UNISA – UNIVERSIDADE DE SANTO AMARO
CURSO BACHARELADO EM SISTEMAS DE INFORMAÇÃO

CHRISTIAN SOUSA
LÚCIO LIMA
MARCELO FELIPE

APLICATIVO ANDROID BOHEMIAN

São Paulo
2013

CHRISTIAN SOUSA
LÚCIO LIMA
MARCELO FELIPE

APLICATIVO ANDROID BOHEMIAN

Monografia apresentada para obtenção do título de Bacharel em Sistemas de Informação ao Curso de Sistemas de Informação da Universidade de Santo Amaro – UNISA sob orientação da (Eugênio Akihiro Nassu).

São Paulo
2013

CHRISTIAN SOUSA
LÚCIO LIMA
MARCELO FELIPE

APLICATIVO ANDROID BOHEMIAN

Monografia apresentada para obtenção do título de Bacharel em _____ao Curso de _____da Universidade de Santo Amaro – UNISA sob orientação da Área de Concentração_____.

Data de Aprovação: ____/____/____

BANCA EXAMINADORA

_____

_____
(nome do orientador e Titulação)

_____

_____
(nome do professor e Titulação)

CONCEITO FINAL: _____

DEDICATÓRIA

Trabalho dedicado a todos os nossos familiares, esposas e filhos, que acreditam em nossos potenciais, nos motivando a ir sempre além de todas as expectativas, e nos tornarmos pessoas realizadas, informadas e sempre dispostas a fazer o melhor, buscando sempre os caminhos corretos para o nosso desenvolvimento que também será o melhor para eles, sempre seguindo os nossos corações orientados pela força de Deus.

AGRADECIMENTOS

A todos os professores, que nos apoiaram para que hoje possamos apresentar esse trabalho muito importante, feito através de muitas pesquisas, estudos e a ajuda diretamente deles tirando as nossas dúvidas, aconselhando, mostrando sempre os melhores caminhos a serem seguidos para o nosso desenvolvimento pessoal e acadêmico.

“A maior invenção do mundo não é a minha tecnologia!

É a morte! Pois através dela,

o velho sempre dará lugar para o novo!”

Steve Jobs

RESUMO

Com o grande avanço da tecnologia, a cada dia temos novidades nesse vasto mercado, uma das novidades são os aplicativos para dispositivos móveis, que a cada dia ganham mais funcionalidades úteis para o usuário, aplicativos são desenvolvidos e disponibilizados a todo o momento, e para todas as áreas de conhecimento, como saúde, diversão, informação, e muito mais. Com base nessa situação desenvolvemos um protótipo de aplicativo na plataforma Android com o nome Bohemian, a fim de ajudar usuários que tenham necessidades específicas, nosso aplicativo simula uma comanda, mas em tempo real, você não faz o pedido diretamente ao garçom, você usa o Bohemian para fazer pedido, o garçom apenas serve, evitando assim que seja marcado em sua comanda um item que você não solicitou, para quem frequenta bares é muito comum acontecer e pode causar constrangimentos. Foi utilizada a tecnologia Java e Android para desenvolver nosso aplicativo que é simples, porém muito objetivo na tarefa que executa.

Palavras-chaves: Android, Bohemian, Java.

ABSTRACT

With breakthrough technology, we have news every day from this vast market, one of those are the mobile device applications, that every day earn more useful features for the uses, applications are developed and made available every moment, and for all areas of knowledge, like health, entertainment, information, and a lot more. Based on this situation we developed an Android Plataform's prototype application called Bohemian, to help users that have specific needs, our application simulates an order, but in real time, you don't do the order directly to the waiter, you use the Bohemian to make that order, and the waiter only deliveries that request, avoiding some item that you did not asked for it, to be marked on your order, for those who uses to frequent bars it is specially likely to occur and can cause embarrassment . Was used Java and Android's technology to develop our application that it is simple, but very purpose on the tasks that executes.

Keywords: Android, Bohemian, Java.

LISTA DE FIGURAS

Sumário

1 INTRODUÇÃO

Nesse trabalho temos o objetivo de propor uma solução para um problema que apesar de simples, causa constrangimentos em frequentadores de bares e pequenos restaurantes, algumas vezes em nossas comandas itens que não consumimos, pensando em como resolver esse problema foi desenvolvido um o protótipo de aplicativo que funciona como uma comanda o Bohemian, onde o cliente é quem envia os seus pedidos para o garçom ou atendente, abordaremos um tipo de tecnologia a que cada dia cresce mais, as plataformas Mobile, especificamente o sistema operacional da Google, o Android, que hoje em dia é utilizada por 75% dos smartphones no mundo, e por isso está em evidência no mercado há anos. Levando em consideração a grande evidência de tal tecnologia, e através de muito empenho, dedicação e estudos sobre a linguagem, foi desenvolvido um aplicativo protótipo utilizando Android/Java com o sugestivo nome de Bohemian.

Bohemian é uma comanda que permite a iteração em tempo real entre o garçom e o cliente, assim que o cliente faz o pedido, o garçom pode listar a informação do pedido identificado pela mesa e os produtos solicitados, assim o não pagará pelo o que não pediu, já que é ele que controla o pedido. Já existem aplicativos similares no mercado, o Bohemian é relativamente simples, mas bem funcional e com boas práticas de programação e escolhemos como Design Patterns o (Dao) para melhor controle do banco de dados.

1.1 ORGANIZAÇÃO DO TRABALHO

O trabalho está dividido em doze partes, capítulo 1 Introdução, capítulo 2 Revisão bibliográfica, capítulo 3 História da programação, capítulo 4 As vantagens do Android, capítulo 5 Caso de uso, capítulo 6 Diagrama de classes, capítulo 7 Análise de requisitos, capítulo 8 Funcionamento do aplicativo, capítulo 9 Desenvolvimento, capítulo 10 Conclusão, capítulo 11 Referências bibliográficas, capítulo 12 Codificação do software.

Antes de começarmos o trabalho, é importante falar um pouco da história da programação, um breve resumo de como tudo começou até os dias de hoje.

2 REVISÃO BIBLIOGRÁFICA

2.1 ANDROID

O mercado de telefonia móvel vem crescendo cada vez mais nos últimos anos. Estudos apontam que mais ou menos 3 bilhões de pessoas tem um aparelho celular, existem hoje mais celulares do que pessoas no Brasil por exemplo, o que corresponde a metade da população do planeta.

A telefonia corporativa também cresceu em ritmo acelerado, muitas empresas buscam aplicações que possam ajudar a agilizar seus negócios. Como o objetivo principal das organizações é obter lucro os smartphones com sua mobilidade vieram pra ficar.

Atualmente, vários serviços podem ser acessados diretamente de um smartphone, bancos, por exemplo, disponibilizam aplicativos nos quais é possível efetuar transferências, pagar contas, consultar saldo e extrato, entre muitas outras.

O S.O Android do Google veio para preencher esse espaço, é uma nova plataforma de desenvolvimento para aplicativos móveis, baseada em Linux, que já conta com algumas aplicações instaladas e, um ambiente de desenvolvimento flexível, ousado e poderoso.

Para desenvolver a aplicação mostrada nesse trabalho foi utilizado a linguagem Java,o IDE Eclipse e o SDK do Android.

3 HISTÓRIA DA PROGRAMAÇÃO

Quem é Ada Augusta Byron King? - Será que você a conhece? E Ada Lovelace? - Provavelmente não, mas saiba que muitos a consideram a mãe da programação, graças a um projeto em que Ada trabalhou, ela escreveu um programa para ser utilizado na máquina analítica de Charles Babbage, tal máquina é considerada o marco inicial da programação de computadores, o algoritmo que Ada escreveu calculava a sequência de Bernoulli, também conhecida como a Lei dos Grandes Números. Ada na verdade não implementou o algoritmo ela apenas o traduziu e o estudou, também conheceu a fundo a máquina de Babagge. Com esse estudo em mãos, Ada foi capaz de escrever um artigo com o passo a passo de como calcular os números de Bernoulli, esse seu artigo é considerado por muitos o primeiro programa de computador, mas pesquisadores apontam possíveis pais da programação bem antes de Ada, Joseph-Marie Jacquard, por exemplo, um francês que inventou o Tear Mecânico é um deles, mas Ada foi a primeira a registrar academicamente e conceber um algoritmo, além de tratar implantar realmente em uma máquina de fato.

3.1 HISTÓRIA DO JAVA

Java é uma linguagem de programação orientada a objetos que foi criada em 1991, pela empresa Sun Microsystems, os programadores da Sun trabalhavam no desenvolvimento de um programa para set-top Box. No início do projeto escolheram a linguagem C/C++ para o projeto, mas logo perceberam que com essa linguagem, seu sistema não poderia ser portátil.

Com base no que precisavam, decidiram tirar alguns recursos da plataforma e acabaram quase que acidentalmente criando uma nova linguagem de programação, que tinha como nome ‘OaK’ (Carvalho em português) em homenagem ao desenvolvedor que criou os códigos de baixo de uma árvore de carvalho, mesmo sendo uma grande novidade no mercado, a linguagem não foi aceita.

Mas cinco anos depois com o grande crescimento da internet, a ‘Oak’ passou a ser a melhor opção para a web, já que a internet precisava de uma linguagem de programação portátil e que pudesse ser usada para escrever programas que funcionassem em diferentes plataformas, o que é a principal característica do Java.

Olhando para o futuro e com a aceitação imediata da linguagem, a empresa Sun optou no mesmo ano em renomear a linguagem para ‘JAVA’ após mais algumas alterações na plataforma, foi relançada no mercado. Os desenvolvedores de todo o mundo descobriram o potencial da tecnologia Java, que virou febre e entre os programadores, e hoje em dia estima se que aproximadamente 800 milhões de computadores e 3,5 bilhões de cartões magnéticos inteligentes utilizam Java.

3.2 HISTÓRIA DO ANDROID

3.3 O QUE É ANDROID?

Segundo o site www.areadecobertura.com,

> “O Android é um Sistema Operacional de Código aberto desenvolvido pela Open Handset Alliance, ou como define o site oficial do Android é a primeira plataforma totalmente personalizável, gratuita e de código aberto. Deixando em panos limpos, dizer que o Android é um Sistema Operacional, é dizer que ele está para os telefones assim como o Windows está para os computadores. Assim, criado o Android, fabricantes estão construindo inúmeros telefones capazes de funcionar com ele. É importante afirmar que sabendo todos que o Android é um sistema operacional, é muito errado dizer "Celular Android", mas tudo bem vai, vamos deixar o mercado falar assim para ficar mais fácil.”

Quando se fala em Android, todos associam o nome imediatamente ao Google, mas a história do Android começou muito antes, em outubro de 2003, na cidade de Palo Alto na Califórnia, quando Andy Rubin , Rich Miner, Nick Sears e Chris White fundaram a Android Inc. O foco era em desenvolvimento de sistemas operacionais para celulares, baseados no sistema operacional do núcleo Linux para dispositivos móveis, o desenvolvimento tinha por objetivo ser

uma plataforma flexível, aberta e de fácil migração para outros fabricantes, todos os projetos eram secretos, o primeiro sistema Android criado por eles foi o Android 1.5 chamado de Cupcake.

Só em outubro de 2005 a Google comprou a Android, fazendo uma subsidiária e investiu pesado em desenvolvimento para que a linguagem fosse realmente bem difundida e que posteriormente se tornasse a principal tecnologia nos dispositivos móveis, os fundadores Andy Rubin, Rich Miner e Chris White foram mantidos na equipe.

Em dezembro de 2006 saiam as primeiras notícias no canal (BBC), e (The Wall Street Journal), ambos publicaram que a Google estava entrando para o mercado de telefones móveis, tecnologia que era desenvolvida com uma parceria da empresa de software Sun Corp.

O primeiro telefone móvel comercializado e disponível a rodar com o sistema Android foi o HTC Dream com a versão 1.5 na data de 22 de outubro de 2008. Segundo o Google, mais de um milhão e 300 mil aparelhos com este sistema operacional são ativados todos os dias utilizados por fabricantes de diversas empresas como: HTC, Samsung, Sony, Motorola, LG e recentemente a Positivo Informática. Atualmente a versão do Android está na 4.2.2 chamada Jelly Bean.

4 VANTAGENS DO ANDROID

4.1 ALGUMAS VANTAGENS DE SE USA ANDROID

Fazendo algumas comparações com ao principal concorrente (Iphone) segundo o site http://www.bspcn.com, temos uma lista com dez vantagens do Android sobre o Iphone, são elas:

1-Permite executar mais de um programa ao mesmo tempo, seja nativo ou oriundo do Android Market, enquanto o da Apple só permite aplicações do próprio telefone rodando ao mesmo tempo.

2-O uso dos widgets (atalhos rápidos para serviços) é bem dinâmico no Android, as informações que eles apresentam estão sempre visíveis e ao alcance, assim fica mais fácil checar o email, as redes sociais e o tempo por exemplo.

3- Como é baseado em ferramentas de código aberto, o Android Market, embora ainda menor que o App Market do Iphone, cresce muito e traz muitas alternativas gratuitas.

4- As notificações do Android são mais organizadas, e contam com uma barra com ícones diferenciados para melhor identificação dos avisos, além do sistema permitem aos programadores a inserção de notificações diretamente na área de trabalho inicial do telefone.

5- O sistema do Google está em várias marcas de celulares diferentes - Motorola,Samsung, HTC, Samsung, LG, enquanto o iPhone, é iPhone.

6- Embora aqui no Brasil o Iphone esteja em diferentes operadoras, nos EUA ele exige contrato com a AT&T, enquanto o Android se espalha por Sprint, Verizon, T-Mobile.

7- O Android permite programação mais personalizada, inclusive portando interfaces customizadas de um aparelho a outro.

8- A mudança de configuração (por exemplo, mudar a conexão de WiFi para 3G) do sistema do Google é mais rápida.

9- A integração com serviços do Google e redes sociais é a melhor parte do Android, que permite uma conexão rápida e transparente com os sites.

10- Os preços ao menos nos EUA são mais convidativos, lá, as operadoras já oferecem smartphones com Android gratuitamente se ligados a um plano de dois anos. No Brasil temos planos semelhantes, porém com preços nada agradáveis.

Essas são vantagens citadas por usuários e desenvolvedores de Android, podem até ser tendenciosas devido aos usuários serem os principais envolvidos nas pesquisas, porém temos que ponderar, e reconhecer que o ótimo IOS da Apple arranca suspiros de seus fãs e usuários e que a questão de qual é o melhor é bem relativa, depende de muita coisa para ser avaliada, e esse tipo de pesquisa só tende a deixar as concorrências bem saudáveis, e quem ganha com isso? Os simples usuários.

7 CÓDIGO ABERTO (OPEN SOURCE)

Após seu lançamento, ao Android se tornou a primeira plataforma móvel a ser livre e ter seu código totalmente aberto (open-source), o que acabou se tornando um grande diferencial diante dos seus principais concorrentes. O sistema passa periodicamente por melhorias, e conta com a contribuição de milhares de desenvolvedores ao redor do mundo para ajudar no desenvolvimento da plataforma.

Essa é a maior vantagem para os fabricantes, pois eles podem utilizar o S.O Android em seus dispositivos sem pagar absolutamente nada por isso. Outra vantagem é uma licença chamada Apache Software Foundation (ASF) permite que o código-fonte possa ser alterado e assim criar dispositivos personalizados sem a necessidade de divulgar as alterações.

Todo o código fonte do Android pode ser baixado no nesse endereço: http://source.android.com/index.html.

5 CASO DE USO BOHEMIAN (USE CASE)

O caso de uso é uma técnica utilizada para demonstrar o que o sistema fará após ser projetado. Normalmente é feito com base na abordagem junto ao cliente, mas no caso do Bohemian os desenvolvedores são os próprios clientes, as especificações foram feitas conforme as necessidades levantadas e apontadas através de situações presenciadas. O comportamento do sistema é mostrado amplamente nesse caso de uso mostraremos o que acontece quando acontece a iteração do usuário com o software.

5.1 PRINCIPAIS OBJETIVOS DO CASO DE USO

-Definir detalhadamente os requisitos do sistema;

-Fornecer informações do que faz o sistema;

-Descobrir os requisitos funcionais, classes e como opera o sistema.

Componentes do caso de Uso.

Ator, que solicita ações e recebe as reações do software.

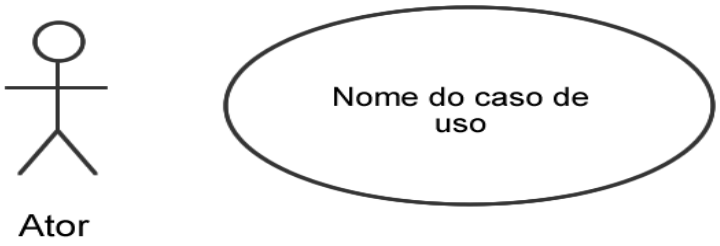
Sistema, o software que será modelado.

Caso de Uso

É um conjunto específico de ações feitas pelo sistema que contém um resultado observável. Caso de uso é representado por uma elipse, com o nome do caso de uso dentro ou abaixo. Se há limites do sistema no diagrama, o caso de uso deve ficar dentro.

Será utilizado o Unified Method Language (UML) para representar o caso de uso.

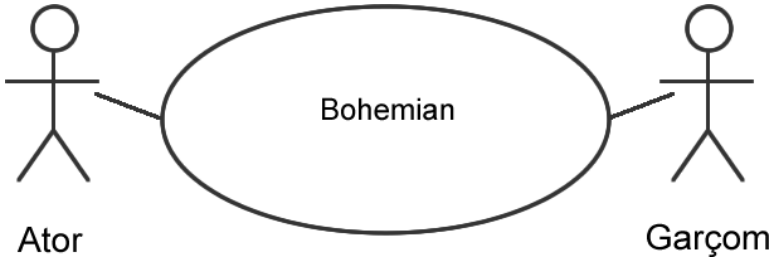
Figura 1- Use Case



Fonte: Elaborada pelo próprio autor

Para o projeto foi construído o caso de uso com dois atores, um faz o papel do cliente e o outro faz o garçom do estabelecimento e ambos interagem como o software.

Figura 2- Use case Bohemian

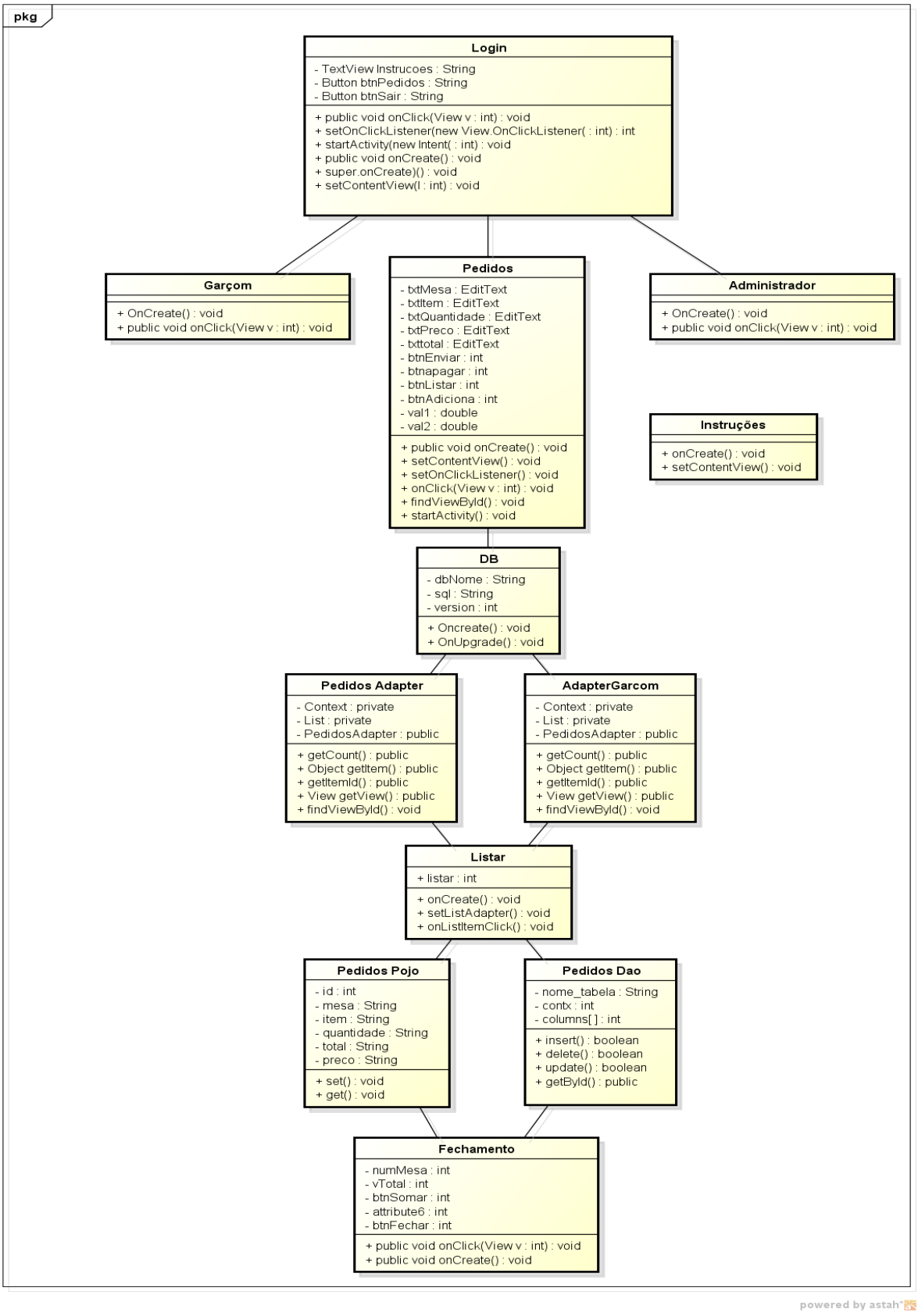


Fonte: Elaborada pelo próprio autor

6 DIAGRAMA DE CLASSES

O diagrama de classe exibe toda a estrutura e relacionamento entre as classes, é uma modelagem muito útil e serve de base para a construção dos diagramas como o de sequência, o de estado e outros. Nesse trabalho temos apenas o de uso, descrito acima e o de classes que será apresentado abaixo:

Figura 3 Diagrama de Classes



Fonte: Print da tela do programa Astah

7 ANÁLISE DE REQUISITOS

A análise de requisitos é a chave para o sucesso de qualquer que seja o projeto, nessa etapa do projeto é feita toda a análise de todas as necessidades do cliente, esse levantamento é essencial para que se possa definir definitivamente como será e como funcionará o software.

Os requisitos do projeto se dividem em partes:

7.1 REQUISITOS DO PROJETO

Levantamento de requisitos referentes ao projeto, essa etapa é feita logo no início projeto, mas deve ser gerenciada até o fim ela é determinante para o sucesso de um bom projeto.

7.2 REQUISITOS FUNCIONAIS

Os requisitos funcionais como diz o próprio nome, especifica com é o funcionamento do software, ele torna a vida dos envolvidos na construção do sistema mais fácil porque detalha todo o funcionamento do software.

7.3 REQUISITOS NÃO FUNCIONAIS

Contempla tudo o que não faz o seu software, todas as restrições que ele tem juntamente com as suas propriedades.

7.4 REQUISITOS DO PRODUTO

Requisitos do produto descrevem propriedades de um software ou produto é uma das várias maneiras de conseguir satisfazer um conjunto de requisitos de negócio que foi proposto. Abaixo descreveremos todos os requisitos acima citados dentro do nosso projeto.

7.5 ANÁLISE DOS REQUISITOS BOHEMIAN

Como já foi citado, o grupo é o idealizador do projeto, por isso desenvolvemos a análise dos requisitos do projeto, uma vez que a necessidade de criação desse software foi levantada pelo grupo, assim toda a funcionalidade e necessidade foram feitas de acordo com o que julgamos que seria válido.

7.6 REQUISITOS DO PROJETO BOHEMIAN

Os requisitos do projeto foram levantados e discutidos por todos os integrantes do grupo, veio de uma necessidade em comum, por isso foi desenvolvido esse software para que o cliente possa ter o real controle do que é pedido em sua comanda, e pagar apenas pelo que realmente solicitou, essa foi a principal necessidade levantada para o projeto.

7.7 REQUISITOS FUNCIONAIS DO BOHEMIAN

Após ser instalado no dispositivo móvel, o cliente faz os seus pedidos, o garçom lista os pedidos e serve o cliente em sua mesa, o cliente continua a fazer os seus pedidos até ele clicar em qualquer pedido e fechar a sua conta, feito isso ele vai até o caixa e efetua o pagamento para a atendente.

7.8 REQUISITOS NÃO FUNCIONAIS DO BOHEMIAN

Quanto ao uso do nosso software, é necessário ter o sistema Android instalado no dispositivo móvel, o Bohemian não funciona em outro sistema operacional, é exclusivamente projetado para Android.

7.9 REQUISITOS DO PRODUTO BOHEMIAN

O Bohemian é um software simples, com apenas cinco telas, bem intuitivas, na principal o cliente escolhe fazer os pedidos, na tela de pedidos o cliente digita as quantidades e os itens que deseja consumir, na tela de

listagem o cliente verifica o que pediu depois ele fecha os pedidos e verifica qual o valor da sua conta.

É como se o usuário estivesse fazendo o pedido em um simples papel, simula uma comanda, mas o cliente tem total controle dos seus pedidos, a necessidade de criação é quase que exclusiva da parte dos clientes, para os donos de estabelecimento, talvez ele não seja tão interessante.

8 FUNCIONAMENTO DO BOHEMIAN

O Bohemian foi construído para funcionar em qualquer dispositivo móvel com Android instalado, qualquer seja a versão, o modo da instalação atual é um protótipo, ele ainda não é a versão definitiva proposta, mas mostra o funcionamento do programa, para que o Bohemian funcione da forma em que foi idealizado precisaremos instalar o banco de aplicação em Web Service, assim seria possível promover a iteração do cliente com o garçom do estabelecimento com um banco de dados em comum nos dois aparelhos, clientes/estabelecimentos.

8.1 TELA PRINCIPAL

Figura 4 Tela de Login

Fonte: Print do aplicativo Bohemian funcionando

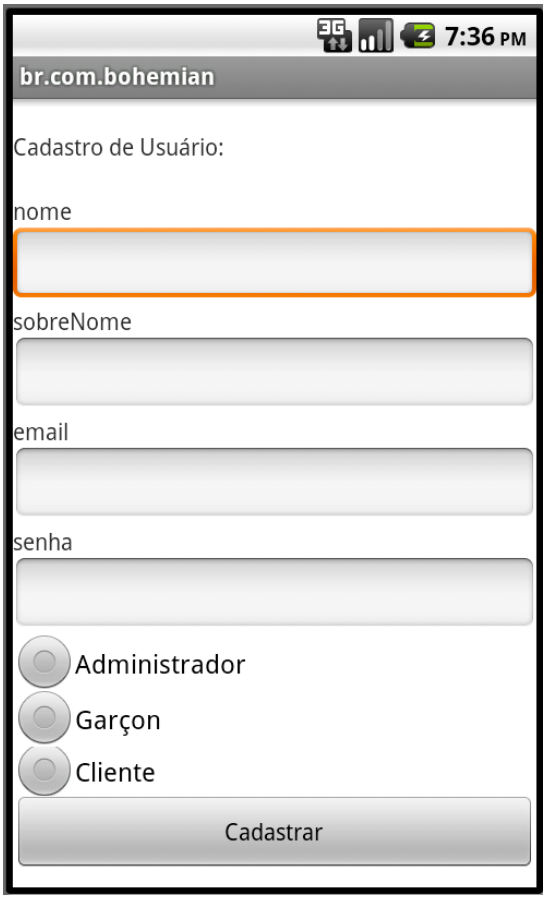
Na figura 4, temos a tela de Login do software, aqui o usuário fará o seu cadastro, no caso essa tela para o usuário servirá para que ele crie um login e uma senha e possa acessar a tela de pedidos do Bohemian, se ele ainda não cadastrado no sistema poderá criar um login clicando no botão cadastrar, o sistema vem com uma senha master para o administrador.

No momento do cadastro aplicamos uma permissão para cada usuário, se o usuário receber a permissão garçom, ele poderá apenas listar os pedidos das mesas e fechá-los depois de atendidos, se a permissão for cliente, ele terá acesso à tela de pedidos para começar a usar o software, e por ultimo se receber permissão de administrador, terá acesso à tela de cadastro de produtos, clientes, garçom e ao faturamento.

A seguir veremos as figuras das telas do cadastro de usuários, garçom e administrador respectivamente.

8.2 TELA CADASTRO

Figura 5 Cadastro de Usuários

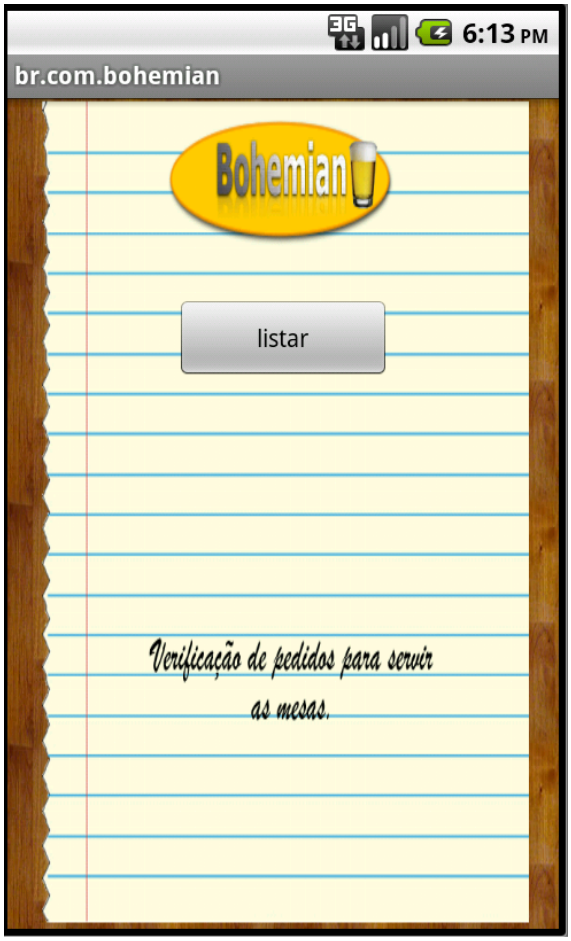


Fonte: Print do aplicativo Bohemian funcionando

Nessa tela se faz o cadastro de usuários no sistema, nota-se que abaixo temos três opções para serem marcadas, essas são as permissões que cada usuário recebe no momento do cadastro, e de acordo com a permissão ele recebe uma tela para visualizar, lembrando que essa tela é exclusiva dos usuários administradores.

8.3 TELA GARÇOM

Figura 6 Tela Garçom



Fonte: Print do aplicativo Bohemian funcionando

Nessa tela o garçom clica em listar para que possa visualizar os pedidos e servir os clientes, após encaminhar os pedidos às respectivas mesas, ele clica no item atendido, isso evita que a mesa seja servida com o mesmo pedido mais de uma vez.

8.4 TELA ADMINISTRADOR

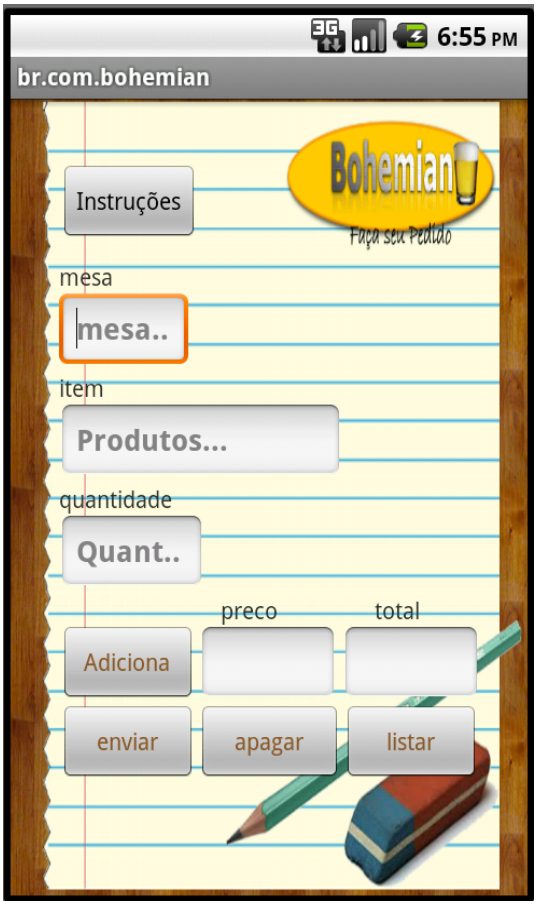
Figura 7 Tela Administrador

Fonte: Print do aplicativo Bohemian funcionando

Nessa tela o administrador do sistema faz todos os cadastros de produtos, clientes, garçons e verifica todo o faturamento do estabelecimento, tudo isso tendo que ser logar antes com a senha de administrador.

8.5 TELA DE PEDIDOS

Figura 8 Tela Pedidos

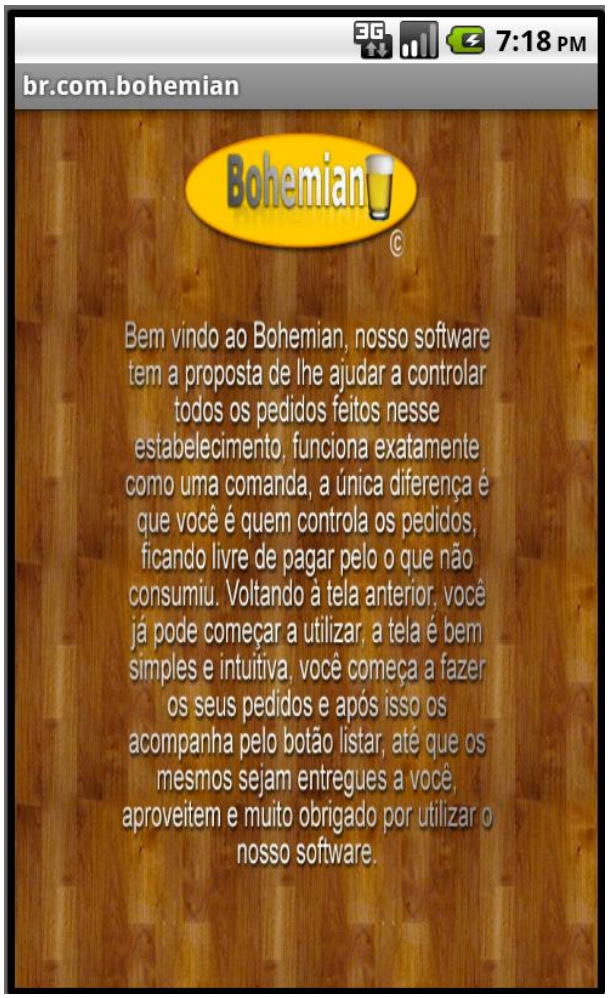


Fonte: Print do aplicativo Bohemian funcionando

A figura 8 mostra a tela em que o cliente faz os seus pedidos, o primeiro botão mostra uma breve explicação de como utilizar essa tela, o cliente preenche todos os campos desta tela com os itens que estão pré-cadastrados no banco de dados, não é possível deixar nenhum campo em branco, ao deixar campos sem preenchimento o sistema não prossegue, se o usuário clicar em enviar sem preencher os campos, o sistema apresenta uma mensagem para que o usuário preencha os campos faltantes corretamente.

8.6 INSTRUÇÕES

Figura 9 Tela Instruções

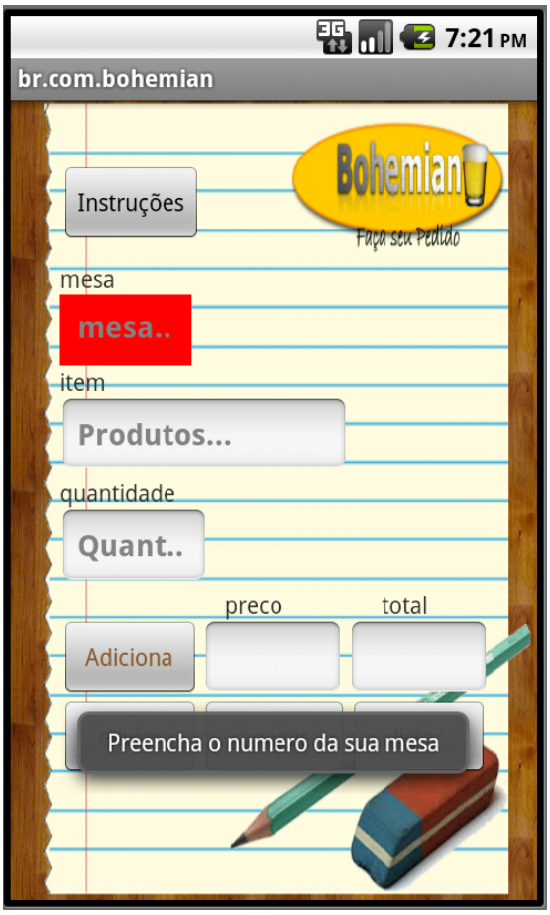


Fonte: Print do aplicativo Bohemian funcionando

Na figura 10 temos a tela com a mensagem de erro caso falte alguma informação no pedido do cliente, o sistema pede para que seja corrigido o campo, assim o pedido pode ser realizado.

8.7 ERRO TELA PEDIDOS

Figura 10 Erro Tela de Pedidos



Fonte: Print do aplicativo Bohemian funcionando

Temos na figura 11, a tela de pedidos devidamente preenchida, assim o cliente pode dar prosseguimento ao seu pedido, é necessário que digite o número da mesa, o item e a quantidade e clique em adicionar, os preços e o total serão preenchidos automaticamente pelo sistema, assim o garçom pode visualizar todos os pedidos feitos em diversas mesas e posteriormente poderá servir todos os clientes do estabelecimento através da visualização na tela de listagem, a seguir, tela preenchida corretamente.

8.8 TELA PEDIDOS PREENCHIDA

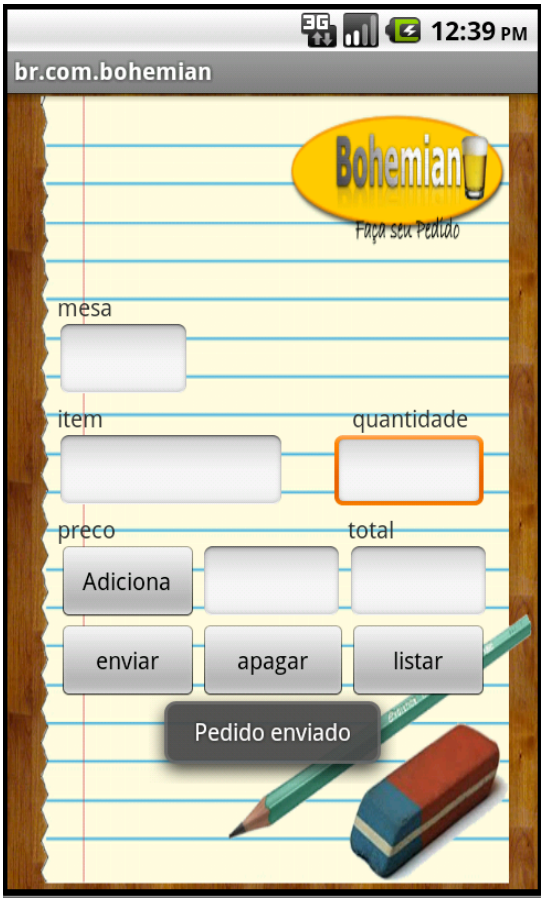
Figura 11 Tela Pedidos preenchida

Fonte: Print do aplicativo Bohemian funcionando

Após o pedido ser preenchido o cliente clica em enviar, para que o pedido seja processado e enviado para o sistema, assim o garçom poderá verificar os pedidos das mesas na tela de listar e servir os clientes veja a mensagem de confirmação de pedido que foram enviados e gravado no banco de dados na figura 12.

8.9 PEDIDOS ENVIADOS

Figura 12 Pedido Enviado



Fonte: Print do aplicativo Bohemian funcionando

Agora veremos a tela de listagem, que é chamada através do botão listar, nessa tela o cliente visualiza todos os seus pedidos, sendo que o garçom verifica em que mesa e o que servir aos respectivos clientes os pedidos são identificados por um número gerado automaticamente pelo sistema e sua respectiva mesa, vejam a figura 13.

8.0.1 TELA LISTAR

Figura 13 Tela Listar

Fonte: Print do aplicativo Bohemian funcionando

Nessa tela o cliente verifica o seu pedido, visualiza os seus pedidos, enquanto o garçom visualiza e atende todos os pedidos.

8.0.2 TELA TOTAL

Figura 14 Tela Total

Fonte: Print do aplicativo Bohemian funcionando

Finalizando, temos na figura 14 a tela de Total de pedidos, nela temos a soma de tudo o que foi pedido pela mesa, assim o cliente pode pagar a sua conta no caixa.

9 DESENVOLVIMENTO DO SOFTWARE

Agora iremos mostrar o desenvolvimento do projeto, etapa por etapa, tudo o que foi usado para a construção do Bohemian, que começa coma escolha do IDE Eclipse, ferramenta de desenvolvimento Open Source que é considerada por muitos a melhor ferramenta para desenvolvimento, o Eclipse IDE for Java EE Developers oferece edição de Java superior com compilação incremental, suporte a Java EE 5, a HTML / JSP / JSF editor gráfico, ferramentas de gerenciamento de banco de dados e suporte a mais popular servidores de aplicativos, tudo bem explicado para mostrar como chegamos ao resultado final.

9.1 BANCO DE DADOS

A aplicação usa um banco de dados open source chamado Sqlite, foi desenvolvido pela empresa Hwaki (www.hwaki.com), segundo artigo do site www.devmedia.com.br,

> "O pequeno notável Ganhador do premio Google O'Reilly 2005 Open Source Awards Winner!, o SQLite tem subido muito no conceito dos programadores, ele gera um banco de que pode ser entregue junto com a aplicação, excelente para aplicações pequenas que com um instalador tipo NNF (Next, Next, Finish), instala perfeitamente um sistema simples, sem as complicações da instalação de um cliente/servidor."

Ele é bem simples de ser utilizado, sua interface gráfica é a Sqlite Express Personal, é bastante intuitiva, para começar a criar os bancos só é preciso ter noções básicas de SQL, assim o desenvolvedor poderá criar bancos bem completos para as suas aplicações, os comandos SQL são os mesmos existentes em outros bancos de dados, e a grande vantagem de tudo isso é que ele é totalmente gratuito. Existem apenas alguns pontos negativos do banco que são as ausências de alguns comandos SQL que são:

Instrução DELETE em múltiplas tabelas;

Suporte a FOREIGN KEY;

Suporte a Triggers;

Suporte Completo ao ALTER TABLE (Somente algumas funcionalidades estão implementadas);

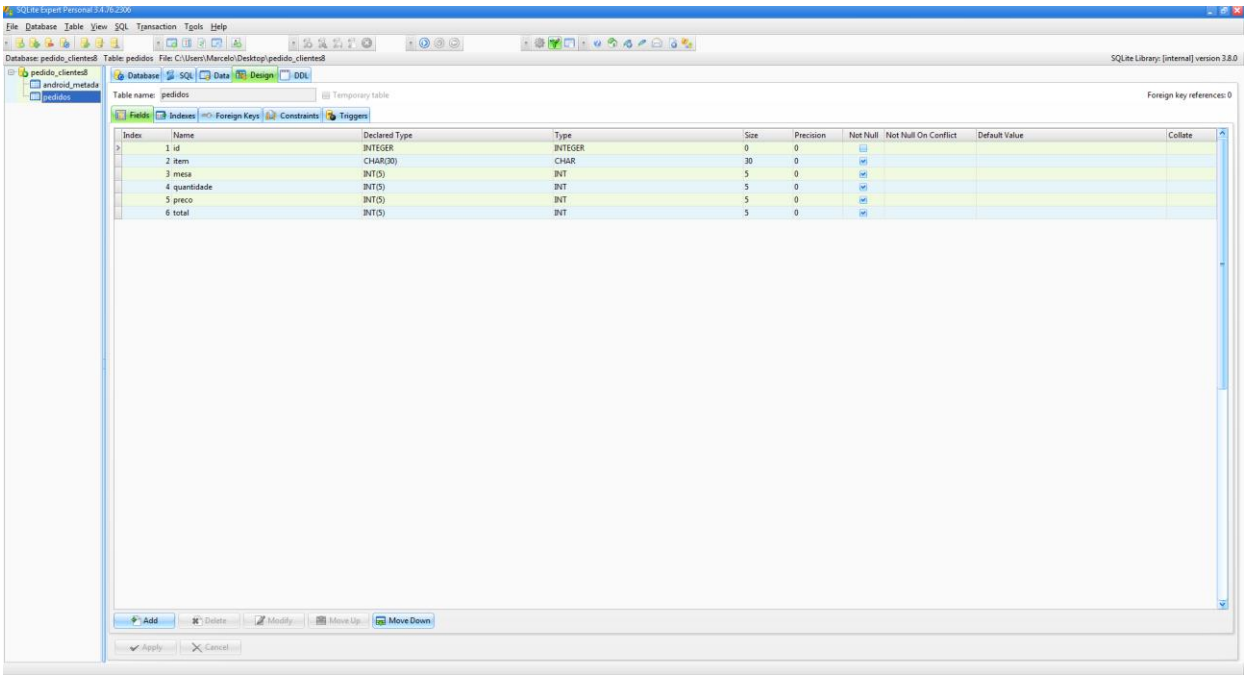
Ausência do RIGHT JOIN e FULL OUTER JOIN;

GRANT e REVOKE.

9.2 TELA DE CRIAÇÃO DE BANCOS DE DADOS

A figura mostra o layout do Sqlite para criação de bancos

Figura 15 Interface Sqlite



Fonte: Print do aplicativo Bohemian funcionando

9.3 ANDROID/JAVA

No desenvolvimento Android, é necessário usar Java juntamente com a linguagem Android, os arquivos do Android são todos em xml, todos os layouts que aparecem nas telas são feitos em xml, para cada tela criada para o Android, temos que criar uma nova classe Activity em Java, para que as duas se tornem uma só dentro do software, dentro das Activitys tem especificado qual é o layout xml que as mesmas estão utilizando. E o principal, é necessário registrar as Activitys no arquivo chamado Manifest XML, ele gerencia todas as Activitys existentes no sistema. Funciona assim, no xml adicionamos os campos que os usuários utilizarão para digitar os dados, cada campo tem seu nome e respectivo id, após ser construído o xml, é hora de fazer as classes que irão recuperar os valores digitados nas caixas de diálogo, e os cliques que serão feitos no botões, veja um exemplo de recuperação de um Button (Botão) no código que está na tela principal, é a chamada da tela pedidos através do botão chamado btnPedidos:

```
btnPedidos.setOnClickListener(new View.OnClickListener() {

        public void onClick(View v) {

    startActivity(new Intent(getBaseContext(),Pedido.class));

        }

});
```

9.4 STYLES

A maioria das letras, backgrounds e cores nas caixas de texto e botões, estão configurados em um arquivo externo xml chamado Styles, nesse arquivo configuramos as aparências das caixas de texto, botões, textos em geral, seu funcionamento é similar ao arquivo css no HTML, serve para organizarmos os códigos xml e deixamos mais limpos e sem repetição durante a fase de desenvolvimento.

9.5 TELA LOGIN

Essa tela é onde o usuário tem a opção de criar logins e senhas para a utilização do sistema, os acessos são dados com permissões de administrador, cliente e garçom, cada uma recebe privilégios distintos dentro do sistema e visualiza a tela de acordo com esses provilégios.

9.6 TELA CADASTRO DE USUÁRIOS

Ao realizar o cadastro de usuários, o administrador delega as permissões que o determinado usuário receberá, o sistema virá com uma senha master para que os administradores possam realizar todas as tarefas de cadastro e parametrização no sistema.

Ao ser cadastrado no sistema o usuário receberá a tela referente aos privilégios que recebeu se for garçom, poderá apenas visualizar os pedidos, caso seja o cliente, receberá a tela de Pedidos após o login e por fim se for um administrador, receberá a tela de administração do software.

9.7 TELA PEDIDOS

Nessa tela é onde temos mais chamadas de métodos e Activitys, é onde se concentra quase todos as chamadas de telas e operações do sistema, nossas regras de negócio, nessa tela temos como listar os pedidos, e enviar ao banco de dados, além de validações como de não deixar que o cliente envie um pedido em branco, que digite incorretamente o nome dos itens, e faz a confirmação dos pedidos.

Usamos o método findViewById para recuperar os comandos dos xmls nas classes Java, instanciamos também os objetos para que possam ser inicializados e utilizados no nosso código.

9.8 TELA DB (BANCO DE DADOS)

Essa é a Activity que faz a criação do banco de dados e das tabelas, utilizados no projeto, ela tem que herdar os métodos de uma classe abstrata chamada SqliteOpenHelper para que possamos fazer as operações no banco de dados e trabalhar com os design patterns, no nosso projeto usamos o DAO, que nos possibilita também trabalhar com os objetos Pojos.

Ao herdar dessa classe, temos classe dois métodos que são muito úteis para futuras alterações no software.

```
public void onCreate(SQLiteDatabase db) {

    db.execSQL(sql);}
```

Esse método cria o banco de dados na aplicação, quando o usuário acessa o software pela primeira vez, esse método cria o banco de dados em modo exclusivo, só para que a nossa aplicação utilize esse banco.

```
public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {

}
```

Esse método pode ser utilizado quando houver a necessidade de mudança no banco de dados do software, se na versão anterior foram detectadas algumas necessidades como adição ou subtração de tabelas, os comandos podem ser executados nesse método, assim a nova versão passará a funcionar com todas as alterações necessárias.

Existem outros métodos para a criação do banco de dados, mas a aplicação utiliza as melhores práticas de programação, o DAO (Data Access Object) como design patterns, isso deixa o software mais leve e principalmente de fácil manutenção, o trabalho em equipe fica muito mais prático já que é

preciso apenas alterar a classe DAO para acrescentar o remover argumentos no código na parte de conexão de dados.

9.9 DESIGN PATTERNS

O Design Patterns é um padrão de desenvolvimento de software, são soluções para problemas conhecidos no desenvolvimento de software. É uma solução para um problema de desenvolvimento de software recorrente, enfatizando um contexto e forçando a aproximação do problema, e as consequências e os impactos de sua solução. São dispositivos que permitem que os programas compartilhem conhecimento sobre o seu desenvolvimento. Quando programamos, encontramos problemas que sempre insistem em ocorrer. A questão é como agir para solucionar este problema desta vez? Colocando tudo isso em uma documentação, o padrão (pattern) é uma maneira em que podermos reutilizar e compartilhar informações que aprendemos sobre a melhor maneira de se resolver um problema de desenvolvimento de software.

9.0.1TELA PEDIDOSPOJO

Essa Activity é onde é feito o mapeamento dos campos da aplicação, o mapeamentos dos objetos relacionais, os nossos Pojos. Declaramos os atributos privados da classe e após geramos o Getters and Setters, assim temos o espelho do nosso banco de dados, como resultado temos uma classe criada com todos os atributos relacionados ao banco de dados necessários.

9.0.2 TELA PEDIDOSDAO

Essa classe é responsável pela persistência dos dados no banco de dados, nela utilizamos os famosos métodos CRUD (Create, Read, Update e Delete), que são as funções básicas do banco de dados, o DAO sempre recebe um objeto Pojo, assim não relacionamos a interface do software com o acesso aos dados, e é importante dizer que para chegarmos a classe DB através do DAO, temos que ir passando contextos na classe, o uso do DAO se faz necessário porque como já dito, buscamos usar as melhores práticas de programação para que o desenvolvimento siga um padrão.

9.0.3 TELA PEDIDOSADAPTERS E ADAPTERGARCOM

Essas Activitys que também herdam métodos de uma classe abstrata chamada Base adapter criam os adaptadores para que seja possível acesso aos dados e trazer listagens de dados no sistema, o adapter funciona como dataset, ou datasource similar em linguagens como o Delphi, ele busca as informações na base de dados para trabalharmos com esses dados na aplicação, as telas são exatamente iguais, só foram duplicadas para que o cliente e garçom verifiquem os pedidos ao mesmo tempo.

9.0.4 TELA LISTAR E LISTAR2

Essas Activitys retornam os resultados que foram digitados no banco de dados, herdam os métodos de uma classe abstrata chamada ListActivity, não podemos ter um setContentView nessa classe, pois, quando usamos um ListActivity e Adapters configuramos o layout no próprio adapter, se definirmos o layout novamente nessa classe, a mesma não será executada. Esta Activity é a que menos tem linhas de código, devido à concentração de dada ser toda feita nas classes Pojo e DAO.

No desenvolvimento desse software, temos para cada tabela do banco de dados, um adapter, um DAO e um Pojo, para uma boa prática de desenvolvimento é necessário que seja feito desta forma, assim temos o controle total dos dados concentrados nessas classes e muito mais agilidade no acesso aos dados, mais uma vez foi duplicada a Activity para a visualização de clientes e garçons.

9.0.5 TELA FECHAMENTO

Nessa Activity temos o fechamento dos pedidos feitos pelos clientes, à soma de todos os pedidos é controlado por cada mesa, o cliente digita o

número da mesa em que estava sentado assim o cliente finaliza a operação com o software, verificando se está tudo certo com os pedidos realizados e posteriormente paga a sua comanda, sem problemas com os itens da mesma, já que ele mesmo os fez e acompanhou passo a passo até a entrega, com isso foi explicado com muitos detalhes todo o funcionamento do Bohemian.

A idéia de utilização do sistema é essa, simples e objetiva, para que o software possa ser utilizado em produção ainda será preciso implementar uma série de modificações e melhorias, mas a idéia principal continuará a mesma.

10 CONCLUSÃO

O Android é um sistema operacional com infinitas possibilidades de criação, o programador pode desenvolver softwares em todos os ramos do

conhecimento, pode ajudar a resolver grandes problemas através de softwares específicos, dependendo da necessidade dos seus clientes ou da sua própria, por ser um software livre existe grande base de conhecimento na própria internet, os programadores contribuem para que cada vez mais o Android melhore e seja difundido. A fácil integração do Eclipse com o plugin de desenvolvimento Android, também facilita o muito, uma vez que a IDE acomoda bem as duas ferramentas integradas e o uso das duas é muito rápido para que o desenvolvedor assimile o uso.

Nesse caso através de uma necessidade bem simples, conseguimos achar uma solução prática e interessante para os usuários controlarem os seus pedidos através de uma simples comanda, funcional e muito fácil de ser utilizada, ela pode ser instalado em qualquer dispositivo móvel com Android como sistema operacional, é uma proposta, um protótipo, pode ser que realmente venha a ser útil um dia, mas ajuda a entender o funcionamento da linguagem Android/Java e completar o ciclo no curso de sistemas de informação, agregando um vasto conhecimento nas pessoas envolvidas nesse projeto.

11 REFERÊNCIAS BIBLIOGRÁFICAS

LECHETA, R. R. **Google Android - Aprenda a Criar Aplicações para Dispositivos Móveis com o Android SDK**. São Paulo: Editora Novatec, 2010. 608p.

http://developer.android.com/tools/sdk/eclipse-adt.html Acessado em: 20/09/2013.

http://www.felipesilveira.com.br/2010/03/comecando-a-desenvolver-aplicativos-para-android/ Acessado em 25/08/2013

http://www.motorola.com/sites/motodev/us-en/motodev_lp.html

Acessado em 10/09/2013

https://developer.android.com/training/index.html Acessado em 28/09/2013

http://www.mobiltec.com.br/blog/index.php/tutorial-desenvolvimento-android-passo-a-passo/ Acessado em 08/09/2013

http://www.bspcn.com

Acessado em 10/09/2013

http://www.devmedia.com.br/videoaulas acessado de 01/05/2013 até 29/10/2013

12 CODIFICAÇÃO DO PROJETO BOHEMIAN

Activity Login

```
package com.example.br.bohemian;
import android.app.Activity;
import android.content.Context;
import android.content.Intent;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.os.Bundle;
import android.view.View;
import android.widget.EditText;
import android.widget.Toast;

public class Login extends Activity {
	@Override
	protected void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.login);

	}

	public void LoginOnClick(View v) {

		EditText txtUsuario = (EditText) findViewById(R.id.edtUsuario);
		EditText txtSenha = (EditText) findViewById(R.id.edtSenha);

		if (txtUsuario.getText().toString().length() <= 0) {
			txtUsuario.setError("Preenchimento obrigatório!");
			txtUsuario.requestFocus();
		} else if (txtSenha.getText().toString().length() <= 0) {
			txtSenha.setError("Preenchimento obrigatório!");
			txtSenha.requestFocus();
		} else {
			try {
				String usuario = txtUsuario.getText().toString();
				String senha = txtSenha.getText().toString();

				SQLiteDatabase db =
openOrCreateDatabase("pedido_clientes.db",
						Context.MODE_PRIVATE, null);

				Cursor c = db
						.rawQuery(
							"SELECT  email, senha,
permissao FROM usuarios WHERE email=?",
							new String[] {
String.valueOf(usuario) });

				if (c.moveToFirst()) {

					String busuario = c.getString(0);
					String bsenha = c.getString(1);
					int permissao = Integer.parseInt(c.getString(2));
					if (usuario.equals(busuario) &&
senha.equals(bsenha)) {
```

```
                                        if (permissao == 1) {
                                                Intent it = new
Intent(getBaseContext(),
                                                        Pedido3.class);/*
Abre a classe e instancia o objeto do tipo intente e chama a classe
                a
                                                .*/
                                                startActivity(it);// inicializa a nova
classe "it"

                                        } else if (permissao == 2) {
                                                Intent it = new
Intent(getBaseContext(),
                                                        Pedido2.class);/*
Abre a classe e instancia o objeto do tipo intente e chama a classe a.*/
                                                startActivity(it);// inicializa a nova
classe "it"

                                        } else if (permissao == 3) {
                                                Intent it = new
Intent(getBaseContext(),
                                                        Pedido.class);/* Abre
a classe e instancia o objeto do tipo intente e chama a classe
            a
                                                .*/

                                        }
                                        limpa();// Metodo para apagar campos

                                } else {
                                        Toast.makeText(getBaseContext(), "Senha
inválida!",

        Toast.LENGTH_SHORT).show();
                                }
                        } else {
                                Toast.makeText(getBaseContext(), "Usuário não
cadastrado!",
                                        Toast.LENGTH_SHORT).show();
                        }

                } catch (Exception ex) {
                        Toast.makeText(getBaseContext(), ex.getMessage(),
                                Toast.LENGTH_SHORT).show();
                }
        }

    }

    private void limpa() {
        EditText txtUsuario = (EditText) findViewById(R.id.edtUsuario);
        EditText txtSenha = (EditText) findViewById(R.id.edtSenha);
        txtUsuario.setText("");
        txtSenha.setText("");

    }
```

```
	public void InserirUsuarioOnClick(View v) {
		Intent it = new Intent(getBaseContext(), InserirUsuarioActivity.class);/*
Abre a classe e instancia o objeto do tipo intente e chama a classe
		 a

                                                                .*/
		startActivity(it);// inicializa a nova classe "it"

	}

}
```

Activity DB

```
package com.example.br.bohemian;

import android.content.Context;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteDatabase.CursorFactory;
import android.database.sqlite.SQLiteOpenHelper;

public class DB extends SQLiteOpenHelper {

	private static String dbNome = ("pedido_clientes.db");
	private static String sql = "CREATE TABLE [pedidos] (  [id] INTEGER
PRIMARY KEY AUTOINCREMENT,  [item] [CHAR(30)] NOT NULL,  [mesa] [INT(5)]
NOT NULL,  [quantidade] [INT(5)] NOT NULL,  [preco] [INT(5)] NOT NULL,  [total]
[INT(5)] NOT NULL);";
	public static int version = 1;

	public DB(Context contx) {
		super(contx, dbNome, null, version);

	}

	@Override
	public void onCreate(SQLiteDatabase db) {
		db.execSQL(sql);

	}

	@Override
	public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {

	}

}
```

Activity Fechamento

```
package com.example.br.bohemian;

import java.sql.*;
import java.util.ArrayList;
import java.util.List;

import com.example.br.bohemian.R.id;
```

```
import android.app.Activity;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.CheckBox;
import android.widget.EditText;
import android.widget.Toast;

public class Fechamento extends Activity {
	@Override
	public void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.fechamento);

		final EditText numMesa;
		final EditText vTotal;

		numMesa = (EditText) findViewById(R.id.txtMesa);
		vTotal = (EditText) findViewById(R.id.txtTotal);

		Button btnSomar = (Button) findViewById(R.id.btnSomar);
		Button btnFechar = (Button) findViewById(R.id.btnFech_Conta);

		btnSomar.setOnClickListener(new View.OnClickListener() {
			public void onClick(View v) {

				String mesa = numMesa.getText().toString();
				// Toast.makeText(getBaseContext(), mesa,
				// Toast.LENGTH_SHORT).show();

				SQLiteDatabase db =
openOrCreateDatabase("pedido_clientes.db",
						Context.MODE_PRIVATE, null);

				Cursor c = db.rawQuery("SELECT SUM(quantidade *
preco) AS Total FROM pedidos WHERE mesa=?",new String[] { String.valueOf(mesa)
});

				if (numMesa.getText().toString().equals("")){


					Toast.makeText(getBaseContext(), "Preencha o
número de sua mesa", Toast.LENGTH_SHORT).show();


				}

				else if (c.moveToNext()) {
					EditText txtMesa = (EditText)
findViewById(R.id.txtMesa);
```

```
                                        EditText txtTotal = (EditText)
findViewById(R.id.txtTotal);

                                        txtTotal.setText(c.getString(0));

                                }
                                        else {
                                        Toast.makeText(getBaseContext(),
                                                "Não há pedidos na mesa
informada!",
                                                Toast.LENGTH_SHORT).show();



                                }

                        }

                });

        }

}
```

Activity Inserir Usuários

```
package com.example.br.bohemian;
import android.app.Activity;
import android.content.ContentValues;
import android.content.Context;
import android.content.Intent;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.os.Bundle;
import android.view.View;
import android.widget.EditText;
import android.widget.RadioGroup;
import android.widget.Toast;

public class InserirUsuarioActivity extends Activity{
        @Override
        protected void onCreate(Bundle savedInstanceState) {
                super.onCreate(savedInstanceState);
                setContentView(R.layout.form_inserirususario);
        }
        private int rdGSelecionado;

public void onClickCadastrarUsuario(View v){

                //Recupera a instacia dos campos digitado.
                EditText txtNome = (EditText)findViewById(R.id.txtNome);
                EditText txtSobreNome = (EditText)findViewById(R.id.txtSobreNome);
                EditText txtEmail = (EditText)findViewById(R.id.txtEmail);
                EditText txtSenha = (EditText)findViewById(R.id.txtSenha);

                RadioGroup grupo = (RadioGroup)findViewById(R.id.rdGPermissao);
```

```
		grupo.setOnCheckedChangeListener(new
RadioGroup.OnCheckedChangeListener() {

			public void onCheckedChanged(RadioGroup grupo, int
checkedId) {
				switch(checkedId){
				case R.id.rdAdmin:{
					rdGSelecionado = 1;
					break;
				}

				case R.id.rdGarcon:{
					rdGSelecionado = 2;
					break;
				}

				case R.id.rdCliente:{
					rdGSelecionado = 3;
					break;
				}

				}

			}
		});

		//Valida campos

		int permissao = grupo.getCheckedRadioButtonId();
		permissao = rdGSelecionado;
		if(txtNome.getText().toString().length() <= 0){
			txtNome.setError("O campo 'Nome' não pode ficar em branco!");
			txtNome.requestFocus();
		}else if(txtSobreNome.getText().toString().length() <= 0){
			txtSobreNome.setError("O campo 'Sobre Nome' não pode ficar
em branco!");
			txtSobreNome.requestFocus();
		}else if(txtEmail.getText().toString().length() <= 0){
			txtEmail.setError("O campo 'E-mail' não pode ficar em branco!");
			txtEmail.requestFocus();
		}else if(txtSenha.getText().toString().length() <= 0){
			txtSenha.setError("O campo 'Senha' não pode ficar em branco!");
			txtSenha.requestFocus();
		}else if(permissao <= 0){
			Toast.makeText(getBaseContext(), "Selecione a permissão!",
Toast.LENGTH_SHORT).show();
		}else{
			try{
				SQLiteDatabase db =
openOrCreateDatabase("pedido_clientes.db", Context.MODE_PRIVATE, null);//Abre a
conexão com o banco


				ContentValues cadusu = new ContentValues();//Cria um
novo objeto do tipo ContentValues e depois chamar o metodo put para inserir os
dados
```

```
                                cadusu.put("nome", txtNome.getText().toString());
                                cadusu.put("sobreNome",
txtSobreNome.getText().toString());
                                cadusu.put("email", txtEmail.getText().toString());
                                cadusu.put("senha", txtSenha.getText().toString());
                                cadusu.put("permissao", permissao);

                                if(db.insert("usuarios", "_id", cadusu) > 0){//insere na
tabela usuarios. Se tiver algum campo Autoincrement (id) especificar.
                                        limpar();
                                Toast.makeText(getBaseContext(), "Inserido com
sucesso", Toast.LENGTH_SHORT).show();
                                }else{
                                        Toast.makeText(getBaseContext(), "Erro ao
gravar", Toast.LENGTH_SHORT).show();
                                }
                        }catch(Exception ex){
                                Toast.makeText(getBaseContext(), ex.getMessage(),
Toast.LENGTH_SHORT).show();
                        }

                }


        }

        public void limpar(){
                EditText txtNome = (EditText)findViewById(R.id.txtNome);
                EditText txtSobreNome = (EditText)findViewById(R.id.txtSobreNome);
                EditText txtEmail = (EditText)findViewById(R.id.txtEmail);
                EditText txtSenha = (EditText)findViewById(R.id.txtSenha);

                txtNome.setText("");
                txtSobreNome.setText("");
                txtEmail.setText("");
                txtSenha.setText("");


        }



        public void limpa(){

                EditText txtUsuario = (EditText) findViewById(R.id.edtUsuario);
                EditText txtSenha = (EditText) findViewById(R.id.edtSenha);
                txtUsuario.setText("");
                txtSenha.setText("");
        }



}
```

Activity Instruções

```
package com.example.br.bohemian;

import android.app.Activity;
import android.os.Bundle;

public class Instrucoes extends Activity {
        @Override
          public void onCreate(Bundle savedInstanceState) {
             super.onCreate(savedInstanceState);
             setContentView(R.layout.instrucoes);

}
} package com.example.br.bohemian;

import com.example.br.bohemian.R.id;

import br.Bohemian.adapters.PedidosAdapter;
import br.Bohemian.dao.PedidosDao;
import br.Bohemian.pojo.PedidosPojo;
import android.app.Activity;
import android.app.AlertDialog.Builder;
import android.app.ListActivity;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.ListView;
import android.widget.Toast;

public class Listar extends ListActivity {
        @Override
        public void onCreate(Bundle savedInstanceState) {
                super.onCreate(savedInstanceState);

                PedidosDao dao = new PedidosDao(getBaseContext());

                setListAdapter(new PedidosAdapter(getBaseContext(), dao.getAll()));

        }

        @Override
        protected void onListItemClick(ListView l, View v, int position, long id) {

                startActivity(new Intent(getBaseContext(), Fechamento.class));

        }
}

Activity Listar2
package com.example.br.bohemian;

import com.example.br.bohemian.R.id;

import br.Bohemian.adapters.PedidosAdapter;
```

```
import br.Bohemian.dao.PedidosDao;
import br.Bohemian.pojo.PedidosPojo;
import android.app.Activity;
import android.app.AlertDialog.Builder;
import android.app.ListActivity;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.ListView;
import android.widget.Toast;

public class Listar extends ListActivity {
	@Override
	public void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);

		PedidosDao dao = new PedidosDao(getBaseContext());

		setListAdapter(new PedidosAdapter(getBaseContext(), dao.getAll()));

	}

	@Override
	protected void onListItemClick(ListView l, View v, int position, long id) {

		startActivity(new Intent(getBaseContext(), Fechamento.class));

	}
}
```

Activity Pedidos

```
package com.example.br.bohemian;

import java.util.ArrayList;
import java.util.List;

import android.R.integer;
import android.app.Activity;
import android.app.AlertDialog.Builder;
import android.content.Context;
import android.content.Intent;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.graphics.Color;
import android.os.Bundle;
import android.text.Editable;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.AutoCompleteTextView;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;
```

```
import br.Bohemian.dao.PedidosDao;
import br.Bohemian.pojo.PedidosPojo;

import com.example.br.bohemian.R.id;

public class Pedido extends Activity {

	private EditText txtMesa;
	private EditText txtItem;
	private EditText txtQuantidade;
	private EditText txtPreco;
	private EditText txtTotal;
	private double total = 0;
	private EditText x;
	private EditText y;

	public void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.pedido);

		iniProdutos();

		TextView Instrucoes = (TextView) findViewById(id.Leiame);
		Instrucoes.setOnClickListener(new View.OnClickListener() {

			public void onClick(View v) {

				startActivity(new Intent(getBaseContext(),
Instrucoes.class));

			}
		});

		/* Recuperação do botão do xml pedidos */
		Button btnEnviar = (Button) findViewById(id.btnEnviar);
		/* Declarando as variávei s serem utilizadas */
		txtMesa = (EditText) findViewById(id.txtMesa);
		txtItem = (EditText) findViewById(id.txtItem);
		txtQuantidade = (EditText) findViewById(id.txtQuantidade);
		txtPreco = (EditText) findViewById(id.txtPreco);
		txtTotal = (EditText) findViewById(id.txtTotal);

		btnEnviar.setOnClickListener(new View.OnClickListener() {

			public void onClick(View v) {

				/* Estanciando o Pojo */
				PedidosPojo pj = new PedidosPojo();

				pj.setItem(txtItem.getText().toString());
				pj.setMesa(txtMesa.getText().toString());
				pj.setQuantidade(txtQuantidade.getText().toString());
				pj.setPreco(txtPreco.getText().toString());
				pj.setTotal(txtTotal.getText().toString());

				/* Estanciando o Dao */
```

```
                        PedidosDao dao = new PedidosDao(getBaseContext());

                        if (txtMesa.getText().toString().equals("")) {
                                txtMesa.setBackgroundColor(Color.RED);

                                Toast t = Toast
                                                .makeText(getBaseContext(),
                                                                "Preencha o numero
da sua mesa",

        Toast.LENGTH_SHORT);
                                t.show();
                                txtMesa.requestFocus();
                        }
                        /*
                         * Adicionando regras para que não seja digitadas
informações
                         * inconsistentes
                         */

                        else if (txtItem.getText().toString().equals("")) {

                                Toast t = Toast.makeText(getBaseContext(),
                                                "Preencha qual o item escolhido",
                                                Toast.LENGTH_SHORT);
                                t.show();
                                txtMesa.requestFocus();
                        }

                        else if (txtQuantidade.getText().toString().equals("")) {

                                Toast t = Toast.makeText(getBaseContext(),
                                                "Qual a quantidade?",
Toast.LENGTH_SHORT);
                                t.show();
                                txtMesa.requestFocus();
                        }

                        else {
                                dao.insert(pj);

                                Toast t = Toast.makeText(getBaseContext(),
                                                "Pedido enviado",
Toast.LENGTH_SHORT);
                                t.show();

                        }

                        txtMesa.setText("");
                        txtItem.setText("");
                        txtQuantidade.setText("");
                        txtPreco.setText("");
                        txtTotal.setText("");

                }
```

```
		});

		/* Achamad da tela Listar */
		findViewById(R.id.btnListar).setOnClickListener(
				new View.OnClickListener() {

					public void onClick(View v) {

						startActivity(new Intent(getBaseContext(),
Listar.class));

					}
				});
		/* Evento botão apagar */
		findViewById(R.id.btnApagar).setOnClickListener(
				new View.OnClickListener() {

					public void onClick(View arg0) {

						txtMesa.setText("");
						txtItem.setText("");
						txtQuantidade.setText("");
						txtPreco.setText("");
						txtTotal.setText("");

					}
				});

		/* Adicionando os preços conforme digitação do cliente */
		findViewById(R.id.btnAdiciona).setOnClickListener(
				new View.OnClickListener() {

					public void onClick(View arg0) {

						if (txtItem.getText().toString()
								.equals("coca-cola lata")) {
							txtPreco.setText("3.50");

							double val1 =
Double.parseDouble(txtPreco.getText()
									.toString());
							double val2 =
Double.parseDouble(txtQuantidade
									.getText().toString());

							txtTotal.setText(String.valueOf(val1
* val2));

						}

						else if (txtItem.getText().toString()
								.equals("porcão camarao")) {
							txtPreco.setText("35.50");

							double val1 =
Double.parseDouble(txtPreco.getText()
```

```
                                        .toString());
                        double val2 =
Double.parseDouble(txtQuantidade
                                        .getText().toString());

                        txtTotal.setText(String.valueOf(val1
* val2));

                }

                else if (txtItem.getText().toString().equals("x
bacon")) {
                        txtPreco.setText("8.50");

                        double val1 =
Double.parseDouble(txtPreco.getText()
                                        .toString());
                        double val2 =
Double.parseDouble(txtQuantidade
                                        .getText().toString());

                        txtTotal.setText(String.valueOf(val1
* val2));

                }

                else if (txtItem.getText().toString()
                                .equals("cerveja skol")) {
                        txtPreco.setText("8.50");

                        double val1 =
Double.parseDouble(txtPreco.getText()
                                        .toString());
                        double val2 =
Double.parseDouble(txtQuantidade
                                        .getText().toString());

                        txtTotal.setText(String.valueOf(val1
* val2));

                }

                else if (txtItem.getText().toString()
                                .equals("batata frita")) {
                        txtPreco.setText("8.50");

                        double val1 =
Double.parseDouble(txtPreco.getText()
                                        .toString());
                        double val2 =
Double.parseDouble(txtQuantidade
                                        .getText().toString());

                        txtTotal.setText(String.valueOf(val1
* val2));
```

```
                                        }

                                        else {

                                                Toast.makeText(getBaseContext(),
                                                        "Item não cadastrado
no sistema",

        Toast.LENGTH_LONG).show();
                                        }

                                    }

                                });

        }

        public void iniProdutos() {

                ArrayAdapter<String> adaptador = new ArrayAdapter<String>(this,
                        android.R.layout.simple_dropdown_item_1line,
getAllStrings());

                AutoCompleteTextView produtos = (AutoCompleteTextView)
findViewById(R.id.txtItem);
                produtos.setAdapter(adaptador);
        }

        public List<String> getAllStrings() {

                List<String> produtoList = new ArrayList<String>();
                // Abre o banco
                SQLiteDatabase db = openOrCreateDatabase("pedido_clientes.db",
                        Context.MODE_PRIVATE, null);
                // Seleciona tudo
                Cursor cursor = db.rawQuery("SELECT * FROM produtos", null);

                if (cursor.moveToFirst()) {

                        do {
                                String produto = new String();
                                produto = cursor.getString(1);
                                produtoList.add(produto);
                        } while (cursor.moveToNext());

                        // Fecha conexão
                        db.close();
                }

                // return produtoList
                return produtoList;

        }

}
```

Activity Pedidos2

```
package com.example.br.bohemian;

import java.util.ArrayList;
import java.util.List;

import android.R.integer;
import android.app.Activity;
import android.app.AlertDialog.Builder;
import android.content.Context;
import android.content.Intent;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.graphics.Color;
import android.os.Bundle;
import android.text.Editable;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.AutoCompleteTextView;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
import br.Bohemian.dao.PedidosDao;
import br.Bohemian.pojo.PedidosPojo;

import com.example.br.bohemian.R.id;

public class Pedido2 extends Activity {



	public void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.pedido2);


		findViewById(R.id.btnListar).setOnClickListener(
				new View.OnClickListener() {

					public void onClick(View v) {

						startActivity(new Intent(getBaseContext(),
Listar2.class));

					}
				});



	}


	}
```

Activity Pedidos3

```
package com.example.br.bohemian;

import java.util.ArrayList;
import java.util.List;

import android.R.integer;
import android.app.Activity;
import android.app.AlertDialog.Builder;
import android.content.Context;
import android.content.Intent;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.graphics.Color;
import android.os.Bundle;
import android.text.Editable;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.AutoCompleteTextView;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
import br.Bohemian.dao.PedidosDao;
import br.Bohemian.pojo.PedidosPojo;
import com.example.br.bohemian.R.id;
public class Pedido3 extends Activity {

	public void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.pedido3);



	}
}
```

Arquivos XML Android

Fechamento

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:background="@drawable/fechar">


    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="66dp"
        android:text="Qual é a sua mesa?" />



    <EditText
```

```
        android:id="@+id/txtMesa"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/textView1"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="22dp"
        android:ems="10"
        android:inputType="number" >

        <requestFocus />
    </EditText>



    <Button
        android:id="@+id/btnSomar"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/txtMesa"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="20dp"
        android:text="Somar" />

    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="true"
        android:text="O valor do seu consumo é :" />

    <EditText
        android:id="@+id/txtTotal"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/txtMesa"
        android:layout_below="@+id/textView2"
        android:editable="false"
        android:layout_marginTop="14dp"
        android:ems="10" />


    <Button
        android:id="@+id/btnFech_Conta"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/txtTotal"
        android:layout_alignRight="@+id/txtTotal"
        android:layout_below="@+id/txtTotal"
        android:layout_marginTop="21dp"
        android:text="Fechar Conta" />

</RelativeLayout>
```

Inserir Usuário

```
<?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:orientation="vertical" >

<TextView
    android:id="@+id/txtTexto"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:paddingBottom="30px"
    android:paddingTop="30px"
    android:text="Cadastro de Usuário:" />

<TextView
    android:id="@+id/txtTexto"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/nome" />

<EditText
    android:id="@+id/txtNome"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:ems="10"
    android:inputType="textPersonName" >
</EditText>

<TextView
    android:id="@+id/txtNumMesa"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/sobreNome" />

<EditText
    android:id="@+id/txtSobreNome"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:ems="10"
    android:inputType="textPersonName" />

<TextView
    android:id="@+id/textView3"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/email" />

<EditText
    android:id="@+id/txtEmail"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:ems="10"
    android:inputType="textEmailAddress" />

<TextView
    android:id="@+id/textView4"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/senha" />

<EditText
```

```
        android:id="@+id/txtSenha"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:ems="10"
        android:inputType="textPassword" />

    <ScrollView
        android:id="@+id/scrollView1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" >

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:orientation="vertical" >

            <RadioGroup
                android:id="@+id/rdGPermissao"
                android:layout_width="match_parent"
                android:layout_height="wrap_content" >

                <RadioButton
                    android:id="@+id/rdAdmin"
                    android:layout_width="wrap_content"
                    android:layout_height="34dp"
                    android:checked="false"
                    android:text="Administrador" />

                <RadioButton
                    android:id="@+id/rdGarcon"
                    android:layout_width="wrap_content"
                    android:layout_height="34dp"
          android:checked="false"
                    android:text="Garçon" />

                <RadioButton
                    android:id="@+id/rdCliente"
                    android:layout_width="wrap_content"
                    android:layout_height="30dp"
                    android:checked="false"
                    android:text="Cliente" />

            </RadioGroup>

        </LinearLayout>
    </ScrollView>

    <Button
        android:id="@+id/Leiame"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:onClick="onClickCadastrarUsuario"
        android:text="@string/Cadastrar" />

</LinearLayout>
```

Instruções

```
<?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:orientation="vertical"
    android:background="@drawable/logo_inst" >



</LinearLayout>
```

Listar

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >

    <ImageView
        android:id="@+id/imageView1"
        android:layout_width="50dp"
        android:layout_height="50dp"
        android:layout_alignParentLeft="true"
        android:layout_alignParentTop="true"
        android:layout_marginLeft="15dp"
        android:src="@drawable/logo" />

    <TextView
        android:id="@+id/textView1"
        style="@style/CxTextos"
        android:layout_alignParentLeft="true"
        android:layout_alignParentRight="true"
        android:layout_below="@+id/imageView1"
        android:layout_marginLeft="20dp"
        android:text="Pedido"
        android:textColor="#1874CD" />

    <TextView
        android:id="@+id/txtID"
        style="@style/CxTextos"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/textView1"
        android:layout_marginLeft="20dp"
        android:text="TextView" />

    <TextView
        android:id="@+id/textView5"
        style="@style/CxTextos"
        android:layout_alignParentLeft="true"
        android:layout_below="@+id/txtID"
        android:layout_marginLeft="20dp"
        android:layout_marginTop="16dp"
        android:text="Item"
        android:textColor="#1874CD" />

    <TextView
        android:id="@+id/txtMesa"
        style="@style/CxTextos"
```

```
        android:layout_alignParentLeft="true"
        android:layout_below="@+id/textView5"
        android:layout_marginLeft="20dp"
        android:text="TextView" />

    <TextView
        android:id="@+id/textView3"
        style="@style/CxTextos"
        android:layout_alignParentLeft="true"
        android:layout_below="@+id/txtMesa"
        android:layout_marginLeft="20dp"
        android:layout_marginTop="18dp"
        android:text="Mesa"
        android:textColor="#1874CD" />

    <TextView
        android:id="@+id/txtItem"
        style="@style/CxTextos"
        android:layout_alignParentLeft="true"
        android:layout_below="@+id/textView3"
        android:layout_marginLeft="20dp"
        android:text="TextView" />

    <TextView
        android:id="@+id/textView7"
        style="@style/CxTextos"
        android:layout_alignParentLeft="true"
        android:layout_below="@+id/txtItem"
        android:layout_marginLeft="20dp"
        android:layout_marginTop="17dp"
        android:text="Quant"
        android:textColor="#1874CD" />

    <TextView
        android:id="@+id/txtQuantidade"
        style="@style/CxTextos"
        android:layout_alignParentLeft="true"
        android:layout_below="@+id/textView7"
        android:layout_marginLeft="20dp"
        android:text="TextView" />

    <TextView
        android:id="@+id/textView9"
        style="@style/CxTextos"
        android:layout_alignParentLeft="true"
        android:layout_below="@+id/txtQuantidade"
        android:layout_marginLeft="20dp"
        android:layout_marginTop="25dp"
        android:text="Preço"
        android:textColor="#1874CD" />

    <TextView
        android:id="@+id/txtPreco"
        style="@style/CxTextos"
        android:layout_alignParentLeft="true"
        android:layout_below="@+id/textView9"
        android:layout_marginLeft="20dp"
        android:text="TextView" />
```

```
    <TextView
        android:id="@+id/textView11"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/txtID"
        android:layout_below="@+id/txtPreco"
        android:layout_marginLeft="3dp"
        android:layout_marginTop="28dp"
        android:text="Total"
        android:textColor="#1874CD"
        android:textStyle="bold" />

    <TextView
        android:id="@+id/txtTotal"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/textView11"
        android:layout_below="@+id/textView11"
        android:text="TextView" />

</RelativeLayout>
```

Listar2

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >

    <ImageView
        android:id="@+id/imageView1"
        android:layout_width="50dp"
        android:layout_height="50dp"
        android:layout_alignParentLeft="true"
        android:layout_alignParentTop="true"
        android:layout_marginLeft="15dp"
        android:src="@drawable/logo" />

    <TextView
        android:id="@+id/textView1"
        style="@style/CxTextos"
        android:layout_alignParentLeft="true"
        android:layout_alignParentRight="true"
        android:layout_below="@+id/imageView1"
        android:layout_marginLeft="20dp"
        android:text="Pedido"
        android:textColor="#1874CD" />

    <TextView
        android:id="@+id/txtID"
        style="@style/CxTextos"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/textView1"
        android:layout_marginLeft="20dp"
        android:text="TextView" />

    <TextView
        android:id="@+id/textView5"
```

```
        style="@style/CxTextos"
        android:layout_alignParentLeft="true"
        android:layout_below="@+id/txtID"
        android:layout_marginLeft="20dp"
        android:layout_marginTop="16dp"
        android:text="Item"
        android:textColor="#1874CD" />

    <TextView
        android:id="@+id/txtMesa"
        style="@style/CxTextos"
        android:layout_alignParentLeft="true"
        android:layout_below="@+id/textView5"
        android:layout_marginLeft="20dp"
        android:text="TextView" />

    <TextView
        android:id="@+id/textView3"
        style="@style/CxTextos"
        android:layout_alignParentLeft="true"
        android:layout_below="@+id/txtMesa"
        android:layout_marginLeft="20dp"
        android:layout_marginTop="18dp"
        android:text="Mesa"
        android:textColor="#1874CD" />

    <TextView
        android:id="@+id/txtItem"
        style="@style/CxTextos"
        android:layout_alignParentLeft="true"
        android:layout_below="@+id/textView3"
        android:layout_marginLeft="20dp"
        android:text="TextView" />

    <TextView
        android:id="@+id/textView7"
        style="@style/CxTextos"
        android:layout_alignParentLeft="true"
        android:layout_below="@+id/txtItem"
        android:layout_marginLeft="20dp"
        android:layout_marginTop="17dp"
        android:text="Quant"
        android:textColor="#1874CD" />

    <TextView
        android:id="@+id/txtQuantidade"
        style="@style/CxTextos"
        android:layout_alignParentLeft="true"
        android:layout_below="@+id/textView7"
        android:layout_marginLeft="20dp"
        android:text="TextView" />

    <TextView
        android:id="@+id/textView9"
        style="@style/CxTextos"
        android:layout_alignParentLeft="true"
        android:layout_below="@+id/txtQuantidade"
        android:layout_marginLeft="20dp"
        android:layout_marginTop="25dp"
```

```
        android:text="Preço"
        android:textColor="#1874CD" />

    <TextView
        android:id="@+id/txtPreco"
        style="@style/CxTextos"
        android:layout_alignParentLeft="true"
        android:layout_below="@+id/textView9"
        android:layout_marginLeft="20dp"
        android:text="TextView" />

    <TextView
        android:id="@+id/textView11"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/txtID"
        android:layout_below="@+id/txtPreco"
        android:layout_marginLeft="3dp"
        android:layout_marginTop="28dp"
        android:text="Total"
        android:textColor="#1874CD"
        android:textStyle="bold" />

    <TextView
        android:id="@+id/txtTotal"
        style="@style/CxTextos"
        android:layout_alignParentLeft="true"
        android:layout_below="@+id/textView11"
        android:layout_marginLeft="20dp"
        android:text="TextView" />

    <Button
        android:id="@+id/Leiame"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignBaseline="@+id/textView11"
        android:layout_alignBottom="@+id/textView11"
        android:layout_centerHorizontal="true"
        android:text="Button" />

</RelativeLayout>
```

Login

```
<?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="@drawable/login"
    android:orientation="vertical" >

    <TextView
        android:layout_width="wrap_content"

        android:layout_gravity="center"
        android:layout_height="wrap_content"
        android:text="Usuário (E-mail):" />

    <EditText
```

```
            android:id="@+id/edtUsuario"
            android:layout_width="241dp"
            android:layout_height="wrap_content"
            android:ems="10"
            android:layout_marginLeft="30dp"
            android:inputType="textPersonName" >

            <requestFocus />
        </EditText>

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="center"
            android:text="@string/Senha" />

        <EditText
            android:id="@+id/edtSenha"
            android:layout_width="241dp"
            android:layout_height="wrap_content"
            android:layout_marginLeft="30dp"
            android:ems="10"
            android:inputType="textPassword" />

        <Button
            android:id="@+id/btnLogin"
            android:layout_width="241dp"
            android:layout_height="wrap_content"
            android:layout_marginLeft="30dp"
            android:onClick="LoginOnClick"
            android:text="@string/Login" />

        <TextView
            android:id="@+id/txtTexto"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="center"
            android:paddingTop="40px"
            android:text="Não tenho cadastro." />

        <Button
            android:id="@+id/Leiame"
            android:layout_width="239dp"
            android:layout_height="wrap_content"
            android:layout_marginLeft="30dp"
            android:onClick="InserirUsuarioOnClick"
            android:text="@string/Cadastrar" />

    </LinearLayout>
```

Pedido

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/Instrucao"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/logo_ped"
    android:orientation="vertical" >
```

```
<Button
    android:id="@+id/Leiame"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginLeft="30dp"
    android:layout_marginTop="40dp"
    android:text="Instruções" />

<TextView
    android:id="@+id/txvMesa"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginLeft="30dp"
    android:layout_marginTop="10dp"
    android:text="@string/Mesa" />

<EditText
    android:id="@+id/txtMesa"
    style="@style/CxTextos"
    android:layout_width="80dp"
    android:layout_marginLeft="30dp"
    android:layout_marginRight="30dp"
    android:ems="10"
    android:hint="mesa.."
    android:inputType="number" >

    <requestFocus />

</EditText>

<TextView
    android:id="@+id/txvItem"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginLeft="30dp"
    android:text="@string/Item" />




<AutoCompleteTextView
    android:id="@+id/txtItem"
    style="@style/CxTextos"
    android:layout_width="175dp"
    android:layout_height="wrap_content"
    android:layout_marginLeft="30dp"
    android:ems="10"
    android:hint="Produtos..."
    android:textColor="#000000"

    />

<TableRow
    android:id="@+id/tableRow6"
    android:layout_width="match_parent"
    android:layout_height="wrap_content" >
```

```
    <TextView
        android:id="@+id/txvQtde"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="30dp"
        android:textColorHighlight="#000000"
        android:text="@string/Quantidade" />
</TableRow>

<TableRow
    android:id="@+id/tableRow4"
    android:layout_width="match_parent"
    android:layout_height="wrap_content" >

    <EditText
        android:id="@+id/txtQuantidade"
        style="@style/CxTextos"
        android:layout_width="90dp"
        android:layout_height="wrap_content"
        android:ems="10"
        android:layout_marginLeft="30dp"
        android:hint="Quant.."
        android:inputType="number" />

</TableRow>

<TableRow
    android:id="@+id/tableRow5"
    android:layout_width="match_parent"
    android:layout_height="wrap_content" >
</TableRow>

<TableRow
    android:id="@+id/tableRow2"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:gravity="left" >

    <TextView
        android:id="@+id/txvPreco"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="130dp"
        android:text="@string/Preco" />

    <TextView
        android:id="@+id/txvTotal"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="60dp"
        android:text="@string/Total" />
</TableRow>

<TableRow
    android:id="@+id/tableRow1"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:gravity="left" >
```

```
    <Button
        style="@style/Botoes"
        android:id="@+id/btnAdiciona"
        android:layout_width="85dp"
        android:layout_height="wrap_content"
        android:layout_marginLeft="30dp"
        android:text="@string/Adiciona" />

    <EditText
        android:id="@+id/txtPreco"
        android:layout_width="85dp"
        android:layout_height="wrap_content"
        android:layout_gravity="left"
        android:layout_marginLeft="1dp"
        android:editable="false"
        android:ems="10"
         />

    <EditText
        android:id="@+id/txtTotal"
        android:layout_width="85dp"
        android:layout_height="wrap_content"
        android:layout_gravity="left"
        android:layout_marginLeft="3dp"
        android:editable="false"
        android:ems="10"
         />
</TableRow>

<TableRow
    android:id="@+id/tableRow3"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:gravity="left" >

    <Button
        style="@style/Botoes"
        android:id="@+id/btnEnviar"
        android:layout_width="85dp"
        android:layout_height="wrap_content"
        android:layout_marginLeft="30dp"
        android:text="@string/Enviar" />

    <Button
        style="@style/Botoes"
        android:id="@+id/btnApagar"
        android:layout_width="90dp"
        android:layout_height="wrap_content"
        android:text="@string/Apagar" />

    <Button
        style="@style/Botoes"
        android:id="@+id/btnListar"
        android:layout_width="85dp"
        android:layout_height="wrap_content"
        android:text="@string/Listar" />
</TableRow>
```

```
    <TableRow
        android:id="@+id/tableRow7"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" >

    </TableRow>

</LinearLayout>
```

Pedido2

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/garcom"
    android:orientation="vertical" >

    <Button
        android:id="@+id/btnListar"
        style="@style/Botoes"
        android:layout_width="125dp"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="117dp"
        android:text="@string/Listar"
        android:textColor="#000000" />

</RelativeLayout>
```

Pedido3

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/adm"
    android:orientation="vertical" >

    <EditText
        android:id="@+id/editText1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="88dp"
        android:ems="10" />

    <EditText
        android:id="@+id/editText2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/editText1"
        android:layout_below="@+id/editText1"
        android:layout_marginTop="18dp"
        android:ems="10" />

    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
```

```
        android:layout_height="wrap_content"
        android:layout_alignBottom="@+id/editText1"
        android:layout_centerHorizontal="true"
        android:layout_marginBottom="49dp"
        android:text="Usuário" />

    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/editText1"
        android:layout_centerHorizontal="true"
        android:text="Senha" />

    <Button
        android:id="@+id/Leiame"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/editText2"
        android:layout_alignRight="@+id/editText2"
        android:layout_below="@+id/editText2"
        android:text="Cadastrar Clientes" />

    <Button
        android:id="@+id/Button01"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/Leiame"
        android:layout_alignRight="@+id/Leiame"
        android:layout_below="@+id/Leiame"
        android:text="Cadastrar Produtos" />

    <Button
        android:id="@+id/Button02"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/Button01"
        android:layout_alignRight="@+id/Button03"
        android:layout_below="@+id/Button01"
        android:text="Cadastrar Garçom" />

    <Button
        android:id="@+id/Button03"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/Button01"
        android:layout_alignRight="@+id/Button01"
        android:layout_below="@+id/Button02"
        android:text="Faturamento" />

</RelativeLayout>
```

```
package com.example.br.bohemian;

import android.app.Activity;
import android.content.Context;
import android.content.Intent;
import android.database.Cursor;
```

```
import android.database.sqlite.SQLiteDatabase;
import android.os.Bundle;
import android.view.View;
import android.widget.EditText;
import android.widget.Toast;

public class Login extends Activity {
	@Override
	protected void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.login);

	}

	public void LoginOnClick(View v) {

		EditText txtUsuario = (EditText) findViewById(R.id.edtUsuario);
		EditText txtSenha = (EditText) findViewById(R.id.edtSenha);

		if (txtUsuario.getText().toString().length() <= 0) {
			txtUsuario.setError("Preenchimento obrigatório!");
			txtUsuario.requestFocus();
		} else if (txtSenha.getText().toString().length() <= 0) {
			txtSenha.setError("Preenchimento obrigatório!");
			txtSenha.requestFocus();
		} else {
			try {
				String usuario = txtUsuario.getText().toString();
				String senha = txtSenha.getText().toString();

				SQLiteDatabase db =
openOrCreateDatabase("pedido_clientes.db",
						Context.MODE_PRIVATE, null);

				Cursor c = db
						.rawQuery(
							"SELECT  email, senha,
permissao FROM usuarios WHERE email=?",
							new String[] {
String.valueOf(usuario) });

				if (c.moveToFirst()) {

					String busuario = c.getString(0);
					String bsenha = c.getString(1);
					int permissao = Integer.parseInt(c.getString(2));

					if (usuario.equals(busuario) &&
senha.equals(bsenha)) {
						if (permissao == 1) {
							Intent it = new
Intent(getBaseContext(),
									Pedido3.class);//
Abre a classe e instacia o

	// objeto do tipo Intent e
```

```
        // Chama a classe.
                                        startActivity(it);// inicializa a nova
classe "it"

                                    } else if (permissao == 2) {
                                        Intent it = new
Intent(getBaseContext(),
                                                Pedido2.class);//
Abre a classe e instacia o

        // objeto do tipo Intent e

        // Chama a classe.
                                        startActivity(it);// inicializa a nova
classe "it"

                                    } else if (permissao == 3) {
                                        Intent it = new
Intent(getBaseContext(),
                                                Pedido.class);// Abre
a classe e instacia o

        // objeto do tipo Intent e

        // Chama a classe.
                                        startActivity(it);// inicializa a nova
classe "it"

                                    }
                                    limpa();// Metodo para apagar campos

                                } else {
                                    Toast.makeText(getBaseContext(), "Senha
inválida!",

        Toast.LENGTH_SHORT).show();
                                }
                            } else {
                                Toast.makeText(getBaseContext(), "Usuário não
cadastrado!",
                                        Toast.LENGTH_SHORT).show();
                            }

                        } catch (Exception ex) {
                            Toast.makeText(getBaseContext(), ex.getMessage(),
                                    Toast.LENGTH_SHORT).show();
                        }
                    }

        }

        private void limpa() {
            EditText txtUsuario = (EditText) findViewById(R.id.edtUsuario);
            EditText txtSenha = (EditText) findViewById(R.id.edtSenha);
            txtUsuario.setText("");
```

```
            txtSenha.setText("");

        }

        public void InserirUsuarioOnClick(View v) {
            Intent it = new Intent(getBaseContext(), InserirUsuarioActivity.class);//
Abre
                                                                    // a
                                                                    // classe
                                                                    // e
                                                                    // instacia
                                                                    // o
                                                                    // objeto
                                                                    // do
                                                                    // tipo
                                                                    // Intent
                                                                    // e
                                                                    // Chama
                                                                    // a
                                                                    // classe.
            startActivity(it);// inicializa a nova classe "it"

        }

}
```