**UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ**
**CURSO DE ESPECIALIZAÇÃO JAVA**

**LUCAS AGNALDO DE MIRANDA CORREIA**

# APLICATIVO PARA DISPOSITIVOS MÓVEIS E SISTEMA WEB PARA GERENCIAMENTO DE EVENTOS

**TRABALHO DE CONCLUSÃO DE CURSO**

**PATO BRANCO**
**2015**

**LUCAS AGNALDO DE MIRANDA CORREIA**

# APLICATIVO PARA DISPOSITIVOS MÓVEIS E SISTEMA WEB PARA GERENCIAMENTO DE EVENTOS

Trabalho de Conclusão de Curso, apresentado ao III Curso de Especialização em Tecnologia Java, da Universidade Tecnológica Federal do Paraná, Câmpus Pato Branco, como requisito parcial para obtenção de título de Especialista.

Orientador: Prof. Dr. Fábio Favarim.

**PATO BRANCO**
**2015**

APLICATIVO PARA DISPOSITIVOS MÓVEIS E SISTEMA WEB PARA GERENCIAMENTO DE EVENTOS

Por

Lucas Agnaldo de Miranda Correia

Esta monografia foi apresentada às 19h00 do dia 21 de outubro de 2015 como requisito parcial para a obtenção do título de ESPECIALISTA, no III curso de Especialização em Tecnologia Java, da Universidade Tecnológica Federal do Paraná, Câmpus Pato Branco. O acadêmico foi arguido pela Banca Examinadora composto pelos professores abaixo assinados. Após deliberação, a Banca Examinadora considerou o trabalho aprovado.

Prof. Dr. Fábio Favarim
Orientador
UTFPR – Campus Pato Branco

Prof. Msc. Vinicius Pegorini
Banca
UTFPR – Campus Pato Branco

Prof. Msc. Robison Cris Brito
Banca
UTFPR – Campus Pato Branco

Prof. Msc. Robison Cris Brito
Coordenador do curso de Especialização
UTFPR – Campus Pato Branco

## RESUMO

MIRANDA CORREIA. Lucas Agnaldo. Aplicativo para dispositivos móveis e sistema web para gerenciamento de eventos. 2015. 51f. Monografia (III Curso de Especialização Java - Universidade Tecnológica Federal do Paraná. Pato Branco, 2015.

Este trabalho apresenta o desenvolvimento de um aplicativo para o gerenciamento de eventos. O seu denvolvimento foi motivado pelo alto crescimento de desenvolvimento de aplicações para dispositivos móveis, principalmente para o sistema operacional Android. As empresas que trabalham na área de desenvolvimento de *software* precisam se adaptar e trabalhar com aplicativos para dispositivos móveis, pois o número de pessoas que possuem smartphones está crescendo ano a ano. Para o desenvolvimento foram realizadas pesquisas, sendo elas elaboradas através de contato com um gerente de projetos e empreendedor da área de *software*, o qual forneceu as informações necessárias para que fosse aplicada neste projeto de conclusão de curso. A área de eventos, por tratar-se de um segmento com poucos aplicativos desenvolvidos, esse estudo representa um diferencial, podendo contribuir de uma forma significativa para os seus usuários.Sendo que os resultados foram satisfatórios, tendo em vista as tecnologias trabalhadas são otimas ferramentas que fornecem ao desenvolvedor ótimos recursos para poder desenvolver o seu produto, assim podendo desenvolver aplicativo integrado com facebook, ferramenta de base de dados ótima para segurança e integridade dos dados, assim também como a camada de interface onde JSF e Primefaces deram ótimos resultados.

**Palavras-chave:** Controle de Eventos, Android. Aplicação Móvel.

## LISTA DE ABREVIATURAS E SIGLAS

| | |
|---|---|
| API | *Application Programming Interface* |
| EJB | *Enterprise Java Beans* |
| HTML | *HyperText Markup Language* |
| HTTP | *HyperText Transfer Protocol* |
| IDE | *Integrated Development Environment* |
| JSF | *Java Server Faces* |
| SQL | *Structured Query Language* |
| UML | *Unified Modeling Language* |
| XML | *Extensible Markup Language* |

## LISTA DE FIGURAS

# LISTA DE QUADROS

# LISTAGENS DE CÓDIGOS

**SUMÁRIO**

# 1 INTRODUÇÃO

## 1.1 CONSIDERAÇÕES INICIAIS

Até pouco tempo, quando os eventos da cidade, casas noturnas, festas particulares e viagens eram agendadas, a divulgação era na maioria dos casos realizada pelas rádios, e ainda é, porém agora para divulgação existe outras opções, como as redes sociais, sites e aplicativos que auxiliam e organizam na divulgação dos eventos.

Atualmente, com os recursos presentes e acessíveis, os usuários estão sempre buscando na área da tecnologia aplicativos e serviços que gerenciem e simplifique seu trabalho. Assim abriu-se uma área para profissionais explorarem e aumentarem a demanda dos serviços para esse tipo de mercado. Os aplicativos sociais são exemplo disso, como: *Whatsapp* (aplicativo de mensagem instantânea), *Instagram* (aplicativo de compartilhamento de imagens), *Linkedin* (aplicativo profissional de emprego), auxiliam as pessoas no seu cotidiano, permitindo com que as pessoas possam agendar eventos entre amigos, compartilhar imagens dos lugares por onde passaram, arrumar novo emprego, enfim é infinita a possibilidade em que os aplicativos *mobile* estão auxiliando na vida do ser humano.

Além desses software de uso mais geral, uma nova porta para *softwares* exclusivos tem surgido, sendo que estes devem atender as necessidades específicas. Entre estes estão os softwares para atender a área de eventos.

## 1.2 OBJETIVOS

Este trabalho de conclusão de curso trata do desenvolvimento de um aplicativo para o gerenciamento de eventos e a seguir serão descritos seus objetivos, geral e específico.

### 1.2.1 Objetivo Geral

Desenvolver um aplicativo exclusivo para agendar eventos, organizar e gerenciar a organização e execução de tarefas do usuário.

### 1.2.2 Objetivos Específicos

Para atingir o objetivo geral deste trabalho de conclusão de curso os seguintes objetivos específicos foram estabelecidos:

- Desenvolver um aplicativo móvel para o gerenciamento de eventos;

## 1.3 JUSTIFICATIVA

O setor de eventos apresenta certa carência de sistemas gerenciadores específicos, por tratar-se de uma área em recente crescimento. Na maioria das vezes o controle é feito via rede social *Facebook,* ou seja, o usuário conhece uma ferramenta única, que ainda não é específica, pois se trata de uma rede social.

Assim, é proposto o desenvolvimento de um sistema para gerenciamento de evento, buscando suprir essa carência, permitindo melhorar a forma de gerenciamento dos eventos nesse segmento. Para os empresários irá auxiliar em diversos fatores, como aceleramento da produtividade, divulgações, segurança das informações, organização dos eventos.

## 1.4 ORGANIZAÇÃO DO TEXTO

Este trabalho de conclusão de curso está organizado em capítulos, dos quais este é o primeiro e apresenta a justificativa do trabalho e os objetivos gerais e específicos.

O segundo capítulo mostra a apresentação o mercado, o domínio da aplicação definindo o desenvolvimento de *software*, sistemas de informação, sistemas de apoio gerencial e a UML apresentando como funciona a modelagem nos sistemas.

Já no terceiro capítulo são descritos os recursos tecnológicos utilizados no desenvolvimento do sistema proposto, entre eles, Eclipse**,** ferramenta utilizada para desenvolver aplicativo *Android* e o portal *Web*, PostgreSQL**,** ferramenta de gerenciamente do banco de dados, Primefaces, ferramenta *design* das telas do portal *Web*, Hibernate, *framework* que realiza coenão do banco de dados com aplicação Java, Facebook APHI Graph, *framework* que realiza a integração do facebook com Android, Android, linguagem de desenvolvimento *mobile*.

São apresentadas, no quarto capítulo, a modelagem, requisitos funcionais, diagramas e a descrição do sistema proposto através de algumas imagens das janelas do *software*, características do funcionamento e particularidades da programação desenvolvida.

No capítulo cinco, são discutidas as perspectivas futuras.

Finalizando com a conclusão e as referências bibliográficas que fundamentaram este trabalho de conclusão de curso de curso.

# 2 REFERENCIAL TEÓRICO

Nesse capítulo é abordado alguns dados sobre o mercado de dispositivos móveis e como sistemas de informação e sistemas gerenciais contribuem para sociedade.

## 2.1 MERCADO DE DISPOSITOS MÓVEIS

A venda de smartphones subiu 55% no Brasil em 2014 na comparação com o ano anterior, alcançando 54,5 milhões de unidades, a consultoria IDC Brasil, projeta crescimento de 16% para 2015. O mercado de celulares encerrou 2014 em alta de 7%, com um total de 70,3 milhões de aparelhos comercializados. Dessa forma, o Brasil ficou na quarta colocação entre os maiores mercados do mundo, segundo pesquisa IDC Brasil, atrás de China, Estados Unidos e Índia. De acordo com o estudo, 95% do mercado está concentrado em seis grandes marcas (IDCBRASIL, 2015).

No quarto trimestre de 2014, foram vendidos 16,2 milhões de celulares inteligentes, alta de 43% frente ao quarto trimestre de 2013, com impulso das vendas na Black Friday. Citando dólar alto e conjuntura econômica, a IDC Brasil projeta crescimento de 16% para o mercado de smartphones em 2015, com a venda de cerca de 63,3 milhões de aparelhos  Em 2014, 15% dos aparelhos vendidos tinham tecnologia de Internet móvel de quarta geração (4G). Para o ano de 2015, a projeção é que essa participação fique entre 30 e 35%, segundo a pesquisa IDC Brasil. (IDCBRASIL, 2015).

Atualmente os principais sistemas operacionais que executam nos dispositivos móveis são: Android, da empresa Google; iOS, da Apple; Windows Phone, da Microsoft; e Symbian, da Accenture. (IDCBRASIL, 2015). O Android  no primeiro semestre de 2014 esteve presente em 91,6% dos smartphones vendidos no Brasil. Ainda de acordo com a pesquisa, os aparelhos com iOS  representaram 2% das vendas, seguidos pelo Windows Phone (1,8%) e Symbian (0,9%). Outros sistemas respondem por 3,7% do total. De acordo com Thiago Moreira, diretor da área de digital da Nielsen Ibope, a dominância do Android no Brasil é justificada porque o sistema é usado por vários fabricantes e em modelos de entrada, mais baratos. "Há variedade e preços convidativos", avalia Moreira. (IDCBRASIL, 2015).

## 2.2 SISTEMAS DE INFORMAÇÃO

Atualmente todos reconhecem que sistemas de informação são essenciais para os administradores, porque a maioria das organizações precisa deles para sobreviver e prosperar. Esses sistemas podem auxiliar as empresas a estender seu alcance a locais distantes, oferecer novos produtos e serviços, reorganizar fluxos de tarefas e trabalho, e talvez transformar radicalmente o modo como conduzem o negocio (LAUDON e LAUDON, 2004, pág. 4).

Os sistemas de informação podem ser definidos tecnicamente como um conjunto de componentes inter-relacionados que coleta (ou recupera), processa, armazena e distribui informações destinadas a apoiar as tomadas decisões, a coordenação e o controle de uma organização. Além de dar suporte à tomada de decisões, à coordenação e ao controle, esses sistemas também auxiliam os gerentes e trabalhadores a analisar problemas, visualizar assuntos complexo e criar novos produtos (LAUDON e LAUDON, 2004, pág. 7).

Eles contêm informações sobre pessoas, locais e coisas significativas para a organização ou para o ambiente que a cerca. No caso, informação quer dizer dados apresentados em uma forma significativa e útil para os serem humanos. Dados, ao contrario, são correntes de fatos brutos que representam eventos que estão ocorrendo nas organizações ou no ambiente físico, antes de terem sido organizados e arranjados de uma forma que as pessoas possam entendê-los e usá-los. (LAUDON e LAUDON, 2004, pág. 7).

Três atividades em um sistema de informação produzem as informações que as organizações necessitam para tomar decisões, controlar operações, analisar problemas, e criar novos produtos ou serviços. Essas atividades são a entrada, o processamento e a saída. A entrada captura ou coleta dados brutos de dentro da organização ou de seu ambiente externo. O processamento converte esses dados brutos em uma forma mais significativa. A saída transfere as informações processada as pessoas que as utilizarão ou às atividades em que serão empregadas (LAUDON e LAUDON, 2004, pág. 8). A Figura 1 ilustra as atividades descritas.

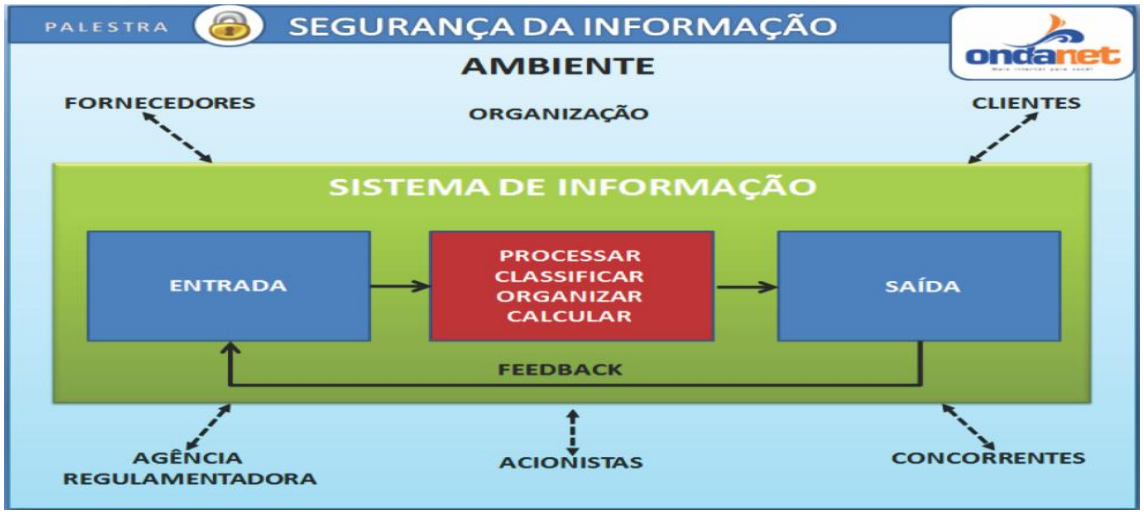



**Figura 1 – Representa a Entrada, Processamento e Saída de Dados.**

Fonte: AZEVEDO (2012).

## 2.3 SISTEMAS DE APOIO GERENCIAL

Quando os sistemas de informação se concentram em fornecer informação e apoio para a tomada de decisão eficaz pelos gerentes, eles são chamados sistemas de apoio gerencial. Fornecer informação e apoio para a tomada de decisão por parte de todos os tipos de gerente (dos altos executivos aos gerentes de nível médio e até os supervisores) é uma tarefa complexa. Em termos conceituais, vários tipos principais de sistemas de informação apoiam uma serie de responsabilidades administrativas do usuário final (O'BRIEN, 2002, pág. 29):

- Sistemas de informação gerencial (SIG);
- Sistemas de apoio à decisão;
- Sistemas de informação executiva.

Sistemas de informação gerenciais fornecem informação na forma de relatórios e exibições em vídeo para os gerentes. Sistemas de apoio á decisão fornecem suporte computacional direto aos gerentes durante o processo de decisão. Sistemas de informação executiva fornecem informação critica em quadros de fácil visualização para uma multiplicidade de gerentes (O'BRIEN, 2002, pág. 29).

Dessa forma os sistemas de informação gerenciais foram o tipo original de sistemas de apoio gerencial e ainda constituem de uma categoria importante de sistemas de informação. Um SIG gera produtos de informação que apoiam muitas das necessidades de tomadas de decisão da administração. Os relatórios, telas e respostas produzidas por esses sistemas fornecem

informações que os gerentes especificaram de antemão para o adequado atendimento de suas necessidades de informação (O'BRIEN, 2002, pág. 29).

# 3 MATERIAIS E MÉTODO

Nesse capítulo são abordadas as ferramentas que foram utilizadas para o desenvolvimento do aplicativo para dispositivos móveis e para o sistema web.

## 3.1 MATERIAIS

As ferramentas e as tecnologias utilizadas para as atividades de implementação e execução do sistema são:

a) Android: linguagem de desenvolvimento da GOOGLE;

b) Servlet: responsável por receber requisições HTTP;

c) Facebook *Plataform framework:* permite ler e gravar dados no aplicativo Android á partir do facebook;

d) JAVA: linguagem de desenvolvimento desenvolvida pela *Sun Microsystems* em 1995;

e) JSF: tecnologia que permite criar aplicações Java para Web utilizando componentes visuais;

f) PrimeFaces: biblioteca de componentes de código aberto para o JSF 2.0 com mais de 100 componentes;

g) Hibernate *framework:* responsável por realizar a persistência dos dados no Banco de Dados utilizando o Mapeamento Objeto-Relacional;

h) PostgreSQL: sistema de gerenciamento de banco de dados.

### 3.1.1 Android

É uma ferramenta para desenvolvimento e execução de programas para dispositivos móveis, robusta e de fácil utilização/aprendizagem, uma vez que essa tecnologia utiliza a

linguagem de programação Java, com algumas semelhanças com o Java ME, e para o desenvolvimento da interface visual, podem-se utilizar arquivos XML, simplificando o processo de desenvolvimento.

Para o desenvolvimento Android, duas IDEs se destacam: Eclipse e Netbeans, com predominância da primeira. Porém recentemente o Google anunciou o IntelliJ IDEA como base para o Android Studio, uma nova IDE para os desenvolvedores dessa plataforma, como pode ser observado na Figura 2.

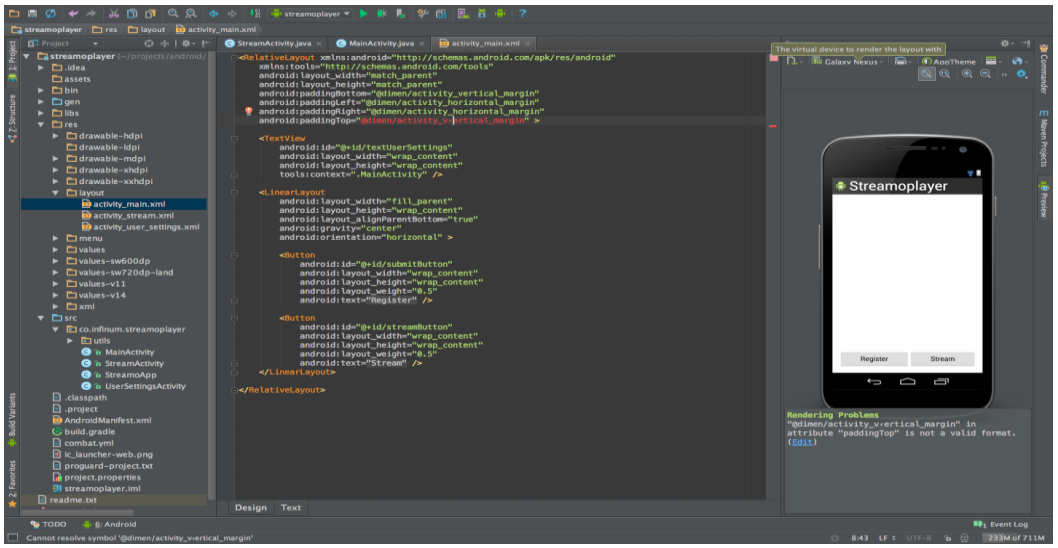



**Figura 2 – Modelo IDE Android.**

Fonte: ANDROID (2015).

No que diz respeito á infraestrutura de software, o Android consiste de uma pilha que engloba um sistema operacional baseado em Linux, conjunto de bibliotecas, uma API chamada Android Runtime, aplicações preexistentes no Android e aplicações diversas.

A IDE Android aposta nos smartphones e tablets, com grande poder de processamento e que integram vários recursos, como alta conectividade coma internet, GPS, sensores, telas sensíveis a toque e muito mais, porém, é possível a programação em outros dispositivos, como TVs, relógios ou óculos.

Uma das características mais importante do Android é a prioridade igualitária para aplicativos nativos e de terceiros.

### 3.1.2 Servlet

As Servlets são uma das formas  para criar páginas dinâmicas com Java.  O nome "servlet" vem da ideia de um pequeno servidor (servidorzinho, em inglês) cujo objetivo é receber chamadas HTTP, processá-las e devolver uma resposta ao cliente (CAELUM, 2015).

O funcionamento se dá da seguinte forma:

- Cliente (navegador) faz uma requisição HTTP ao servidor;
- O servlet responsável trata a requisição e responde ao cliente;
- O cliente recebe os dados e exibe.

Para se criar uma classe servlet, esta deve ser subclasse de HttpServlet e sobrescrever o método service(), que é responsável por receber as requisições e retornar suas respostas, como é ilustrado na Figura 3.

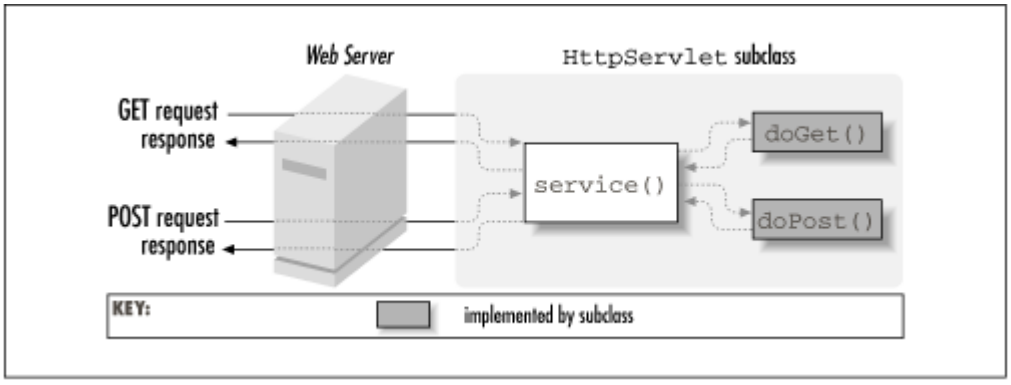



**Figura 3 – Estrutura básica de um Servlet.**

Fonte: SERVLET (2015).

Na Figura 4 é apresentada a uma nova classe Servlet no projeto.

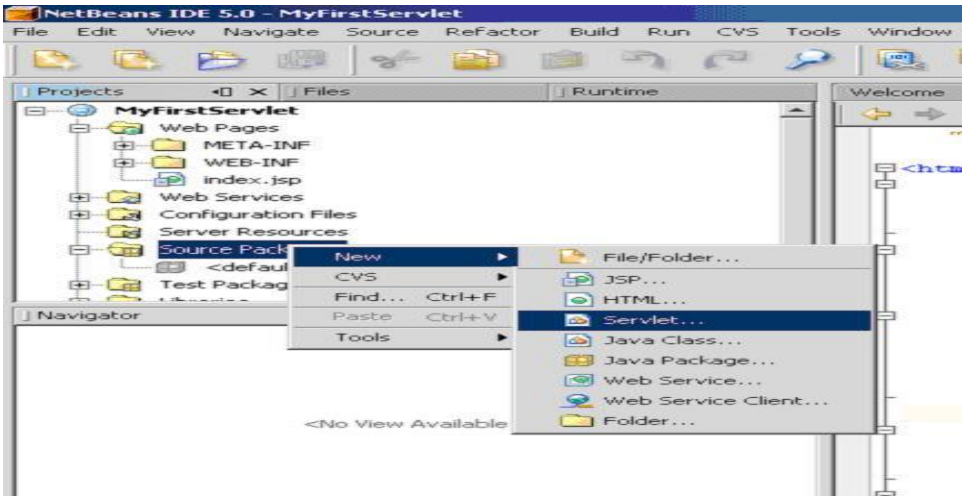



**Figura 4 – Classe Servlet.**

Fonte: SERVLET (2015).

Algumas das vantagens dos Servlets são (SERVLET, 2015):

- *Portabilidade* - Os servlets permitem serem movidos para outras aplicações Java e sistemas operacionais;
- *Desempenho* - Ao contrário da tecnologia CGI, cada solicitação é gerenciada pelo processo do container;
- *Agilidade* - Possui acesso às bibliotecas Java, que ajudam no ciclo do desenvolvimento.
- *Robustez* - Pelos servlets serem gerenciados pela JVM, não tem muitos problemas de memória.

### 3.1.3 Facebook Platform

Permite que qualquer um construa aplicativos sociais no Facebook e na web. Para permitir que você construa tais aplicativos, o Facebook oferece uma coleção abrangente de APIs e SDKs.

A API principal da Facebook Platform é a API Graph que permite que se leia e grave dados no aplicativo a partir do Facebook para uma nova forma que usa objetos (perfis de usuário, amigos, postagens, fotos, preferências, e assim por diante) e os relacionamentos ou conexões

entre eles. Essa abordagem simplifica a API do Facebook e a torna mais consistente ao trabalhar com objetos. É apresentada na Figura 5 modelo Facebook Platform API Graph (API GRAPH, 2015).

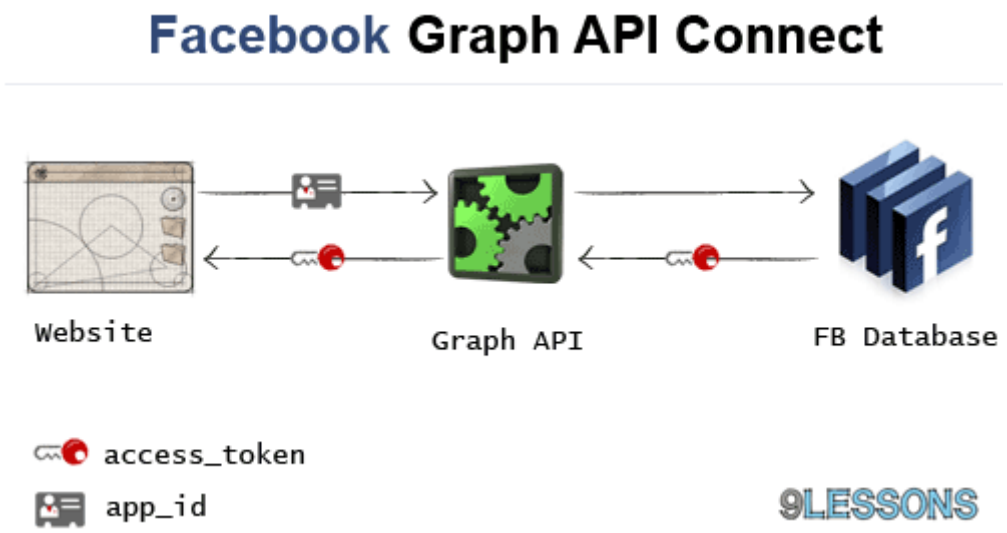



**Figura 5 – Facebook Platform API Graph.**
Fonte: API GRAPH (2015).

Para instalar Facebook Platform é preciso ter as seguintes qualificações e ferramentas:

- Conhecimento básico da tecnologia Java e como usar o Eclipse (ou seu IDE favorito).
- Java Development Kit (é requerida a versão 5 ou 6).
- Eclipse (versão 3.4 ou 3.5).
- SDK do Android e plug-in ADT. (API GRAPH, 2015).

Depois de instalado o Facebook Platform é necessário fazer a configuração da API. Desenvolvedor irá logar no facebook como usuário desenvolvedor e criar um projeto. Assim que projeto for criado receberá “Application_id” que será configurado no “manifest” do projeto Android.

### 3.1.4 JAVA

A linguagem Java foi desenvolvida pela *Sun Microsystems* em 1995. Apesar de relativamente nova, a linguagem obteve uma espetacular aceitação por programadores do mundo inteiro, tendo se difundido como nunca antes ocorreu com uma linguagem de programação. Um fator que colaborou com isso, é o fato da linguagem possuir vantagens agregadas tais como: orientação a objetos, independência de plataforma, multitarefa, robusta, segura e distribuída. Com o advento da Internet, que tornou ainda mais necessário a comunicação entre plataformas heterogêneas, estes fatores fizeram com que o Java fosse a tecnologia perfeita para este novo cenário.

A criação dos chips inteligentes foi abandonada devido ao alto custo de produção. Posteriormente, a explosão da Internet no mundo todo fez surgir a necessidade de uma tecnologia onde computadores de diferentes plataformas pudessem conversar. Surge assim, a linguagem Java. (JAVA, 2015)

Abaixo algumas características da linguagem.

- Orientada a Objetos : Paradigma atual mais utilizado na construção de softwares. Dentre suas vantagens, podemos citar reaproveitamento de código e aumento da manutenibilidade dos sistemas assim desenvolvidos.
- Simples e Robusta : Java representa em muitos aspectos um aperfeiçoamento da linguagem C++. Ela possui certas características que permitem a criação de programas de forma mas rápida, pois tiram do programador a possibilidade de cometer erros que são comuns de ocorrer em C++. Algumas dessas características são o tratamento obrigatório de exceções e o gerenciamento automático de memória.
- Gerenciamento Automático de Memória : Em Java não existe ponteiros, isto é, não é permitido ao programador acessar explicitamente uma posição de memória. Java automaticamente gerencia o processo de alocação e liberação de memória, ficando o programador livre desta atividade. O mecanismo responsável pela liberação de memória que não está mais sendo utilizada é conhecido como Garbage Collector.

- Independência de Plataforma : Um dos elementos chave da linguagem Java é a independência de plataforma. Um programa Java escrito em uma plataforma pode ser utilizado em uma outra distinta da original. Este aspecto da linguagem é geralmente referenciado como “write once, run anywhere”. Isto é conseguido através da utilização da Java Virtual Machine (JVM) a qual roda numa plataforma específica e interpreta um programa Java para código de máquina específico da plataforma em questão. Como os programas em Java executam sob o controle da JVM, eles podem rodar em qualquer plataforma que possua uma disponível.
- Multi-threading : Um programa Java pode conter múltiplas threads para realizar várias tarefas em paralelo.

A tecnologia Java está organizada em três plataformas com objetivos específicos (JAVA, 2015):

- **Java 2 Standard Edition (Java SE):** ferramentas e APIs (Application Program Interface) essenciais para qualquer aplicação Java (inclusive para as outras plataformas). É suficiente a utilizarmos se quisermos desenvolver aplicações desktop com ou sem interface gráfica.
- **Java 2 Enterprise Edition (Java EE):** ferramentas e APIs para o desenvolvimento de aplicações distribuídas. Engloba tecnologias tais como RMI, EJB, CORBA, JMS, etc.
- **Java 2 Micro Edition (Java ME):** ferramentas e APIs para o desenvolvimento de aplicações para aparelhos portáteis (palms, celulares, eletrodomésticos).

O JRE é um conjunto de programas que possibilita executar aplicações Java. O coração do JRE é a Máquina Virtual Java ou Java Virtual Machine (JVM). É a JVM que possibilita uma das características mais impressionantes da linguagem Java, a portabilidade do código (JAVA, 2015).

Na Figura 6 é mostrado o esquema de compilação. Código Java desenvolvido (.java), após seu desenvolvimento o código é compilado e gerado um (.class) e enviado para máquina virtual Java, assim essa máquina virtual Java é rodada tanto nos sistemas operacionas Windows, Linux e Mac.

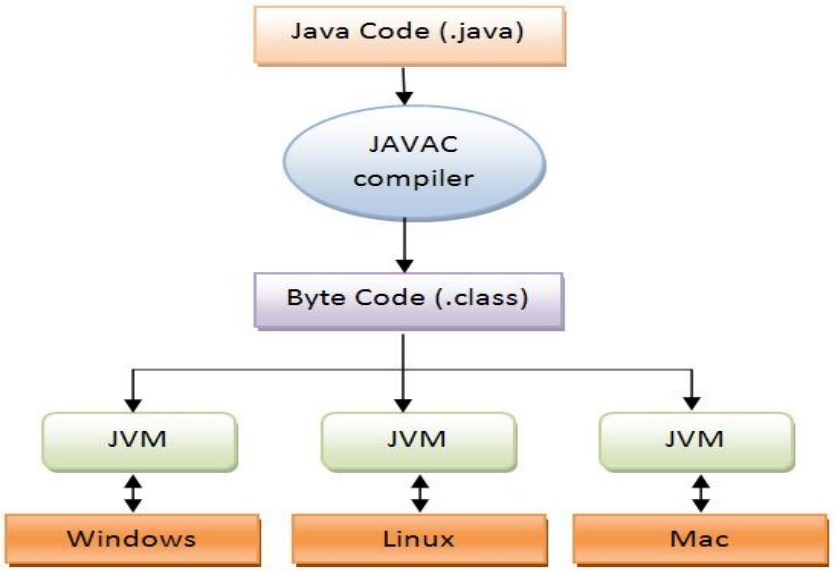



**Figura 6 – Esquema de compilação e execução de um programa em Java.**
Fonte: (JAVA -2015).

### 3.1.5 JSF

O JSF é uma tecnologia que nos permite criar aplicações Java para Web utilizando componentes visuais pré-prontos, de forma que o desenvolvedor não se preocupe com Javascript e HTML. Basta adicionarmos os componentes (calendários, tabelas, formulários) e eles serão renderizados e exibidos em formato html. (JSF, 2015).

Além disso o estado dos componentes é sempre guardado automaticamente, criando a característica Stateful. Isso permite, por exemplo, criar formulários de várias páginas e navegar nos vários passos dele com o estado das telas sendo mantidos. (JSF, 2015).

Outra característica marcante na arquitetura do JSF é a separação que realizada entre as camadas de apresentação e de aplicação. Pensando no modelo MVC, o JSF possui uma camada de visualização bem separada do conjunto de classes de modelo, ainda tem a vantagem de ser uma especificação do Java EE, isto é, todo servidor de aplicações Java tem que vir com uma

implementação dela e há diversas outras disponíveis. A implementação mais famosa do JSF e também a implementação de referência, é a Oracle Mojarra. (JSF, 2015).

Não há componentes sofisticados dentro da especificação e isso é proposital: uma especificação tem que ser estável e as possibilidades das interfaces com o usuário crescem muito rapidamente. A especificação trata do que é fundamental, mas outros projetos suprem o que falta.

Para atender a demanda dos desenvolvedores por componentes mais sofisticados, há várias extensões do JSF que seguem o mesmo ciclo e modelo da especificação. Exemplos dessas bibliotecas são PrimeFaces, RichFaces e IceFaces. Todas elas definem componentes JSF que vão muito além da especificação. (JSF, 2015).

### 3.1.6 Primefaces

De acordo com Robson Fagundes, PrimeFaces é uma biblioteca de componentes de código aberto para o JSF 2.0 com mais de 100 componentes. Ele é considerado muito melhor do que muitas outras bibliotecas de componentes JSF. (FAGUNDES, ROBSON, 2015).

Sendo um rico conjunto de componentes de interface (DataTable, AutoComplete, HTMLEditor, gráficos etc).

Os componentes são definidos por seus atributos configuráveis, que podem ou não está vinculado a classes Java denominadas Managed Bean, que são classes que fornecem ao desenvolvedor o acesso aos estados dos componentes na tela, ou seja, eles fazem a ligação entre interfacesgráficas e a lógica da aplicação. (FAGUNDES, ROBSON, 2015).

O Primefaces oferece os componentes mais populares utilizados em páginas na internet, e outros mais complexos, fazendo com que seja possível encontrar em sua biblioteca um elemento para quase todas as necessidades na construção das interfaces gráficas.

Dentre os principais componentes, cita-se: AutoComplete, BooleanButton, BooleanCheckbox, Calendar, Editor, InputMask, InputText,InputTextarea, Keyboard, ManyButton, ManyMenu, ManyCheckbox, OneRadio, Password, Slider, DataExporter, DataList, DataGrid, DataTable, GMap e muitos outros componentes distribuídos nas categorias Ajax Core, Input, Button, Data, Panel, Overlay, Menu, Charts, Message, Multimedia, File, DragDrop e Misc.

Na Figura abaixo é mostrado o componente Calendar, que como podemos visualizar pode ser implementado de diversas formas pelo desenvolvedor. (FAGUNDES, ROBSON, 2015).

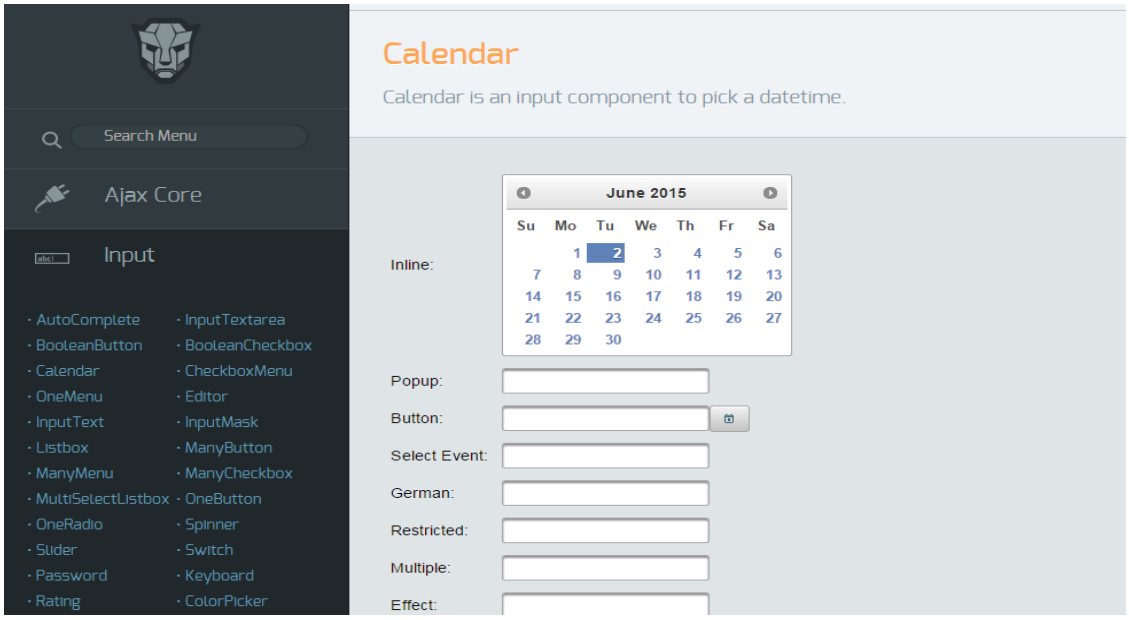



**Figura 7 – Componente Calendar.**
Fonte: PRIMEFACES (2015).

### 3.1.7 Hibernate

Segundo Adriel o Hibernate é um Java Framework responsável por realizar a persistência dos dados no Banco de Dados utilizando o Mapeamento Objeto-Relacional. Ele funciona como um MiddleWare, uma camada que está entre a aplicação e o Banco de Dados, servindo como um intermediário e diminuindo a complexidade da programação, reduzindo substancialmente o tempo de desenvolvimento. (ADRIEL, 2015).

Ele facilita o mapeamento dos atributos entre uma base tradicional de dados relacionais e o modelo objeto de uma aplicação, mediante o uso de arquivos XML para estabelecer esta relação. (ADRIEL, 2015).

Sua principal característica é a transformação das classes (POJO) e tipos de dados do Java para tabelas e tipos de dados SQL. O Hibernate gera as chamadas SQL (Select, Insert, Join, Triggers...) e libera o desenvolvedor do trabalho manual da conversão dos dados resultante, mantendo o programa portável para qualquer banco de dados SQL, porém causando um pequeno aumento no tempo de execução. Para que o Hibernate funcione, além de importar as APIs

necessárias para seu projeto, é preciso configurá-lo corretamente. Ele pode ser configurado de diversas formas:

- Utilizando uma instância de org.hibernate.cfg.Configuration;
- Através do arquivo hibernate.properties;
- Através do arquivo hibernate.cfg.xml.

Há um grande número de propriedades que controlam o comportamento do Hibernate em tempo de execução, sendo possível configurá-las em qualquer uma das técnicas citadas acima. Também é preciso mapear as classes POJO contendo a estrutura das tabelas, que por sua vez pode ser feito de duas maneiras:

- Com o uso de @Annotations;
- Com o uso do hbm.xml.

Os Annotations nos permite fazer o mapeamento diretamente no POJO utilizando o @ antes da declaração da classe, métodos e atributos. Já o hbm.xml (Hibernate Mapping) é um arquivo separado que possui a localização do POJO. A figura 8 mostra o papel do Hibernate em uma aplicação como foi descrito acima. (ADRIEL, 2015).


**Figura 8 – Demonstração do papel Hibernate em uma aplicação.**

Fonte: ADRIEL (2015)

### 3.1.8 PostgreSQL

O PostgreSQL não é um software novo. Seu desenvolvimento começou em 1986, na universidade de Berkeley, na Califórnia, com base no código-fonte do banco de dados INGRES. Por isso, na época ele foi chamado de Postgres - uma abreviação de Pos INGRES.

O projeto foi liderado pelo Professor Michael Stonebraker e financiado pela Agência de Pesquisa de Projetos Avançados de Defesa dos EUA (DARPA), pelo Departamento de Pesquisa do Exército Americano (ARO), pela Fundação Nacional de Ciência (NSF) e pela ESL, Inc. Em 1994 deixou de ser desenvolvido somente em Berkeley e seu código-fonte foi disponibilizado na Internet. Em 1995 ganhou suporte à linguagem SQL e finalmente em 1996 teve seu nome alterado para PostgreSQL. (SQL MAGAZINE – 2015).

Atualmente, o PostgreSQL utiliza a licença BSD, que dá total liberdade: qualquer pessoa pode alterar o código e utilizar o produto em operações comerciais, sem ter que pagar nada a ninguém. A única regra é que a Universidade de Berkeley não é responsável por nada que aconteça com o software. (SQL MAGAZINE, 2015).

Apesar da pouca popularidade, o PostgreSQL é um gerenciador de banco de dados veloz, robusto e que se encontra na lista dos que mais possuem recursos. Foi o pioneiro na introdução de vários conceitos objeto-relacionais que só recentemente foram implementados em alguns bancos de dados comerciais. Dentre as principais características, podemos citar:

- Suporte aos padrões ANSI SQL 89, 92 e 99;
- Suporte a transações;
- Sistema de concorrência de múltiplas versões dos dados (MVCC, no inglês): permite que gravações não bloqueiem leituras e vice-versa;
- Backup online;
- Integridade referencial;
- Herança de tabelas;
- Conexão por interfaces nativas;
- Suporte a várias linguagens de desenvolvimento (PHP, Java, C, ASP, .Net, Perl, Python, VB, Delphi, C++ Builder etc);
- Várias linguagens de programação de funções (plpgSQL, plTcl, plPerl, plPython, C etc);
- Tipos de dados definidos pelo usuário;
- Possui uma estrutura própria para os objetos do banco e não depende de nenhum engine externo, como BerkeleyDB ou InnoDB.

O PostgreSQL pode ser instalado nas plataformas Unix (Linux, FreeBSD, AIX, HPUX, Solaris, NetBSD, OpenBSD etc) – de fato, ele já faz parte da distribuição da maioria delas. (SQL MAGAZINE, 2015).

## 3.2 MÉTODO

O método utilizado para a realização do trabalho está baseado nas fases de análise, projeto, codificação e testes. Essas fases foram utilizadas como base, mas sofreram inclusões e alterações para adaptar-se aos objetivos deste projeto. Ressalta-se que as atividades realizadas não foram estritamente sequenciais. .As principais fases ou etapas definidas para o ciclo de vida do sistema são:

a) Requisitos – a definição dos requisitos do software foi realizada com a análise no processo de desenvolvimento do software.

b) Análise – a análise consistiu na definição dos requisitos funcionais e representá-los sob a forma de casos de uso.

c) Projeto – tendo como base os requisitos definidos na fase de análise foram definidos os modelos para solucionar o problema, ou seja, definir como o sistema atenderia aos requisitos (funcionalidades e requisitos não funcionais).

d) Implementação – na implementação foi realizada a codificação do sistema e realização de testes pelo programador, o autor deste trabalho.

# 4 RESULTADOS E DISCUSSÃO

Nesse capítulo são apresentados a descrição do sistema, os requisitos, a análise e implementação do sistema desenvolvido.

## 4.1 DESCRIÇÃO DO SISTEMA

O INHDGroups é um aplicativo Android e um portal web, moldado para atender usuários que desejam cadastrar eventos. Para o desenvolvimento foi utilizado o Android e o Java. O aplicativo tem como objetivo gerenciar eventos. Para tornar esse gerenciamento possível, conta com diversos controles, alguns dos quais serão descritos na sequência..

Entre os principais controles estão os agendamentos de eventos no qual o usuário que está prestando serviço poderá ter um controle de eventos. O sistema também conta com a integração com o facebook realizada pela FACEBOOK PLATAFORM subsessão 3.1.3, no qual usuário irá realizar a convite para os amigos do facebook que possuem o aplicativo instalado.

No portal web também é possível cadastrar evento, pois plataforma web é integrada com o aplicativo.necessidades dos usuários.

## 4.2 REQUISITOS DO SISTEMA

O objetivo da definição dos requisitos é especificar o que o sistema deverá fazer e determinar os critérios de validação que serão utilizados para que se possa avaliar se o sistema cumpre o que foi definido. Assim facilitando no desenvolvimento.

### 4.2.1 Requisitos Funcionais e Suplementares

No Quadro 1 são apresentados os requisitos funcionais.

| |
|---|
| **Funcionais:** |
| 1. Registrar novos eventos; |
| 2. Vincular conta do usuário ao Facebook; |
| 3. Confirmar presença. |
| 1. **Suplementares:** O sistema deverá operar via interface Mobile e Web. |

**Quadro 1 – Registros**

No Quadro 2 pode-se ver o detalhamento do requisito funcional de cadastro de evento, como será o processo que fará a inclusão de um novo evento no aplicativo. Também é possível ver algumas validações de campos que são obrigatórios para ser realizado o cadastro.

| |
|---|
| **Registrar novos eventos;**<br>**Descrição:** O usuário irá cadastrar novo evento, irá pedir ás informações que estão sendo solicitado no cadastro de evento, o usuário irá fazer o preenchimento no cadastro de evento.<br>**Fonte:** Usuário.<br>**Usuário:** Usuário.<br>**Informações de entrada:** O Usuário preenche os campos obrigatórios e não obrigatórios do cadastro do evento.<br>**Informações de saída:** Novo evento ativo do sistema.<br>**Restrições lógicas:** Não há.<br>**Restrições tecnológicas:** Alguns campos de dados serão de preenchimento obrigatório. |

**Quadro 2 – Registrar Novos Eventos**

No Quadro 3 é mostrado o processo de confirmação de presença no evento, a confirmação do usuário é realizada pelo aplicativo, nesse processo também existem campos que mostram as informações do evento.

| |
|---|
| **Confirmar presença evento;**<br>**Descrição:** O usuário convidado reberá notificação para participar de um evento, assim que usuário visualizar nome evento, local, data, hora e administrador ele poderá confirmar ou recusar presença no evento.<br>**Fonte:** Fulano de tal (convidado do evento)<br>**Usuário:** Usuário.<br>**Informações de entrada:** Usuário receberá informações do evento que foi convidado.<br>**Informações de saída:** Confirmação ou recusa do evento.<br>**Restrições lógicas:** Não há.<br>**Restrições tecnológicas:** Alguns campos de dados serão de mostrador sem poder editar. |

**Quadro 3 – Confirmar Presença Evento**

No Quadro 4 é mostrado o processo de verificar eventos criados.

| |
|---|
| **Verificar eventos criados;**<br>**Descrição:** O administrador irá verificar os eventos que ele criou.<br>**Fonte:** Administrador.<br>**Usuário:** Administrador.<br>**Informações de entrada:** Nome evento, data, hora, local e participantes.<br>**Informações de saída:** Não há.<br>**Restrições lógicas:** Não há.<br>**Restrições tecnológicas:** Não há. |

**Quadro 4 – Verificar Eventos Criados Pelo Administrador**

### 4.2.2 Diagrama de Caso de Uso

O diagrama de casos de uso mostra o ator e os eventos que o usuário ira realizar para completar um processo no aplicativo.

A Figura 10 mostra as funcionalidades de um cenário, do ponto de vista do usuário.

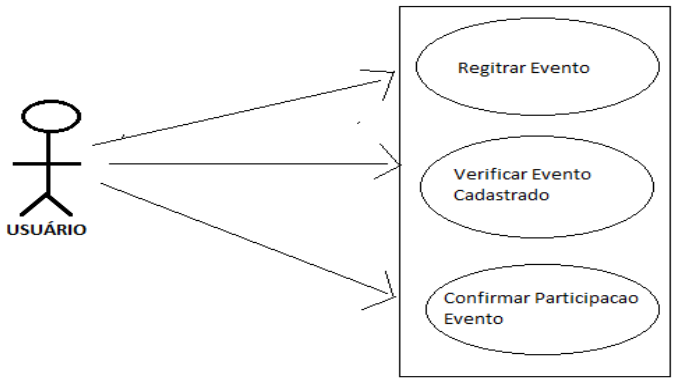



**Figura 9 – Diagrama de Caso de Uso**

### 4.2.2.1 Detalhamento do Diagrama de casos de uso

No Quadro 5, é mostrado o detalhamento para registrar um novo evento utilizando o diagrama de casos de uso.

| |
|---|
| **Caso de uso: Registrar novo evento.**<br>1. [IN] O usuário abrirá aplicativo com usuário facebook.<br>2. [OUT] Sistema localizará o usuário.<br>3. [IN] O usuário irá informar dados sobre o agendamento do evento.<br>4. [OUT] Sistema irá processar as informações.<br>5. [IN] Usuário irá confirmar o agendamento evento.<br>5.1 Variante: Salvar evento.<br>**Variante 5.1: Salvar evento**<br>5.1.1 [OUT] Sistema informa a nome do evento, local, data, hora e participantes.<br>1. "Incluir".<br>2. "Editar". |

**Quadro 5 – Cadastro Novo Evento**

No Quadro 6, são mostradas as variantes do processo, sendo elas incluir, editar e excluir. Onde o usuário faz inserção de dados e o sistema o retorno com as informações.

| |
|---|
| **Variante 1.1: Incluir**<br>1. O usuário informa Nome do evento, data, hora, local e participantes.<br>**Variante 1.2: Editar**<br>1.2.1 Inclui variante 1.2.<br>1.2.2 O usuário informa os novos valores para nome do evento, data, hora, local e participantes. |

**Quadro 6 – Variantes**

No Quadro 7, são mostradas as variantes do processo de logar no sistema *Web*.

| **Variante 1.1: Logar** |
| --- |
| 1. O usuário informa Nome Usuário e Senha. |

**Quadro 7 – Logar Web**

No Quadro 8, é mostrada a variante do processo de realizar filtro do evento na *Web*. Tendo as opções de realizar filtro por Data ou Nome.

| **Variante 1.1: Filtrar por Data** |
| --- |
| 1. O usuário informa data desejada e clica em filtrar. |
| **Variante 2.1: Filtrar por Nome** |
| 2. O usuário informa nome desejado e clica em filtrar. |

**Quadro 8 – Filtrar Evento Web**

No Quadro 9, é mostrada a variante do processo de realizar o cadastro de login do usuário na *Web*.

| **Variante 1.1: Filtrar por Data** |
| --- |
| 1. O usuário informa data desejada e clica em filtrar. |
| **Variante 2.1: Filtrar por Nome** |
| 2. O usuário informa nome desejado e clica em filtrar. |

**Quadro 9 – Cadastro Usuário Web**

No Quadro 10, é mostrada a variante do processo de vincular conta do usuário ao Facebook.

| **Variante 1.1: Logar no Aplicativo** |
| --- |
| 1. Usuário faz login no aplicativo. |
| 2. Ao fazer o login automaticamente está vinculando seus amigos do facebook para sua lista de amigos do aplicativo. |
| **Variante 2.1: Localizar amigos Vinculados** |
| 3. Na tela de incluir eventos terá botão onde usuáro poderá selecionar amigos para evento desejado.. |

**Quadro 10 – Vincular conta do usuário ao Facebook**

### 4.2.3 Diagrama de Classe MVC

A Figura 10 mostra o relacionamento entre as classes dos controles do sistema. Esse exemplo é um dos vários controles do aplicativo, que são todos iguais e seguem o modelo de três camadas. A classe EventoInterface é a classe de interface (*view*). A classe EventoDAO detêm a lógica de negócios da aplicação (*control*), enquanto á classe Evento modelo representa o domínio dos dados (*model).*

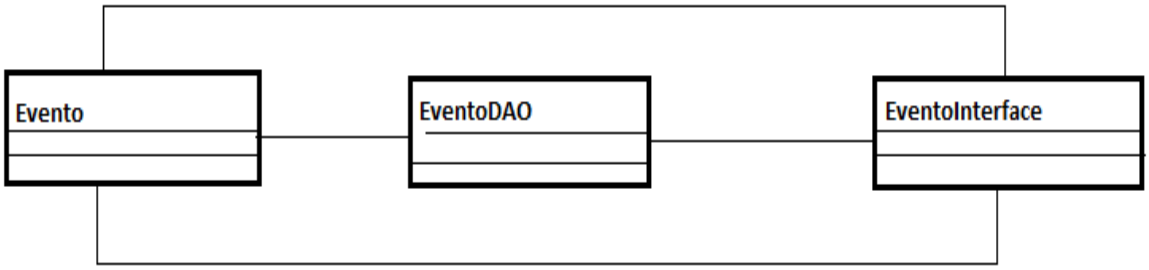



**Figura 10 – Classe dos Controles do Sistema**

## 4.3 APRESENTAÇÕES DO SISTEMA

O diagrama de entidade e relacionamento do banco de dados do software descreve as entidades do sistema e suas relações.

Na Figura 9 é mostrada o diagrama de entidade e relacionamento das tabelas do postgreSQL.

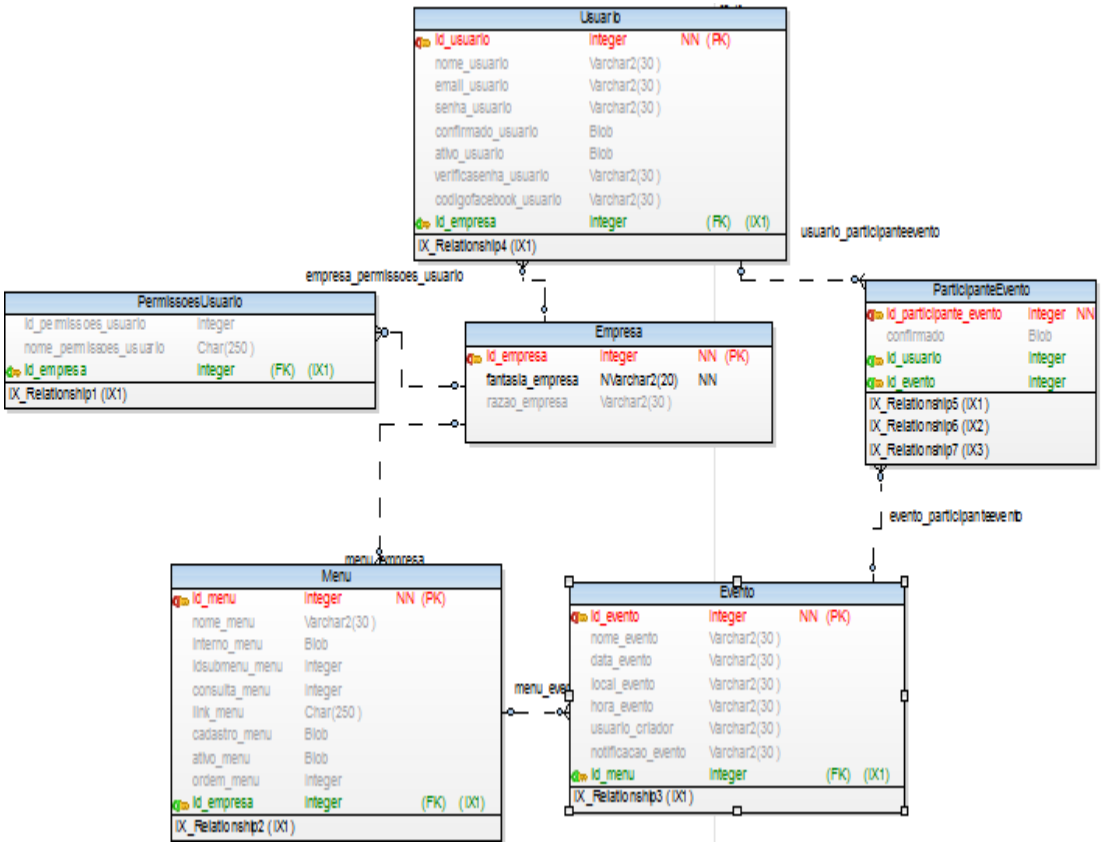



**Figura 11 – Diagrama de Entidade e Relacionamento do Sistema Proposto**

## 4.4 APRESENTAÇÃO DO SISTEMA

O sistema INHDGroups é composto por um aplicativo para dispositivos móves que usam o Android e também uma aplicação web. A seguir são apresentada as telas do aplicativo desenvovido.

### 4.4.1 Tela Inicial

Através da “Tela Inicial”, apresentada na Figura 12, o usuário poderá verificar se tem eventos em que ele foi convidado para participar. Senão tiver evento para participar não irá aparecer na tela inicial nenhum registro.

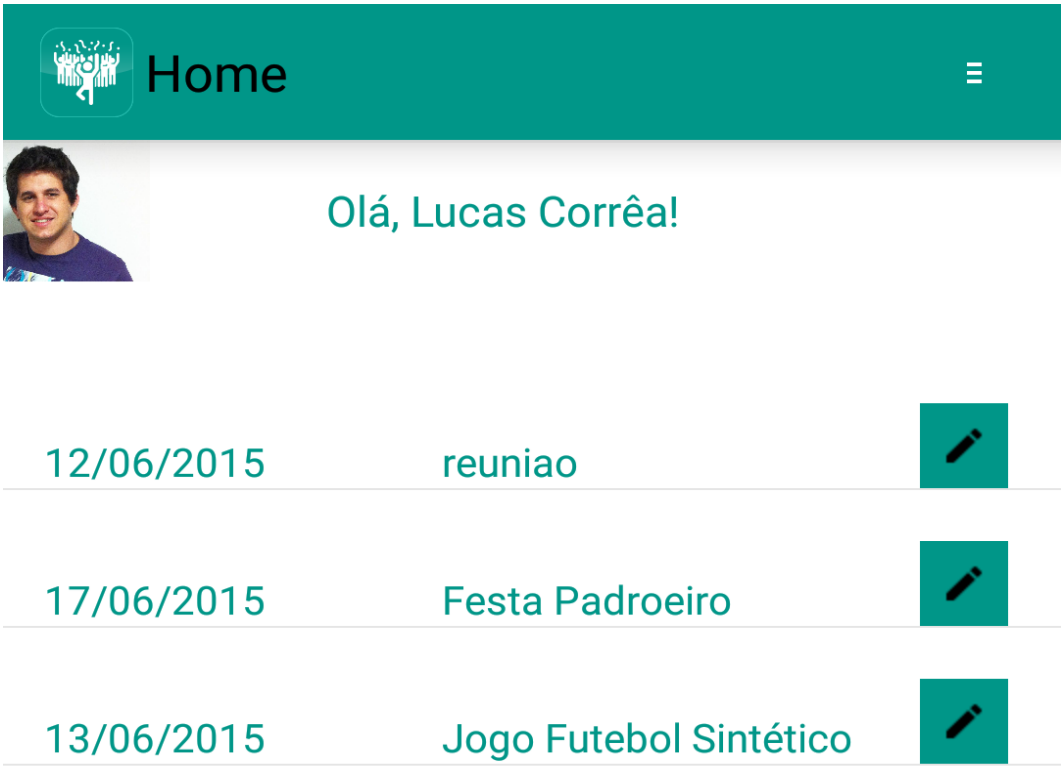



**Figura 12 – Tela Inicial**

### 4.4.2 Evento Convidado

Através da tela “Evento Convidado”, apresentada na Figura 13, o aplicativo permite que o usuário confirme ou não sua presença no evento. Na tela é possível visualizar as informações do evento, como nome evento, data, hora, local, participantes convidados e o administrador do evento. Nessa tela foi implementado restrições, por exemplo: usuário só

poderá selecionar a opção de “Participar” ou ”Recusar” participação do evento, os campos de nome evento, data, hora, local e lista de participantes não poderá ser editado pelo usuário. Somente pelo administrador do evento.

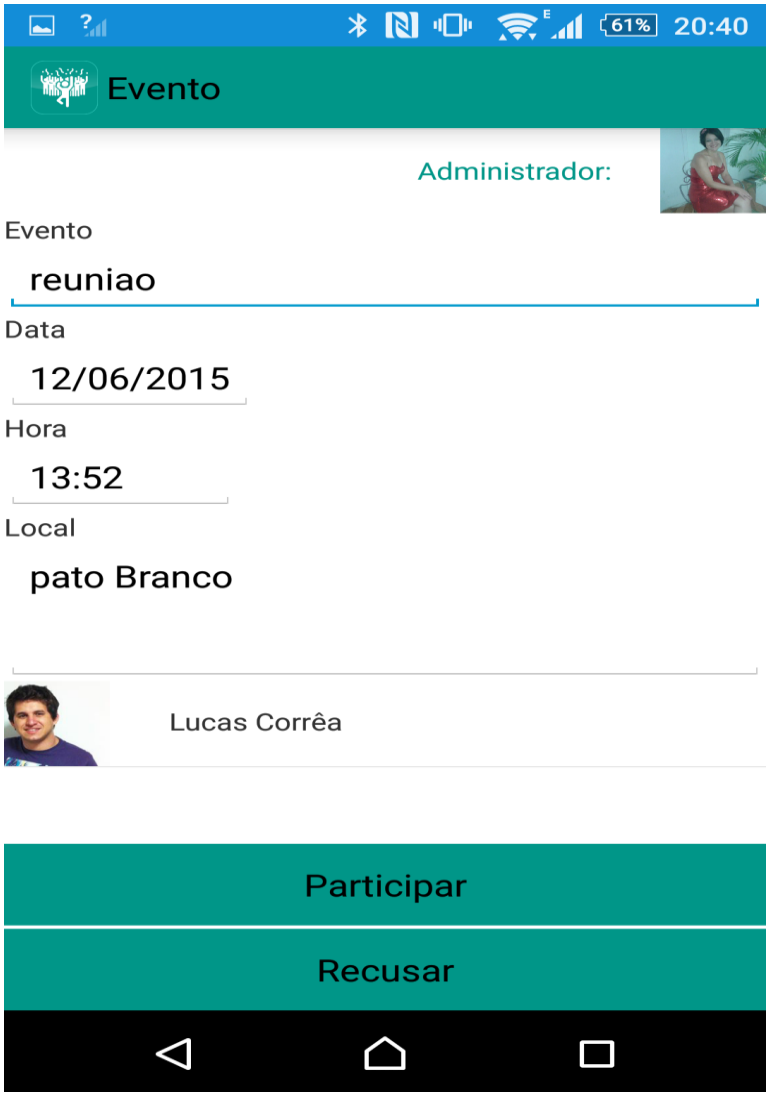



**Figura 13 – Evento Convidado**

### 4.4.3 Meu Evento

Através da tela “Meu Evento” do aplicativo o usuário poderá verificar todos os eventos em que é o administrador. Mostrado na Figura 16.


**Figura 14 – Meus Eventos**

### 4.4.4 Edição de Eventos

O cadastro de eventos na Figura 15 permitirá o usuário administrador editar as informações do evento como nome evento, data, hora, local, lista de participantes. Após fazer a edição do evento usuário clica em “Concluído” e o evento será editado com sucesso, no campo data também a restrição, por exemplo usuário poderá selecionar somente a data atual ou data supoerior. Assim organizando o evento de uma maneira adequada.


**Figura 15 – Edição Evento**

### 4.4.5 Cadastro de Novo Evento

Através do cadastro de novo evento, conforme tela apresentada na Figura 16, o usuário poderá fazer a inclusão das informações do evento, como data, hora, local, participantes. Nessa tela existe o botão que  o usuário ao clicar irá abrir a tela de participantes, a qual é apresentada na Figura 17.

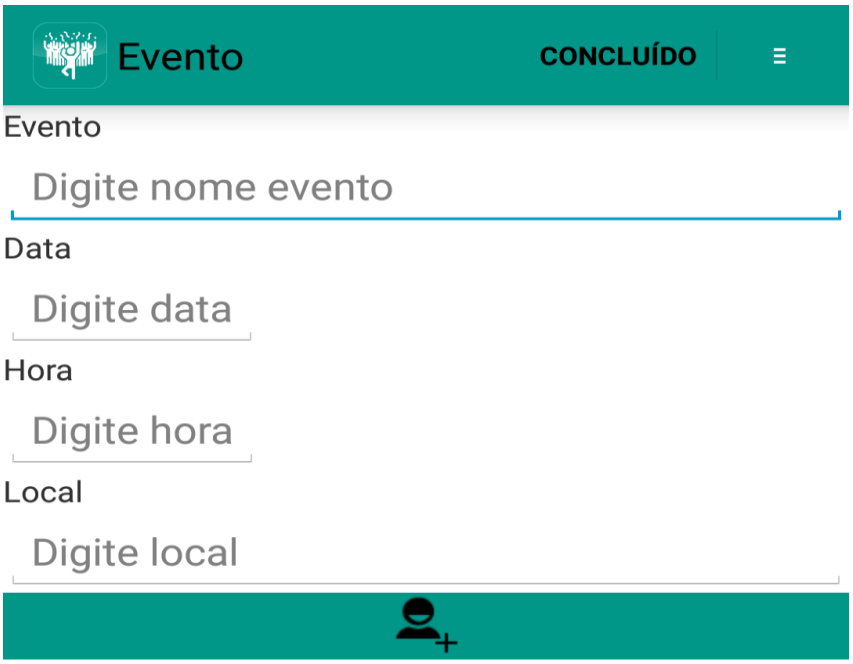



**Figura 16 – Novo Evento**

### 4.4.6 Participantes

Através da tela apresentada na Figura 17 o usuário poderá fazer o filtro dos participantes do evento, selecionando pelo nome, os participantes que irão aparecer na tela serão os amigos do usuário logado no Facebook e que possuem o aplicativo instalado.

Ao selecionar o participante ou os participantes desejados para o evento, usuário poderá fechar a tela normalmente que irá voltar para a tela de evento com as informações do evento e com os participantes selecionados.

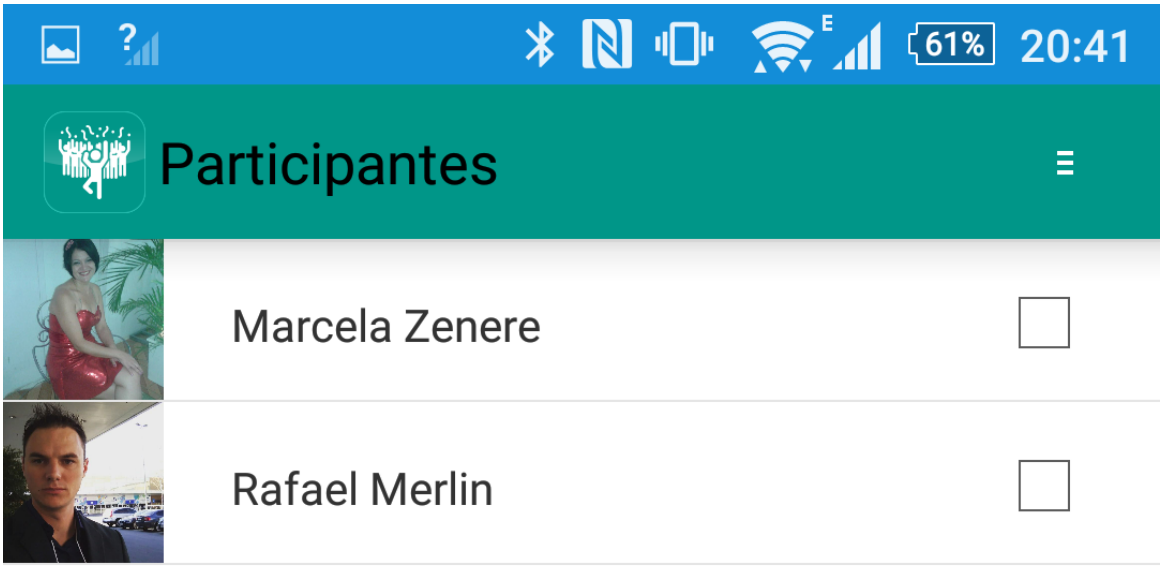



**Figura 17 – Participantes**

### 4.4.7 Tela Inicial Portal Web

Através dessa tela na Figura 18 pode-se visualizar a tela inicial do *Portal Web*, desenvolvido no Java e Primefaces.

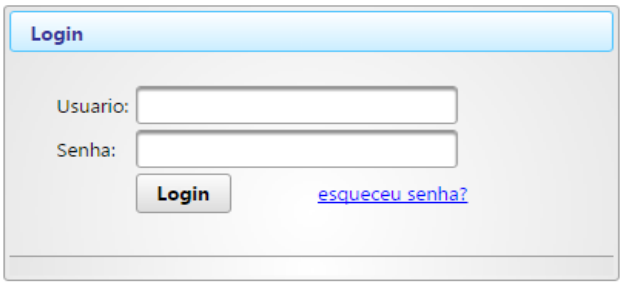



**Figura 18 – Tela Inicial Portal web**

### 4.4.8 Cadastro Evento Portal Web

Através do cadastro de evento no *Portal Web* mostrado na Figura 19, o usuário poderá realizar a inserção de evento pela web, assim ao inserir evento irá aparecer no aplicativo do usuário em "*Meus Eventos*".

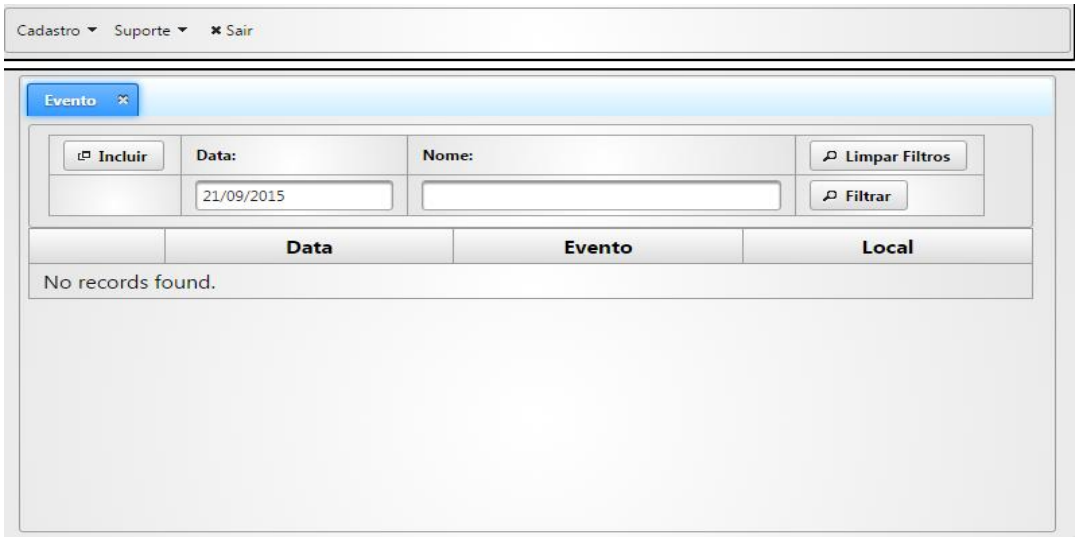



**Figura 19 – Filtro Evento Portal Web**

### 4.4.9 Cadastro Usuário Portal Web

Através do cadastro de usuário, apresentado na Figura 20, o usuário poderá acessar o *Portal Web*. O login de um usuário é o mesmo login do Facebook, pois quando o usuário logar pela primeira vez no aplicativo irá incluir automaticamente o usuário no portal.

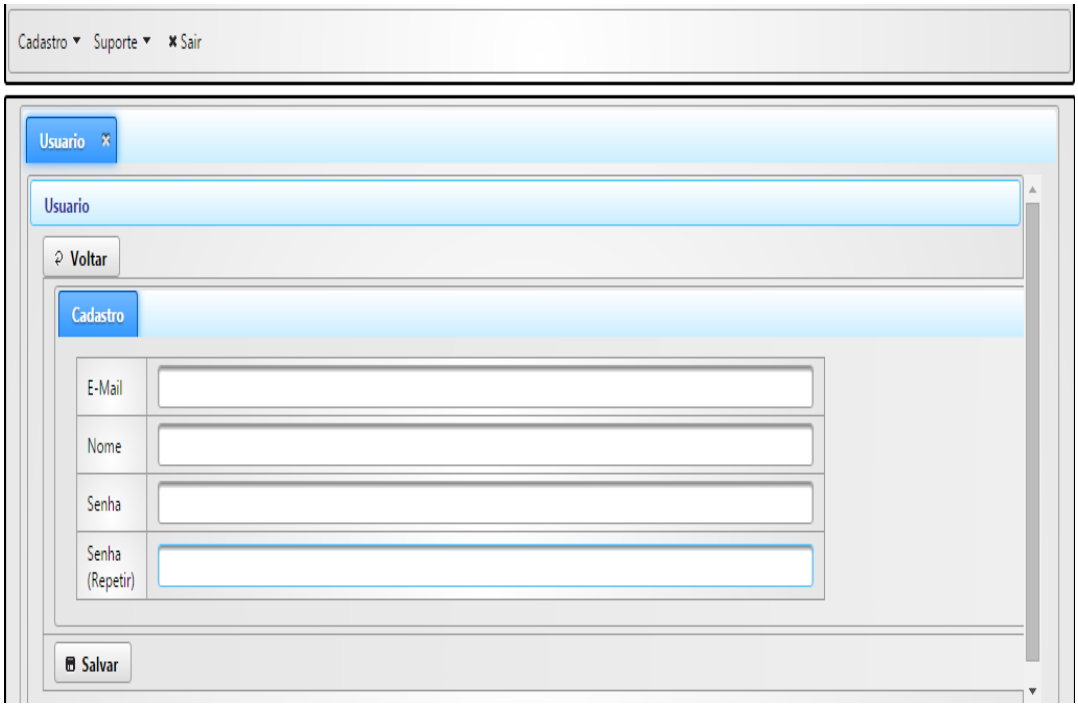
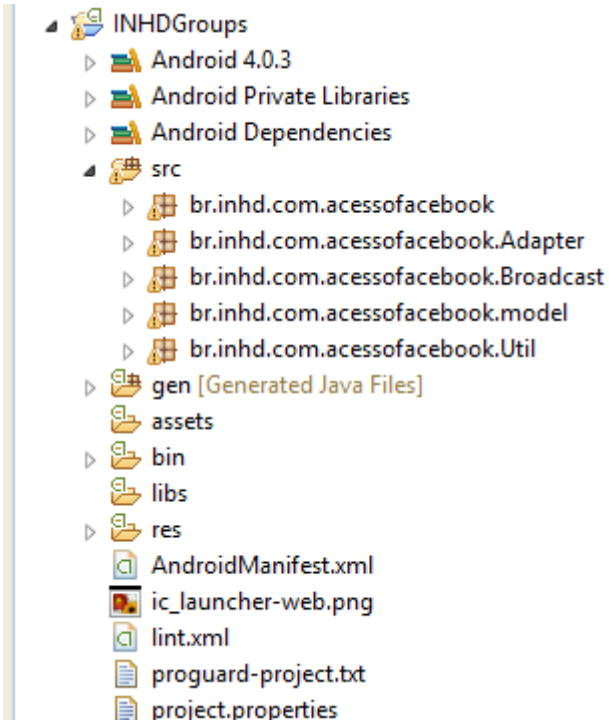



**Figura 20 – Cadastro Usuário Portal Web**

## 4.5 IMPLEMENTAÇÃO DO SISTEMA

Para manter a organização do código fonte do sistema mais simples, o projeto foi desenvolvido utilizando o modelo MVC (*Model – View – Controller*), ou seja, esse modelo divide a aplicação em três camadas: modelo, visualização e controle. O modelo define a visão das entidades do banco de dados, sendo que cada classe que faz parte dessa camada representa uma tabela, a visualização representa a forma através da qual o usuário interagirá com a aplicação, ou seja, através de onde ele inserirá e receberá informações e o controle corresponde à camada que faz a ponte entre a visualização e o modelo, onde estará toda a lógica de negócios da aplicação e de acesso ao banco de dados. Com a utilização desse modelo o desenvolvimento torna-se mais simples e organizado, e a aplicação será mais escalável e extensível.

A Listagem 1 apresenta o *INHDGroups* que mostra a organização do sistema, conforme descrito anteriormente. Na pasta de *util* ficam as classes que as operações de acesso ao portal web e manipulação dos dados, na pasta *model* se encontra as classes do modelo do software, na pasta *broadcast* está a classe que envia notificação para o usuário que ele recebeu um novo evento.




**Listagem 1 – Organização do Sistema**

### 4.5.1 Conexão com o Banco de Dados

Para conectar o sistema com o banco de dados SQLite foi desenvolvida uma classe chamada “BancoDados.java”, a qual têm implementados os recursos para verificar se usuário logou no aplicativo.

Nele foi desenvolvido uma função chamada “VerificaUsuario” onde possui informações referentes à conexão com o banco de dado. A Listagem 2 abaixo mostra como foi desenvolvida a classe “BancoDados”.

```
package br.inhd.com.acessofacebook.Util;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;

public class BancoDados {
private SQLiteDatabase bd;
 public BancoDados(Context context) {
    BancoDadosCore auxBD = new BancoDadosCore(context);
    bd = auxBD.getWritableDatabase();
 }

 public String VerificaUsuario() {
   String usuario_logado = null;
   Cursor     c     =     bd.rawQuery("SELECT     codigo_facebook     FROM
usuariologado",null);
   while (c.moveToNext()) {
   usuario_logado                                                          =
c.getString(c.getColumnIndex("codigo_facebook")).trim();
   }
   return usuario_logado;
 }

   public void onInsertTableUsuarioLogado(String usuario_logado) {
   String  insertDataBase  =  "insert  into  usuariologado  (codigo_facebook)
values ('" + usuario_logado + "');";
   bd.execSQL(insertDataBase);
   }

}
```

**Listagem 1 – Classe Banco de Dados**

Quando é preciso efetuar operações do aplicativo como por exemplo: acesso internet da aplicação é incluída algumas permissões no arquivo AndroidManifest.xml que possuí da aplicação Android. Conforme exemplo apresentado na Listagem 3.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
package="br.inhd.com.acessofacebook"
android:versionCode="1"
android:versionName="1.0" >
<uses-sdk
android:minSdkVersion="15"
android:targetSdkVersion="21" />

<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.WAKE_LOCK" />
<uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED"
/>
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"
/>
<uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED"
/>
```

**Listagem 3 – Android Manifest**

### 4.5.2 Criações de Classes

Buscando realizar o desenvolvimento com base na programação orientada a objetos, todas as entidades do banco de dados do sistema são persistidas através de classes. As classes no Android e Java possuem a extensão ".java".

Todas as classes entidades têm criados os atributos correspondentes aos campos da tabela que representam, no momento da execução das intruções SQL no banco de dados as informações são obtidas dos atributos, vindo a compor tais comandos corretamente.

#### 4.5.2.1 Classes

As classes são responsáveis por implementar o modelo da base de dados, as lógicas de negócio, fazer as conexões entre o banco de dados e o sistema para determinadas tabelas. Na Listagem 4 será mostrado o primeiro método executado na classe "HomeActivity.java" método "OnCreate". Esse método será sempre executado por primeiro em todas as classes de um aplicativo Android.

```
protected void onCreate(Bundle savedInstanceState) {
 super.onCreate(savedInstanceState);
 setContentView(R.layout.activity_home);

 ActionBar ab = getActionBar();

 ab.setBackgroundDrawable(getResources().getDrawable(R.drawable.bg));
 uiHelper = new UiLifecycleHelper(this, callback);
 uiHelper.onCreate(savedInstanceState);

 util = new Util(this);
 util.setServidor("http://servidor.eagente.com.br:8080/");
 StrictMode.ThreadPolicy policy =
 new StrictMode.ThreadPolicy.Builder().permitAll().build();
 StrictMode.setThreadPolicy(policy);
 GeraLogTask geraLogTask =  new GeraLogTask();
 geraLogTask.executeOnExecutor(sDefaultExecutor, ".");
 tvInformacao = (TextView) findViewById(R.id.tvInformacao);
}
```

**Listagem 4 – Método OnCreate**

#### 4.5.2.2 Classe Lógica de Negócio

Toda a classe que faz a lógica de negócio irá interagir com a classe modelo e a classe de *interface.* Porém são classes que não têm contato direto com o banco de dados, mas são responsáveis por buscar as informações nas outras classes, como mostra o código apresentado na Listagem 5 abaixo o método "getJSONArrayFromUrl" responsável por fazer a lógica de negócio que envia informações para *URL*.

```
public JSONArray getJSONArrayFromUrl(String url, List<BasicNameValuePair>
params) {

InputStream is = null;
 try {

  DefaultHttpClient httpClient = new DefaultHttpClient();
  HttpPost httpPost = new HttpPost(url);
  httpPost.setEntity(new UrlEncodedFormEntity(params, HTTP.UTF_8));
  HttpResponse httpResponse = httpClient.execute(httpPost);
  HttpEntity httpEntity = httpResponse.getEntity();

  is = httpEntity.getContent();

 } catch (UnsupportedEncodingException e) {
  e.printStackTrace();
 } catch (ClientProtocolException e) {
  e.printStackTrace();
 } catch (IOException e) {
  e.printStackTrace();
 }
  String json = null;
 try {
   BufferedReader reader = new BufferedReader(new InputStreamReader(
   is, "iso-8859-1"), 8);

   StringBuilder sb = new StringBuilder();
   String line = null;

   while ((line = reader.readLine()) != null) {
     sb.append(line + "n");
   }
   is.close();
   json = sb.toString();
  } catch (Exception e) {
   Log.e("Buffer Error", "Error converting result " + e.toString());
  }

 JSONArray jObj = null;
 // retorna JSONArray
 try {
  jObj = new JSONArray(json);
 } catch (JSONException e) {
  Log.e("JSON Parser", "Error parsing data " + e.toString());
 }
 // return JSON String
 return jObj;

}
```

**Listagem 5 – Método getJSONArrayFromUrl**

#### 4.5.2.3 Classe Interface

As classes de interface fazem a camada de visualização com o usuário, onde será visualizado a imagem, inserido as informações e acionado a ação dos botões.

Na Listagem 6, pode ser visto o código fonte do layout onde mostra para o usuário o nome e a data do evento, logo ao lado um botão em que o usuário poderá editar o evento.

```
<?xml version="1.0" encoding="utf-8"?>
 <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:background="@color/corfundo"
   xmlns:facebook="http://schemas.android.com/apk/res-auto"
   android:orientation="horizontal" >

 <RelativeLayout
   android:layout_width="match_parent"
   android:layout_height="match_parent" >

 <TextView
   android:id="@+id/nome"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:layout_alignParentTop="true"
   android:layout_marginLeft="30dp"
   android:layout_marginTop="20dp"
   android:layout_toRightOf="@+id/data"
   android:textColor="@color/corfundobotao"
   android:text="nome" />

 <TextView
   android:id="@+id/data"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:layout_alignBaseline="@+id/nome"
   android:layout_alignBottom="@+id/nome"
   android:layout_alignParentLeft="true"
   android:textColor="@color/corfundobotao"
   android:layout_marginLeft="14dp"
   android:text="data" />

 <ImageButton
   android:id="@+id/btnEditarEvento"
   android:layout_width="30dp"
   android:layout_height="30dp"
   android:layout_alignBottom="@+id/nome"
   android:layout_alignParentRight="true"
   android:layout_marginRight="17dp"
   android:background="@color/corfundobotao"
   android:src="@drawable/editar" />
 </RelativeLayout>

</LinearLayout>
```

**Listagem 6 – Classe Interface**

# 5 CONCLUSÃO

Os objetivos iniciais propostos no início do trabalho foram atingidos com sucesso, considerando que para o desenvolvimento deste software foi necessária muita pesquisa e um longo caminho de aprendizado, pois o desenvolvimento envolve conhecimentos além da programação: é necessário moldar interfaces e o projeto das telas.

Além disso, sistema deve sempre procurar fornecer o melhor desempenho possível. No decorrer do projeto foram encontradas algumas barreiras no desenvolvimento, algo que já era esperado no planejamento do projeto. Como por exemplo: desenvolvimento Android, linguagem que até então não desenvolvia, análise do projeto e designer das telas.

Trabalhar com a orientação a objetos é sempre surpreendente, pois os resultados são alcançados com menos esforço, e todo o código fica mais organizado e com melhor entendimento.

A linguagem de desenvolvimento Java e Android se mostram uma ótima opção de desenvolvimento, sendo prático e de fácil utilização.

A integração com o Facebook foi muito útil nos primeiros passos do projeto, pois nesta integração foi visto com é realizar desenvolvimento usando plataforma ótima e que dispõe de ótimos recursos.

O PostgreSQL é um ótimo gerenciador de banco de dados, é rápido e seguro, além de ser um gerenciador simples no qual qualquer banco de dados pode ser criado e gerenciado.

Assim sendo que o resultado foi esperado como resultado final um aplicativo para a web para o gerenciamento de eventos permitindo ao promotor de eventos maior abrangência na publicação do seu evento e ao ao usuário maior facilidade e controle dos eventos que pretende participar.

Enfim, a experiência do desenvolvimento de um aplicativo com portal web difere muito dos conceitos vistos ao decorrer do curso, são novos conhecimentos, novas experiências que serão úteis ao longo da vida.

**REFERÊNCIAS**

ADRIEL. http http://adrielcafe.com/cafelog/hibernate/45-introducao-ao-hibernate acessado em 05/06/2015.

ANDROID. https://www.infinum.co/the-capsized-eight/articles/android-studio-vs-eclipse-1-0, acessado em 20/05/2015.

API GRAPH**.** http://www.9lessons.info/2011/01/facebook-graph-api-connect-with-php-and.html acessado em 25/05/2015.

CAELUM. http://www.caelum.com.br/apostila-java-web/servlets/#5-2-servlets acessado em 22/09/2015.

FACEBOOK PLATFORM**.** http://imasters.com.br/artigo/20274/apis/explorando-funcionalidades-das-apis-do-facebook/, acessado em 25/05/2015.

FAGUNDES, ROBSON. http://robsonfagundes.blogspot.com.br/2011/02/introducao-ao-primefaces-com-jfs20.html, acessado em 02/06/2015.

IDCBRASIL. http://g1.globo.com/tecnologia/noticia/2015/04/venda-de-smartphones-sobe-55-no-brasil-em-2014-diz-idc.html acessado em 10/06/2015.

JAVA**.** https://lgformacao.files.wordpress.com/2011/06/52_java.pdf, acessado em 02/06/2015.

JSF, http://www.caelum.com.br/apostila-java-testes-jsf-web-services-design-patterns/introducao-ao-jsf-e-primefaces/#7-2-caracteristicas-do-jsf**,** acessado em 02/06/2015.

LAUDON, Kennet C; LAUDON J.P. **Sistemas de Informação Gerenciais: Administrando a Empresa Digital**. 5ª Edição. São Paulo: Pearson Prentice Hall, 2004.

O'BRIEN, James A. **Sistemas de Informação: e as Decisões Gerenciais na Era da Internet.** Editora Saraiva 2002.

OGLIARI, Ricardo da Silva; BRITO, Robison Cris. **Android: Do Básico ao Avançado**. Rio de Janeiro: Editora Ciência Moderna Ltda, 2014.

PRIMEFACES. http://www.primefaces.org/showcase/ui/input/calendar.xhtml acessado em 02/06/2015.

SERVLET. http://www.devmedia.com.br/introducao-a-servlets-em-java/25285, acessado em 21/05/2015.

SQL MAGAZINE. http://www.devmedia.com.br/artigo-sql-magazine-6-postgresql-introducao-e-conceitos/7185, acessado em 05/06/2015.