos dispositivos de entrada, como botões, e enviá-lo para dispositivos de saída, como relés, motores, leds, emissores de sinais, telas LCD, dentre outros.

Para o controle do microcontrolador, é enviado um conjunto de instruções em uma linguagem de programação específica do Arduino (baseado no Wiring, essencialmente C/C++), e do Software Arduino (IDE), com base em Processing. (Massimo Banzi, 2015)

Possui hardware e software livres, desse modo é possível acessar o código fonte do software, assim como dos circuitos de hardware para estudo, aperfeiçoamento e adaptação para outros fins. (Michael McRoberts, 2015)

Artistas, designers, amadores, hackers novatos, e diversas pessoas interessadas em controlar objetos ou ambientes interativos usufruem da tecnologia pela facilidade de manuseio. Sua flexibilidade em relação a quantidade de dispositivos que podem ser

¹ <https://www.arduino.cc/>

controlados combinada com o software livre e o baixo custo do hardware possibilitam a contribuição de uma grande comunidade de usuários com código e instruções para diversos projetos baseados no Arduino (Simon Monk, 2015).

Para este projeto foi utilizado os seguintes componentes eletrônicos adicionais, além dos jumpers par interligação dos mesmos: sensor de temperatura e umidade, sensor de presença e movimento, emissor de infravermelho, lâmpada fluorescente assim como um módulo relé (um equipamento que contém transistores, conectores, leds, diodos e relés de forma compacta de modo que facilite a interligação do Arduino com um relé e demais componentes conectados a ele).

Para estender suas funções, como conexões Wireless e Ethernet, são utilizados shields - placas de circuito impresso com funções específicas acopladas à placa principal através de pinos. (Michael McRoberts, 2015)

2. [Shield Ethernet](#)

O Arduino Ethernet Shield baseia-se no chip WIZnet Ethernet W5100 o qual fornece acesso à rede (IP) nos protocolos TCP ou UDP possibilitando que uma placa Arduino se conecte n à uma rede Ethernet. Há um conector para o cabo RJ45 e um slot para um cartão micro-SD, permitindo o armazenamento de arquivos associados ao servidor. Possui bibliotecas que permitem a troca de informações com outra aplicação mediante o mecanismo de comunicação Socket (Arduino, 2015).

3. [ASTAH Community](#)

Astah², antigamente denominado JUDE (*Java and UML Developers Environment* (Ambiente para Desenvolvedores UML e Java) é um software para modelagem de sistemas UML (Unified Modeling Language – Linguagem de Programação Unificada), usada para representar um sistema de forma padronizada através de diagramas que possibilitam uma melhor visualização das comunicações entre as classes e objetos, assim como dos relacionamentos e da ordem das atividades (Teixeira de Carvalho Sbrocco e José Henrique, 2015).

4. [Linguagem Java](#)

Java³ é uma linguagem de programação interpretada orientada a objetos de alto nível com propósito geral desenvolvida pela Sun Microsystems, adquirida pela Oracle em 2009. Arquivos com código fonte Java são compilados em um formato chamado bytecode (com a extensão .class), que podem ser executados pela JVM (Java Virtual Machine – Máquina Virtual Java), tornando-o independente da plataforma (Wikipedia, a enciclopedia livre, 2015).

Dentre outras características que o tornam populares estão a vasta biblioteca com recursos de rede, sintaxe bastante semelhante a C/C++, facilidade na criação de aplicações multitarefas, extensa documentação das classes e API's disponíveis e uma grande comunidade para auxílio.

² <http://astah.net/>

³ https://www.java.com/pt_BR/

5. NetBeans IDE

Netbeans⁴ IDE é um ambiente de desenvolvimento integrado gratuito e com código aberto que disponibiliza inúmeros recursos para a criação de software nas linguagens Java, C/C++, PHP, dentre outros, disponível nas plataformas Windows, Linux, Solaris e MacOS. Oferece diversas ferramentas para aplicações Web (CSS, JSP, JSTL, EJBLs, Tomcat), aplicações Swing, linguagens de marcação (XML e HTML) assim como suporte a banco de dados e geração automática da documentação Java a partir dos comentários inseridos no código que facilitam e agilizam o desenvolvimento de aplicativos profissionais de desktop, empresariais, Web e móveis.

6. Socket

Socket é um mecanismo de comunicação entre duas aplicações através de uma rede de computadores, possibilitando a troca de informações entre um servidor e um cliente por meio do protocolo TCP e UDP. Cada socket é identificado pelo número da porta ao qual está conectado e seu endereço IP, dessa forma, uma máquina servidor aguarda requisições do cliente, que, conhecendo as informações acerca do mesmo, solicita uma conexão com o servidor. Ao estabelecerem um vínculo, o servidor direciona o cliente para outra porta, viabilizando o acesso para outras aplicações (Oracle, 2015)

A linguagem Java disponibiliza recursos para a interlocução entre aplicações por meio de Socket. Através destes, foi possível uma intercomunicação entre os clientes, aplicações Android/Desktop, e o Arduino com o servidor mediante um mesmo artifício.

⁴ <https://netbeans.org>

7. JSON/GSON

JSON⁵ (JavaScript Object Notation – Objeto de Notação JavaScript) é um formato de texto legível utilizado para transmissão de dados entre servidores e aplicações, baseado em notações de JavaScript, sendo uma alternativa ao XML. É independente da linguagem e usufrui convenções semelhantes a C/C++ e Java, tornando-o um intercâmbio de dados ideal.

Gson é uma biblioteca aberta para Java, criada pelo Google, que permite a conversão de objetos e classes para o formato Json, provendo mecanismos simples para decodificar e codificar. Foi utilizada para o transporte dos dados através dos diversos dispositivos Android e Desktop para o Servidor, e vice-versa, padronizando assim a transmissão das informações (Ajduke, 2015)

8. JDBC

JDBC (Java Database Connectivity) é uma API (Application Programming Interface – Interface de Programação de Aplicações), isto é, provê um conjunto de classes e interfaces que possibilitam a comunicação com qualquer banco de dados relacional que se baseie na estrutura SQL. Uma das suas vantagens é a possibilidade de reutilização do código para diversas aplicações que requerem acesso aos dados. A biblioteca JDBC inclui métodos para cada uma das tarefas comumente associadas ao bancos de dados, tais como criação da conexão, realização de consultas, visualização e modificação dos dados, todos através de comandos SQL. (Oracle, 2015)

⁵ <http://www.json.org/>

9. MySQL

O MySQL⁶ é um sistema de gerenciamento de banco de dados (SGBD), que utiliza a linguagem SQL (Linguagem de Consulta Estruturada, do inglês *Structured Query Language*) como interface. A portabilidade, compatibilidade com diversas linguagens, estabilidade e interfaces gráficas da MySql Inc. fácil manuseio, além da fácil integração com PHP, o tornam um dos mais populares e confiáveis gerenciadores de banco de dados existentes (Webopedia, 2015).

10. JavaFX

JavaFX é um conjunto de pacotes gráficos e multimídia os quais permitem projetar, testar, criar, depurar e implantar aplicações RIA (Rich Internet Application – Aplicações de Internet Rica) - aplicações Web que possuem características e funcionalidades de softwares tradicionais de Desktop – consistentes em diversas plataformas. A aparência e efeitos do JavaFX podem ser personalizados separadamente através de CSS (Cascading Style Sheets), dessa forma, a interface pode ser criada por designers de forma independente. (Oracle, 2015)

11. Scene Builder

JavaFX Scene Builder (Scene Builder) é uma ferramenta que permite a criação de interfaces de aplicações JavaFX sem codificação, apenas arrastando um componente para uma área de exibição de conteúdo. Além de possibilitar a alteração de propriedades e aplicação de folhas de estilo (CSS) ao projeto, o código FXML para o layout e a classe responsável pelo controle da tela são gerados automaticamente. (Oracle, 2015)

⁶ <https://www.mysql.com/>

12. Programação Android

Android é um sistema operacional baseado no núcleo linux e atualmente desenvolvido pela empresa de tecnologia Google. Com uma interface de usuário baseada na manipulação direta, o Android é projetado principalmente para dispositivos móveis com tela sensível ao toque como smartphones e tablets; com interface específica para TV (Android TV), carro (Android Auto) e relógio de pulso (Android Wear).

Na Aplicação mobile Android do sistema, foram utilizadas “*Activities*”. Estas *Activities* são, na verdade, pequenas “atividades” executada uma de cada vez. A maneira mais fácil de entendê-las é associar cada tela a uma *Activity*. Para controlá-las, já que apenas uma *Activity* pode ser executada de cada vez, existe a *Activity Stack*, ou pilha de Activity controlada pelo gerenciador de memórias do Sistema Operacional. Dentro das próprias Activities, o controle de reconhecimento da fase em que elas se apresentam no aparelho é descrito pelo Ciclo de vida da Activity.

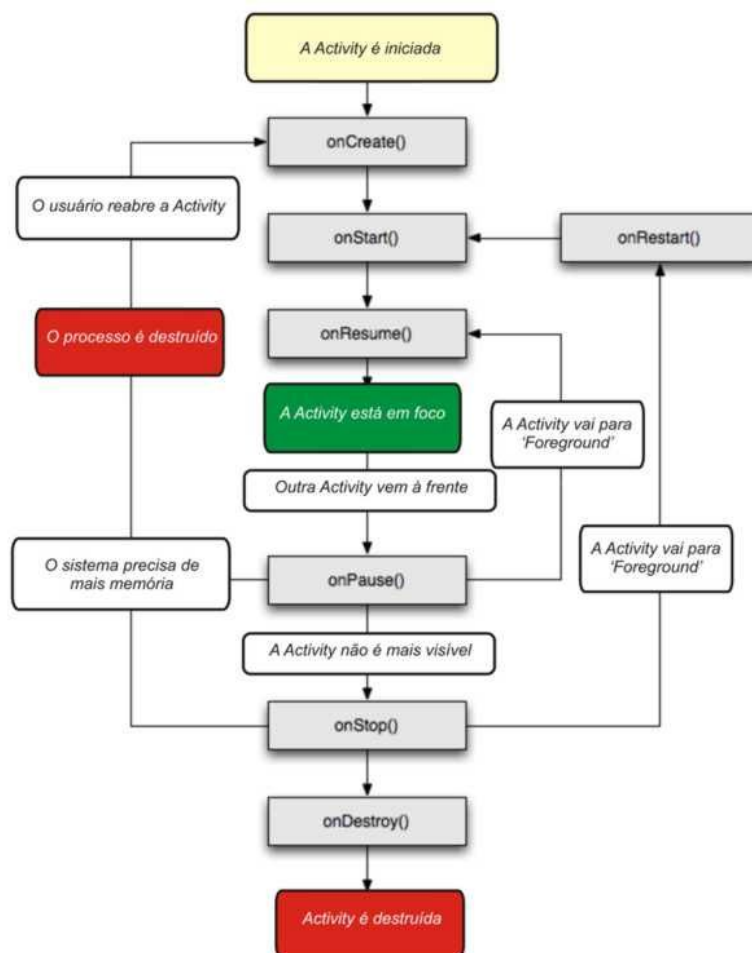


Figura 1: Ciclo de Vida da Activity .

Outro conceito utilizado para a implementação do aplicativo foi o de *Services*.

Um *Service*, seja em qual plataforma for, é uma aplicação que executa tarefas em segundo plano, ou em "background". A plataforma Android já tem em sua API um modelo de componente específico para criar serviços. No caso do IFControl, os serviços foram utilizados para manter fixas as conexões Socket com o servidor enquanto o aplicativo estivesse em uso.

13. Android Studio (IDE)

Android Studio é um ambiente integrado de desenvolvimento (IDE) para desenvolvimento para a plataforma Android. Esse IDE foi anunciada em 16 de maio de 2013 na conferência Google I/O pela Gerente de Produtos da Google, Katherine Chou, sendo de livre uso sob a Licença Apache 2.0.

O Android Studio utiliza um novo sistema de construção com base em Gradle que proporciona flexibilidade, preferencias de compilação, resolução de dependências e muito mais. Este novo sistema de construção permite que você construa seus projetos no IDE, bem como em seus servidores integrações contínuas. A combinação permite que você gerencie facilmente configurações de compilação complexas nativamente, em todo o seu fluxo de trabalho, em todas as suas ferramentas. Esse processo é muito utilizado no uso de bibliotecas externas ao IDE, que geralmente estão disponibilizadas online pela plataforma *Github*.

Além de possuir um poderoso editor de código em Java e XML, permitindo a fatoração de uma alteração que percorra todo o projeto e a rápida indentificação de *bugs*, o ambiente de desenvolvimento possui uma interface GUI dinamica e rica em detalhes, permitindo a visualização da interface de seu projeto em varios tipos de smartphones ou tablets, assim como recursos de “segurar e arrastar” elementos visuais do aplicativo.

CAPÍTULO III

O PROTÓTIPO DO APLICATIVO

Nesta seção será apresentado o protótipo do projeto final, com todas suas funcionalidades e telas. Além dos serviços, o banco de dados foi “povoado” para o teste.

1. Diagramas UML

Neste trabalho o programa *Astah* foi utilizado para descrever os diagramas de casos de uso e o de classes que irão mostrar as funcionalidades de processos que ocorrem no projeto IFCONTROL.

Diagrama de casos de uso

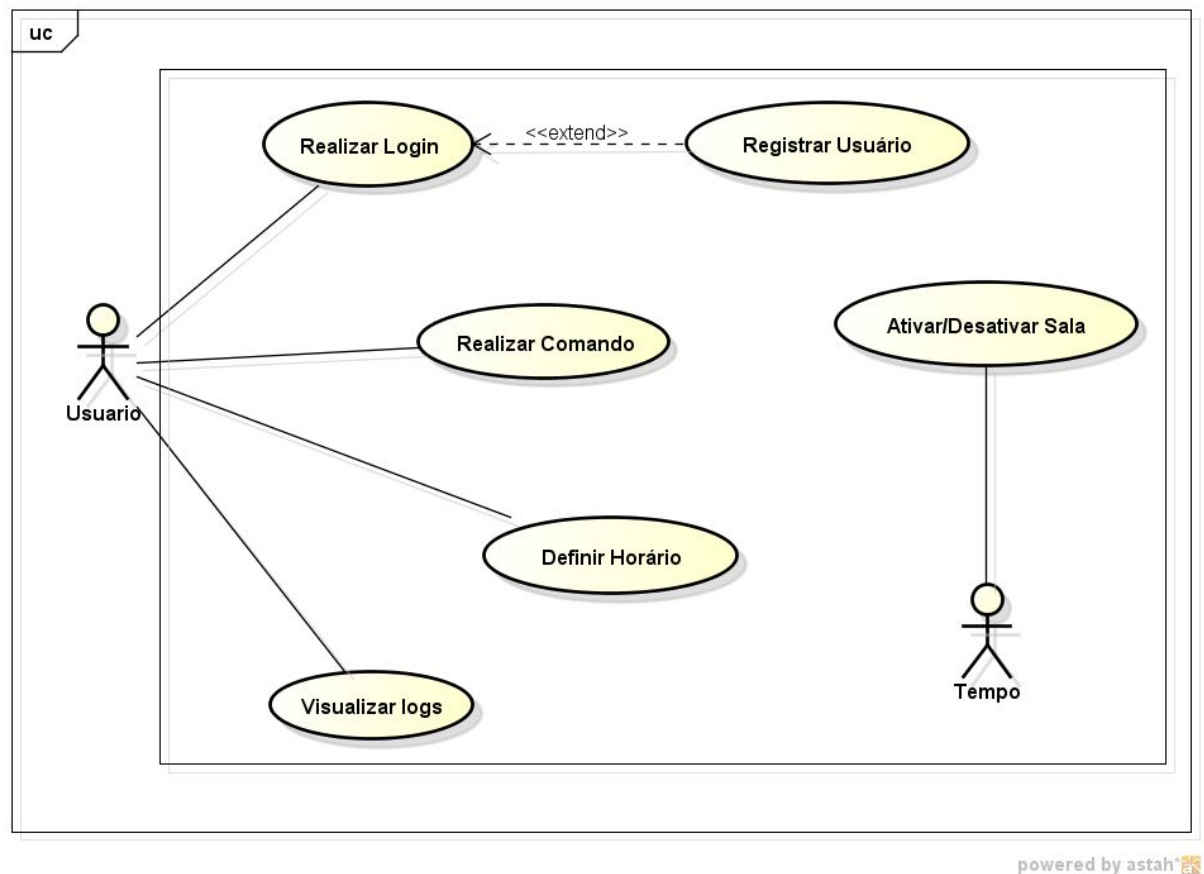


Figura 2: Diagrama de caso de uso

O diagrama apresentado na Figura 2 mostra os casos de uso de cada ator no IFCONTROL. Para os casos de uso Realizar Comando, Definir Horário e Visualizar Logs é necessário que o usuário passe pelo caso de uso Realizar Login, contudo, optou-se por não representar essa obrigatoriedade no diagrama, por meio do esteriótipo “include”, para criar um diagrama de fácil entendimento. O único esteriótipo utilizado no diagrama foi o “extend”

que mostra que o caso de uso Registrar Usuário é uma opção de escolha do usuário ao passar pela tela de login.

Observa-se que os casos de uso Definir Horário e Ativar/Desativar Sala estão ligados, uma vez que, quando o usuário define um horário de ativação ou desativação da sala, o sistema é encarregado de realizar essas ações nas horas determinadas. No caso, o ator Tempo representa o temporizador que está localizado no servidor responsável por efetuar tais ações.

1.1. Diagrama de Classes: Servidor

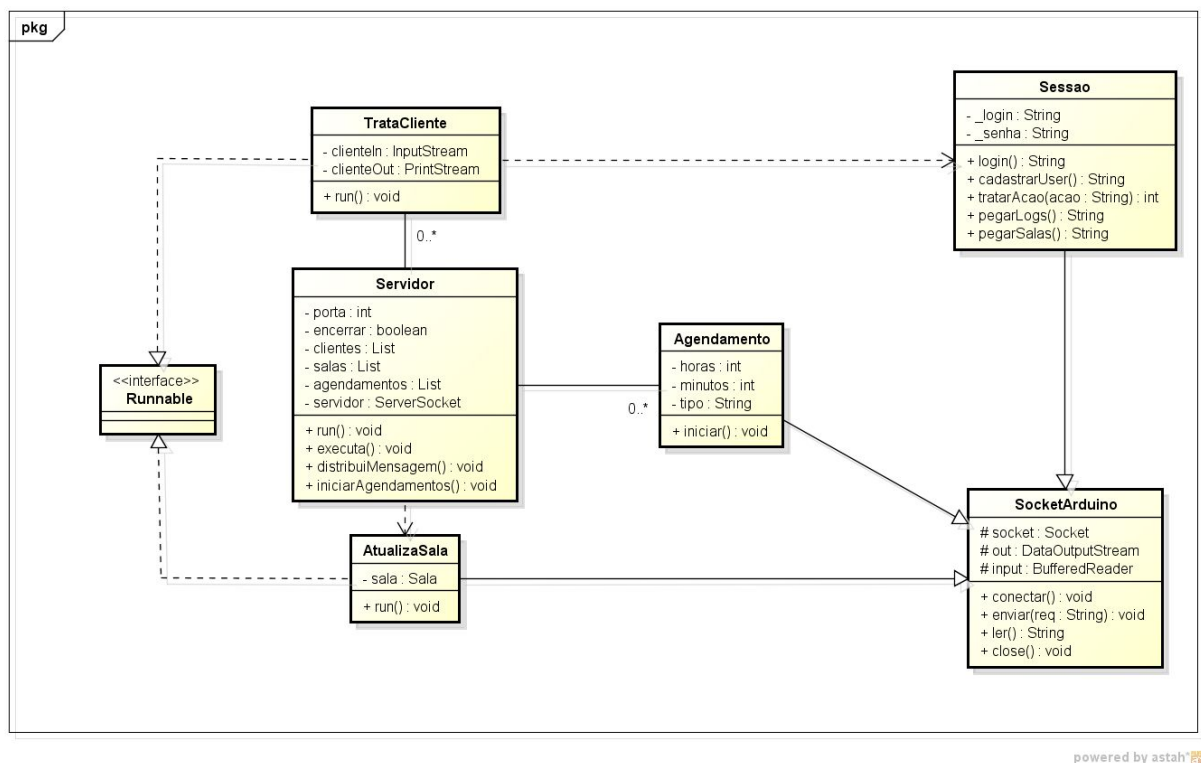


Figura 3: Diagrama de classes do servidor

O diagrama apresentado na Figura 3 mostra as classes utilizadas no servidor do “IFCONTROL”. A classe “Servidor” é responsável por armazenar a lista de agendamentos (horários de ativação e desativação) e por armazenar todas as conexões Socket que aceita. As

classes que lidam com esses agendamentos e conexões são, respectivamente, as classes “Agendamento” e “TrataCliente”. A relação que Servidor possui com essas classes é de “um para zero ou muitos” pois o Servidor pode possuir nenhum ou muitos agendamentos e conexões, porém estes só utilizam de um Servidor. A classe Servidor é responsável, também, por iniciar a classe “AtualizaSala” que, juntamente com a classe “TrataCliente”, é uma *Thread*, pois precisa estar constantemente atualizando os dados de temperatura e presença da sala e atualizando-os no banco de dados. Por padrão, todas as *Threads* implementam a interface “Runnable”, que possui o método “run”.

De modo semelhante, a classe “TrataCliente” fica constantemente em funcionamento, de modo a manter a conexão com o cliente, sendo assim, cada conexão está relacionado a uma classe “TrataCliente”. A relação dessa classe com “Sessão” é de dependência pois “TrataCliente” depende dos resultados das operações realizadas por “Sessão” no banco de dados e no Arduino. Dependendo dos resultados, “TrataCliente” é encarregado de formar textos JSON de diferentes dados e enviá-los ao seu cliente ou todos, dependendo da operação.

As classes “Sessão”, “Agendamento” e “AtualizaSala” possuem uma relação de herança com “SocketArduino”, porque todos realizam operações no Arduino como: conectar, enviar requisições, receber dados e encerrar a conexão.

1.2. Diagrama de Classes: Modelo de Dados

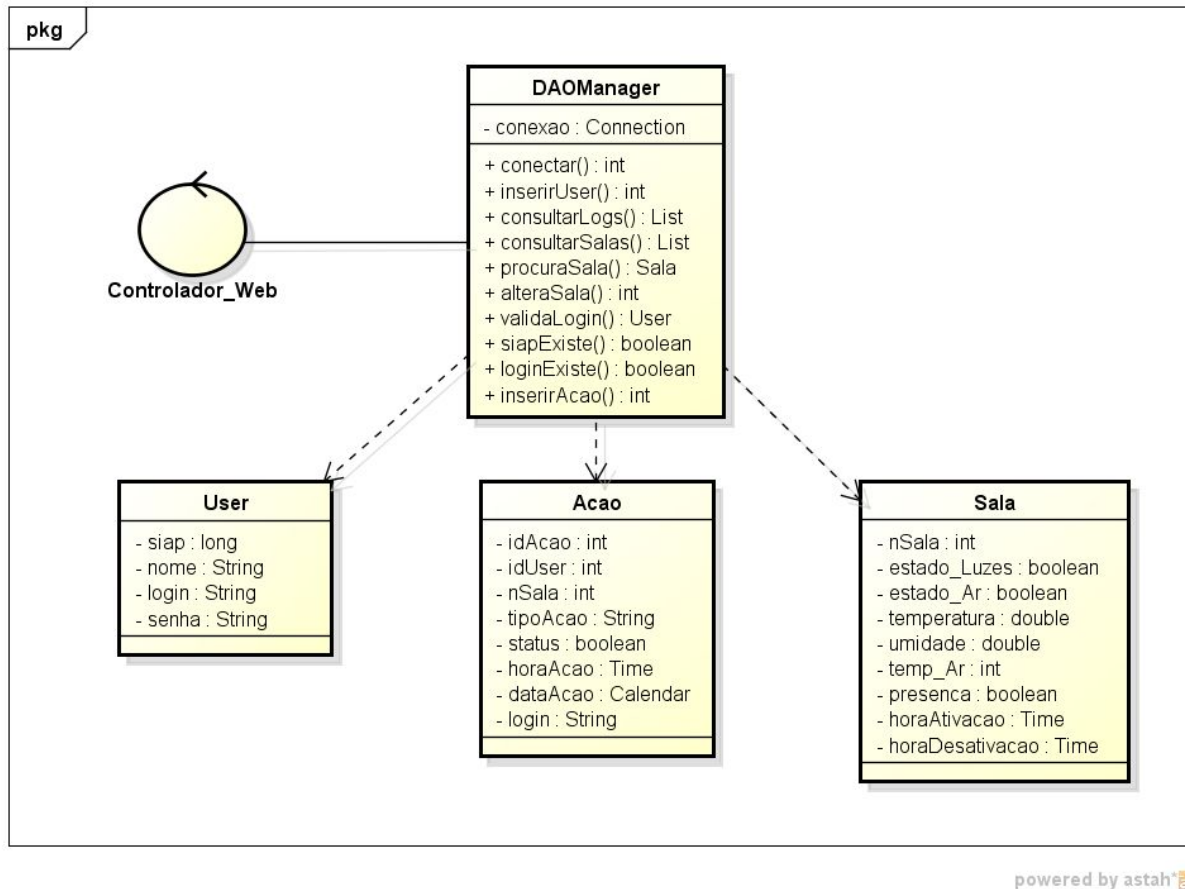
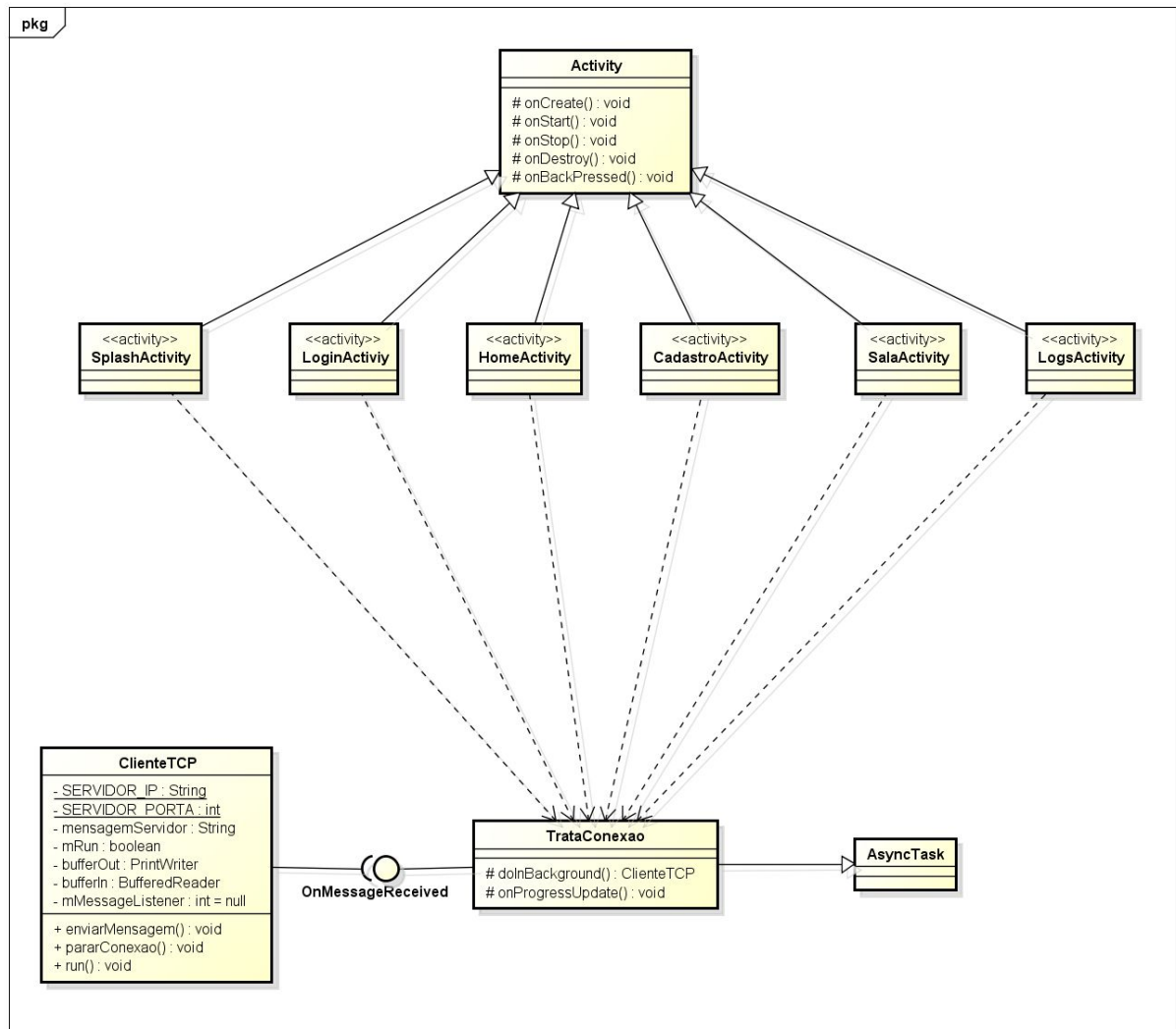


Figura 4: Diagrama de classes do modelo de dados

Seguindo o padrão MVC (Model-View-Controller), dividiu-se as classes que lidam com o banco de dados na parte *Model* do projeto, que é mostrado na Figura 4 acima. A classe “DAOManager” server como uma junção de todos os DAO (Data Access Object) do sistema, ou seja, os objetos que dão acesso ao dados reais. Essa junção é feita por meio de uma relação de dependência com as classes “User”, “Acao” e “Sala”, todas possuindo sua respectiva tabela no *MySQL*.

A interação dessa camada com a mostrada na Figura 3 se mostra no diagrama por meio do estereótipo “control” representado por um círculo.

1.3. Diagrama de Classes: Aplicação Android



powered by astah

Figura 5: Diagrama de classes da aplicação Android.

O diagrama apresentado na ‘Figura 5’ mostra as classes utilizadas na aplicação Android do “IFCONTROL”. Todas as *Activities* são representadas no diagrama possuindo

uma relação de herança com a classe “Activity”, que possui os métodos característicos de qualquer *Activity*.

A classe “TrataConexao” é uma “AsyncTask”, ou seja, age como um serviço que é iniciado ou encerrado dinamicamente. A manutenção da conexão com o servidor se dá por meio da classe “ClienteTCP”, que requiere uma interface chamada “OnMessageReceived”. A classe “TrataConexao”, por sua vez, fornece essa interface para “ClienteTCP”, que no caso se refere ao modo que a activity vai lidar com as mensagens do servidor que forem recebidas. Essa relação está especificada no diagrama acima de acordo com as especificações da UML 2.0.

2 TELAS DO PROTÓTIPO

2.1 Login

A primeira tela do aplicativo *Desktop* e o aplicativo *mobile* podem ser observados, respectivamente, nas figuras 6 e 7.

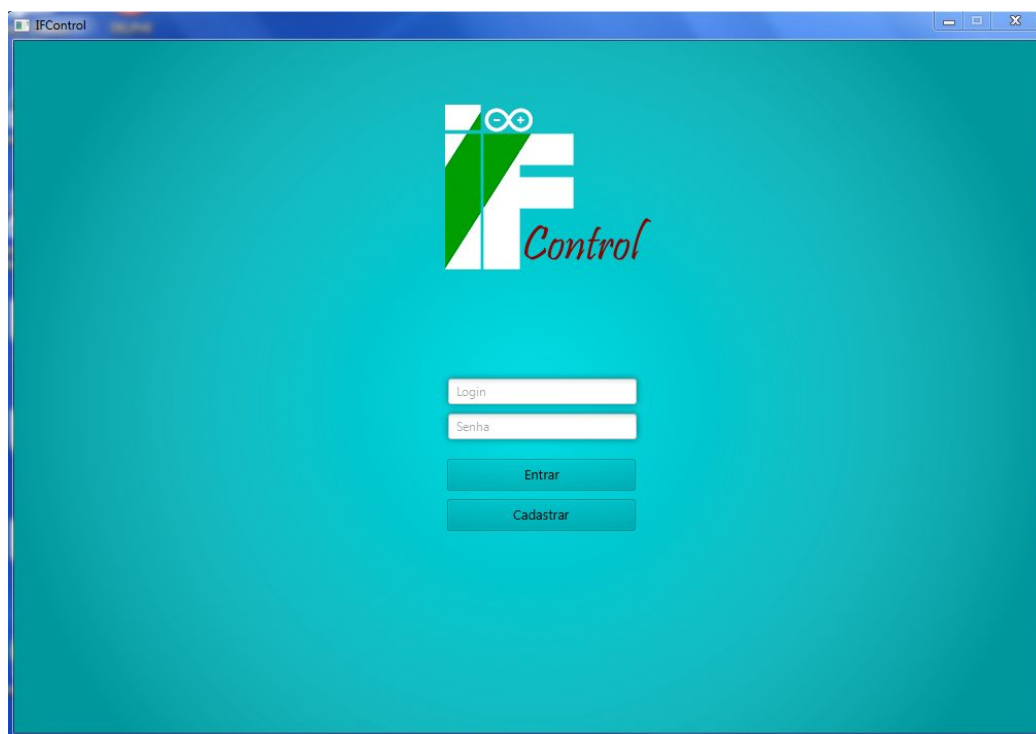


Figura 6: Tela de Login Desktop

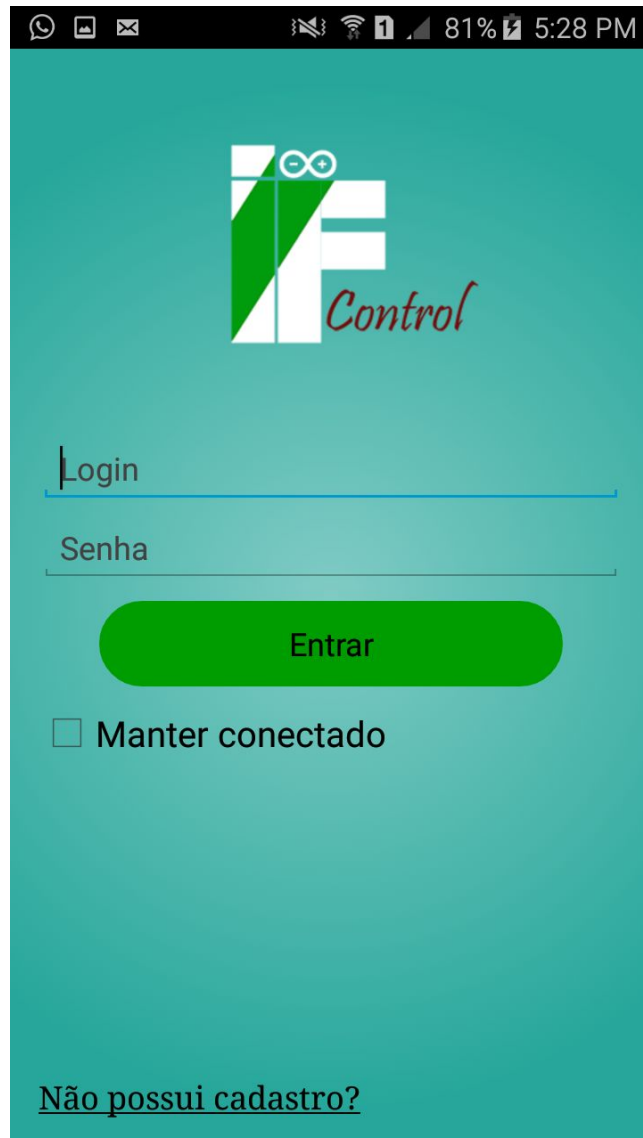
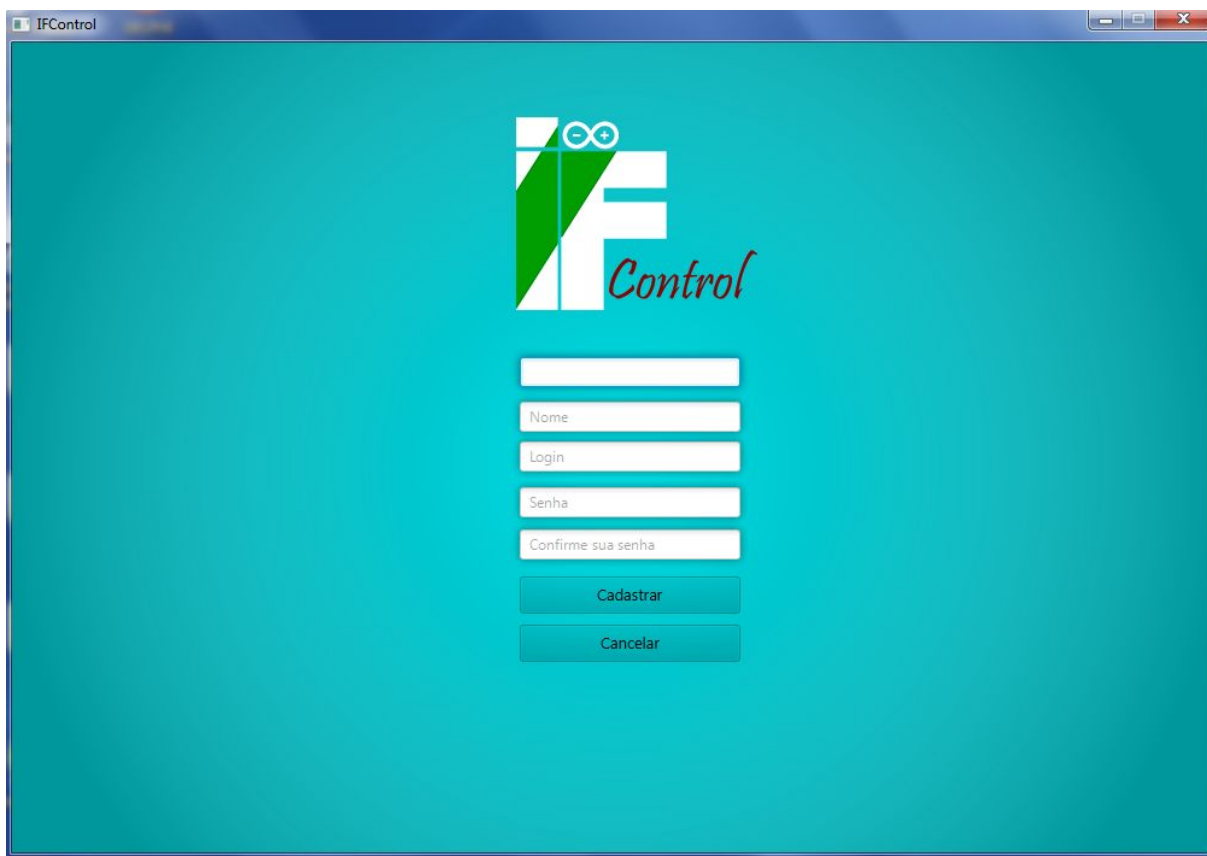


Figura 7: Tela de Login Mobile

As telas de *Login*, apresentadas nas figuras 6 e 7, são acessadas logo após a execução do aplicativo em suas respectivas versões.

2.2 Telas de Cadastro

As telas de cadastro, representadas pelas figuras 8 e 9, podem acessadas pela opção “Cadastro” na versão *Desktop* (correspondente a figura 8) e pela opção “Não possui cadastro?” na versão *mobile* (correspondente a figura 9).



The image shows a desktop application window titled "IFControl". The background is a solid teal color. In the center, there is a logo consisting of a stylized green and white geometric shape with the word "Control" in a red script font. Below the logo, there are five white input fields stacked vertically. The first field is empty. The second field is labeled "Nome". The third field is labeled "Login". The fourth field is labeled "Senha". The fifth field is labeled "Confirme sua senha". Below these fields are two teal buttons: "Cadastrar" and "Cancelar".

Figura 8: Tela de Cadastro Desktop

The image shows a mobile application interface for registration. At the top, there is a status bar with icons for WhatsApp, a gallery, email, and system status (81% battery, 5:28 PM). Below this is a teal header bar with a back arrow, a logo, and the text 'Cadastro'. The main form area is light gray and contains four labeled input fields: 'SIAP:', 'Nome:', 'Login:', and 'Senha:'. Each field has a corresponding text input line. At the bottom of the form is a large, rounded green button with the text 'Cadastrar' in white.

Figura 9: Tela de Cadastro mobile

Em ambas as versões *Desktop* e *mobile*, os campos a serem necessariamente preenchidos são o SIAP do funcionário, o Nome do funcionário, o Login do funcionário e a Senha da conta. Após o preenchimento dos campos com valores válidos, o usuário deve selecionar a opção “Cadastro”, sendo realizado o cadastro e redirecionando o usuário para a página de *login*.

2.3 Tela Home

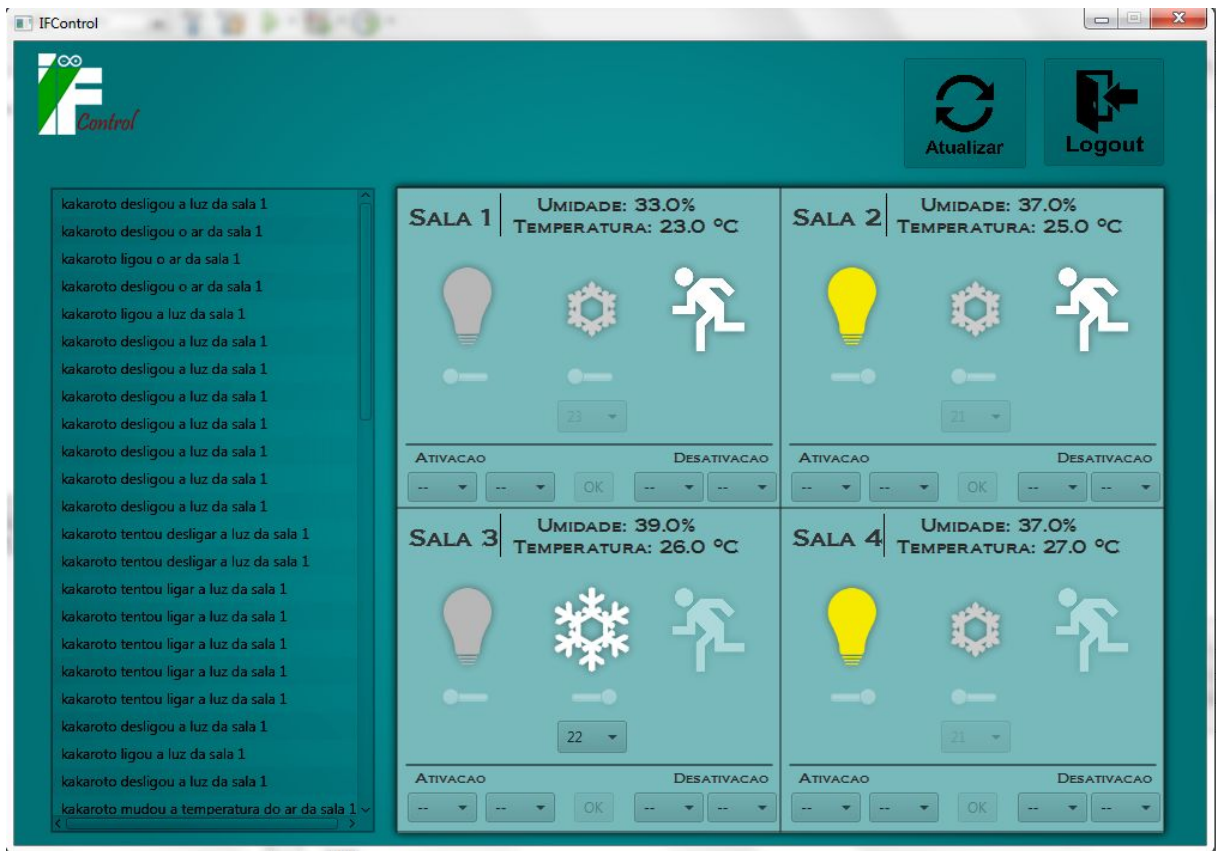


Figura 10: Tela Home Desktop.

Na figura 10 é apresentada a tela *Home* do aplicativo *Desktop*, onde é disponível o controle de salas, controle de *logs* e a saída da conta (*logout*). O controle de salas é realizado através da representação das salas em forma de “quadrados” com os números das respectivas salas. Dentro desses quadrados é possível visualizar as informações de umidade e temperatura.

Também é possível visualizar o estado atual das luzes, que é representado pelo desenho de lâmpada. Caso o desenho esteja amarelo, as luzes estão ligadas, caso o desenho esteja cinza, as luzes estão desligadas. Já o estado atual dos condicionadores de ar é

representado pelo desenho de um floco de neve. Caso o desenho esteja branco, os condicionadores estão ligados, caso o desenho esteja cinza, os condicionadores estão desligados.

Caso haja alguém na sala o símbolo de presença, representado por uma pessoa, se torna branco; caso não haja, o símbolo se mostra cinza. Os horários pré-definidos de ativação e desativação de todos os equipamentos da sala (luzes e condicionadores de ar) e a temperatura atual configurada nos condicionadores de ar também se apresentam no quadrado respectivo a sala a qual esse dados são referentes.

Para a alteração de estado (ligado ou desligado) das luzes ou condicionadores, o usuário deve selecionar o “interruptor” localizado logo abaixo da figura. Para a alteração de hora de ativação/desativação ou temperatura dos condicionadores, o usuário deve selecionar as “caixas” embaixo de suas respectivas funções. Ao lado do controle de salas, pode-se ver a lista de ações já realizadas (*logs*) por todos os usuários, e acima do controle de salas, pode-se observar a opção de *logout*.



Figura 11: Tela Home Mobile.

A tela *Home* da versão mobile apresenta a mesma representação de salas em forma de “quadrados” mas, diferente da versão *Desktop*, apresenta apenas a visualização do número da sala, o estado atual das luzes, o estado atual dos condicionadores de ar e se há alguém na sala. Para poder controlar uma sala e obter informações de temperatura ou umidade o usuário deve selecionar a sala desejada para a realização de tais fins. Os *logs* também não estão disponíveis na tela *home* da versão mobile, para acessá-los o usuário deve selecionar a opção “*logs*”, representada pelo ícone de prancheta logo acima das salas. E para a realização do *logout* deve ser selecionada a opção “Opções”, representada pelo ícone de engrenagem logo ao lado da opção “*logs*”.

2.4 Tela Sala

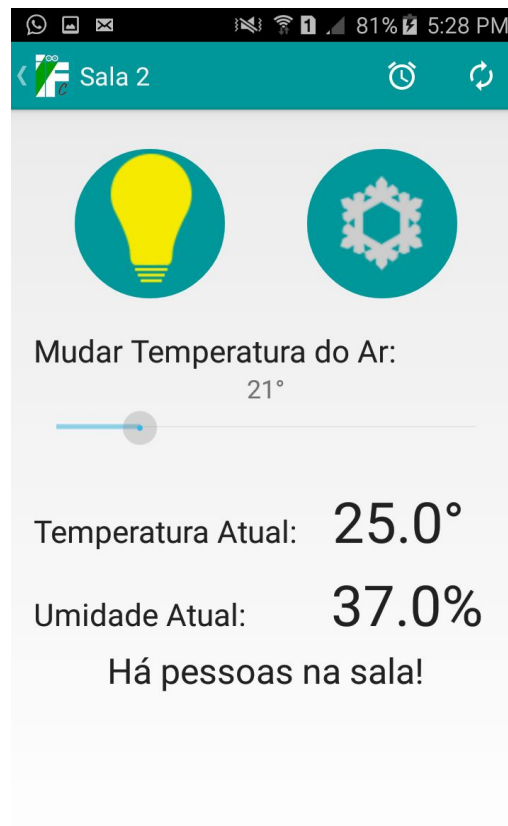


Figura 12: Tela de Sala.

Após selecionar a sala desejada na *Home* da versão mobile, o usuário será redirecionado para a tela “Sala” da sala selecionada. Na tela “Sala” estão disponíveis as opções para alterar o estado das luzes entre ligadas e desligadas (disponível ao selecionar o botão com o desenho de uma lâmpada, semelhante à versão *Desktop*, caso a lâmpada esteja amarela, as luzes estão ligadas, caso a lâmpada esteja cinza, as luzes estão desligadas), alterar o estado dos condicionadores de ar entre ligados e desligados (disponível ao selecionar o botão com o desenho de um floco de neve, semelhante à versão *Desktop*, caso o floco de neve

esteja branco e “grande”, os condicionadores estão ligados, caso o floco de neve esteja cinza e “pequeno”, os condicionadores estão desligados) e a sua respectiva temperatura (disponível através do “*slider*” logo abaixo da temperatura atual do ar), a temperatura/umidade atual na sala e se há ou não pessoas na sala. Para configurar um horário pré-definido para ativação/desativação da sala (semelhante à versão *Desktop*), o usuário deve seleccionar a opção “Horário” no cabeçalho da tela, sendo assim redirecionado para sua respectiva tela.

2.5 Tela Horário

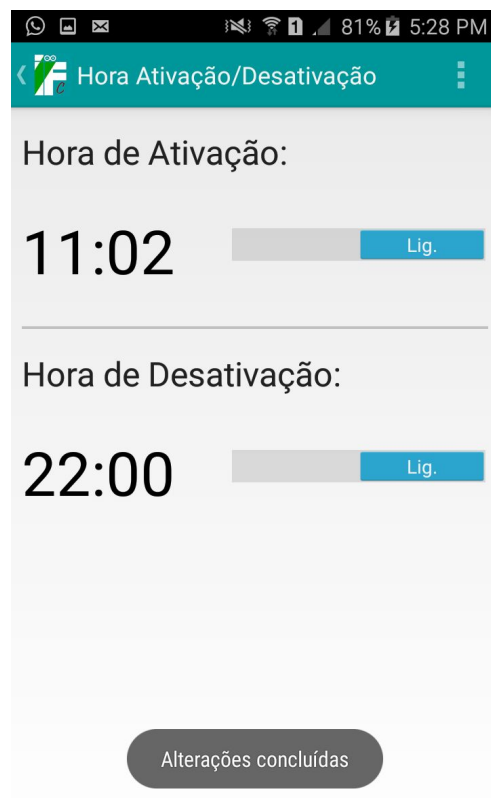


Figura 13: Tela Horário

Na tela “horário”, o usuário pode ligar/desligar horários pré-definidos ao selecionar o “interruptor” ao lado do respectivo horário e pode configurar novos horários ao selecionar a opção “Opções”, representada por um ícone “três-pontos”, no cabeçalho da tela. Para retornar a tela “Sala”, o usuário deve selecionar o ícone “IFc” também no cabeçalho da tela.

2.6 Tela logs

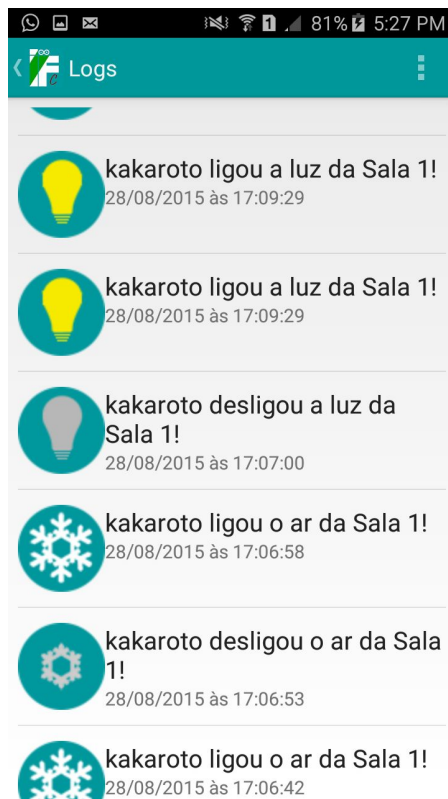


Figura 14: Tela logs.

Caso a opção “logs” seja selecionada na *Home*, o usuário será redirecionado para a tela “logs”, onde estão disponíveis, assim como na versão *Desktop*, a lista de ações realizadas por todos os usuários. Para voltar para a tela *Home*, o usuário deve selecionar o ícone “IFc” no cabeçalho.

CAPÍTULO IV

Conclusões

A necessidade de otimizar o controle de salas dentro da instituição fomentou o desenvolvimento deste trabalho. A prática do funcionário de ir de sala em sala para verificar o estado das luzes e condicionadores de ar, apesar de ainda ser comum, mostra-se ineficiente em uma instituição com um grande número de salas, além do gasto de energia caso um funcionário esqueça uma ou duas salas antes de fechar a instituição.

Dessa forma, o sistema foi criado com a intenção de suprir tal falha, facilitando o controle de salas e auxiliando no gasto de energia da instituição.

Este protótipo não teve a pretensão de superar a qualidade dos sistemas já existentes. Contudo, tem a característica de um sistema fácil de usar e bastante acessível. Fazendo com que este software tenha um diferencial em relação ao restante. Além disso, por se tratar de um sistema de cunho acadêmico, optou-se em utilizar as tecnologias estudadas no decorrer do curso.

Como resultado deste trabalho foi feita a exposição no estande do IFAM na Feira Norte do Estudante realizada no Manaus Plaza no período de 23 a 25 de setembro de 2015, assim como participação no I Congresso de Ciência, Educação e Pesquisa Tecnológica do IFAM na categoria de Mostra Científica e Cultural.

Referências Bibliográficas

Arduino (16 de Novembro de 2015). *Arduino*. Fonte:

<https://www.arduino.cc/en/Guide/Introduction>

MCROBERTS, Michael. *Arduino Básico*. São Paulo: Novated Editora, 2011.

Sparkfun (16 de Novembro de 2015). *Arduino*. Fonte:

<https://learn.sparkfun.com/tutorials/what-is-an-arduino>

Sparkfun (16 de Novembro de 2015). *Shield Ethernet*. Fonte:

<https://learn.sparkfun.com/tutorials/arduino-shields>

Arduino (16 de Novembro de 2015). *Shield Ethernet*. Fonte:

<https://www.arduino.cc/en/Main/ArduinoEthernetShield>

Wikipedia, the free enciplopedia. (16 de Novembro de 2015). *Java*. Fonte:

[https://en.wikipedia.org/wiki/Java_\(programming_language\)](https://en.wikipedia.org/wiki/Java_(programming_language))

Oracle (17 de Novembro de 2015) *What is a Socket?* Fonte:

<https://docs.oracle.com/javase/tutorial/networking/sockets/definition.html>

Gson (18 de Novembro de 2015) *Getting Started with Google Gson*. Fonte:

<http://blog.ajduke.in/2013/07/28/getting-started-with-google-gson/>

Oracle (19 de Novembro de 2015) *Java SE Technologies - Database* Fonte:

<http://www.oracle.com/technetwork/java/javase/jdbc/index.html>

Webopedia (19 de Novembro de 2015) *MySQL* Fonte:
<http://www.webopedia.com/TERM/M/MySQL.html>

Java (20 de Novembro de 2015) *General information on JavaFx* Fonte:
<http://java.com/en/download/faq/javafx.xml>

Oracle (20 de Novembro de 2015) *JavaFx Scene Builder* Fonte:
<http://www.oracle.com/technetwork/java/javase/downloads/javafxscenebuilder-info-2157684.html>

Wikipédia, a enciclopédia livre. (16 de Novembro de 2015). *Android Studio*. Fonte:
Wikipédia, a enciclopédia livre: https://en.wikipedia.org/wiki/Android_Studio
Android Developers (16 de Novembro de 2015). *Android Studio: An IDE built for Android*.
Fonte: <http://android-developers.blogspot.in/2013/05/android-studio-ide-built-for-android.html>

Sampaio, C. (01 de Dezembro de 2015). *Criação de serviços no Android*. Fonte:
<http://www.thecodebakers.org/2011/07/criacao-de-servicos-no-android.html>

Wikipédia, a enciclopédia livre. (01 de Dezembro de 2015). *Android*. Fonte: Wikipédia, a
enciclopédia livre: <https://pt.wikipedia.org/wiki/Android>