

Plano de Trabalho (Previsto)

O projeto tem como objetivo a criação de um sistema para cadastro de clientes, simulando um Banco (tal como o Itaú, porém somente para varejo e em menores proporções). O plano de trabalho tem como seguintes etapas:

- Modelagem da base de dados
- Criação de scripts e criação no SGBD
- Desenvolvimento da aplicação

Tecnologias a serem usadas:

- Para a modelagem da base de dados será utilizado o software DBMain
- O SGBD a ser utilizado será o Postgre SQL, por ser um software livre e já de assimilação e uso pelo desenvolvedor
- A IDE para o desenvolvimento do sistema será o NetBeans IDE 8.2, e a tecnologia na qual será desenvolvida será em Java, por maior domínio e facilidade do desenvolvedor.

A previsão da aplicação no Plano de Trabalho consiste em:

- Tela de login principal (usuário do banco ou cliente)
- Cliente somente poderá criar e cadastrar uma conta fornecendo seus dados e anexando um comprovante de residência
- Um usuário do banco (gerente de conta) cadastrado previamente (diretamente no banco de dados), poderá cadastrar, editar, procurar e excluir um determinado cliente e sua conta

Repositório do projeto se encontra aqui:

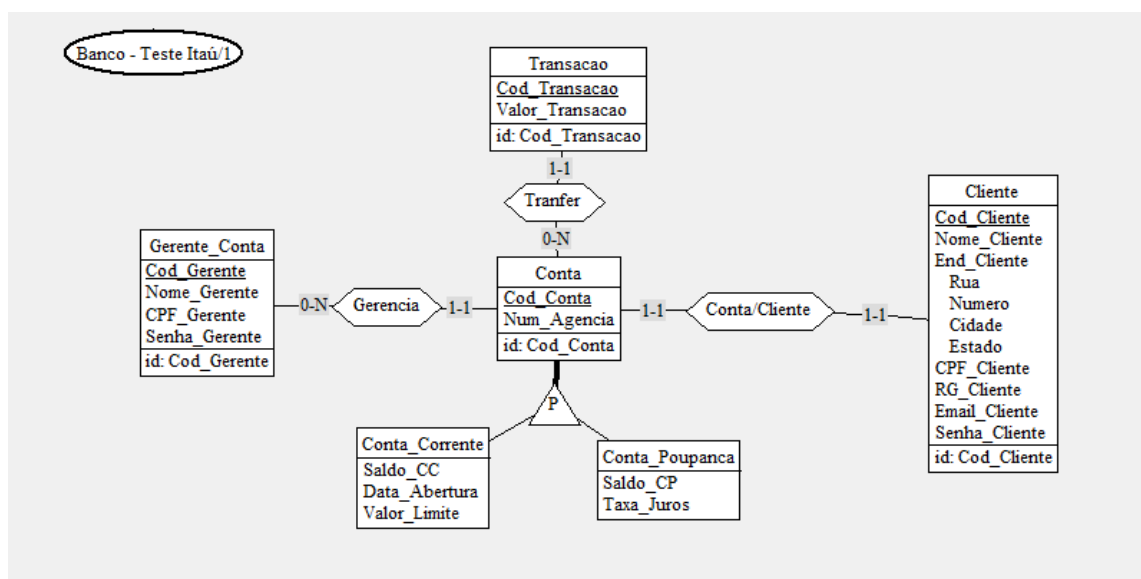
<https://github.com/victorbernardes/TesteItau.git>

Modelagem da Base de Dados

Para a modelagem da base de dados, foi utilizado o software DB-Main, na versão 9.1.2, na qual é uma ferramenta gratuita, e é possível modelar tanto o modelo Entidade-Relacionamento, como o modelo Relacional. Considero esse software mais didático (correspondendo melhor a metodologia formal de modelagem de Banco de Dados) do que por exemplo, o Erwin. Usei como referência o livro *Fundamentals of Database Systems (7th Edition)* (Elsmari & Navathe).

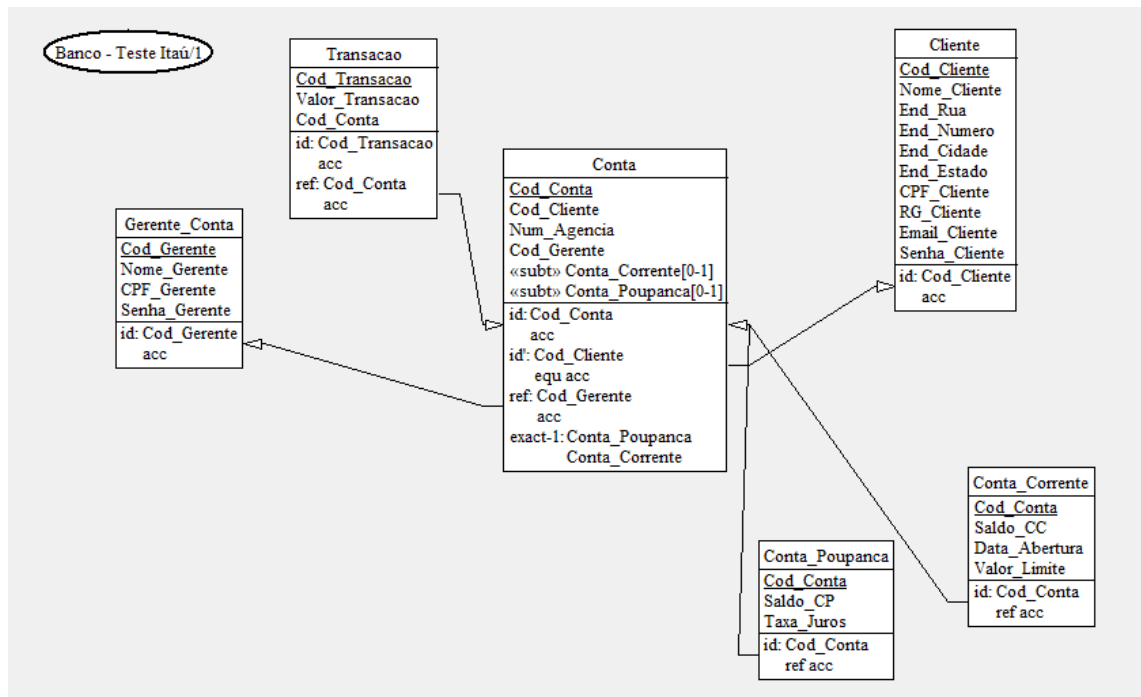
Modelo Entidade-Relacionamento

A modelagem da base de dados foi baseada na simulação de um simplificado funcionamento de um Banco. Com o foco no cadastro do cliente (assim tendo uma conta-corrente ou poupança, ou até ambas) tal como foi requerido.



Modelo Relacional

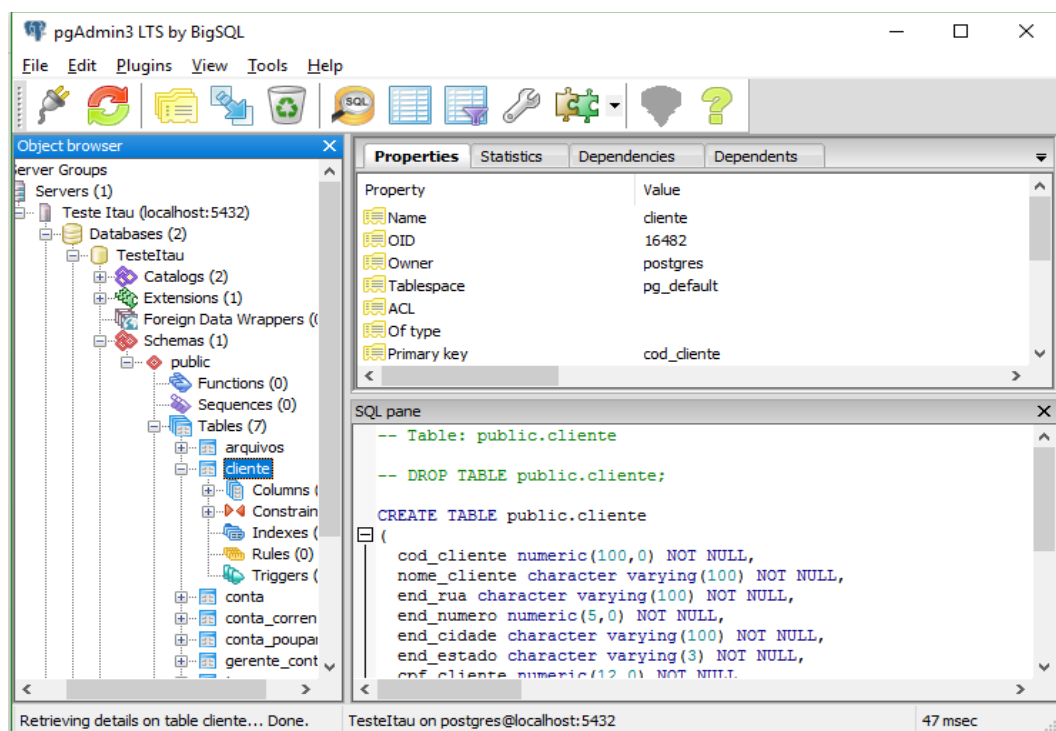
Seguindo a metodologia de modelagem de banco de dados, obtive o modelo relacional a partir do primeiro modelo entidade-relacionamento acima, já contando com a Normalização (por exemplo, endereço é um atributo multivalorado na qual se tornará 4 atributos diferentes).



Scripts SQL e criação no SGBD

Seguindo o modelo Relacional na imagem acima, foi gerado os scripts SQL disponíveis em meu perfil do GitHub ([link aqui](#)).

Utilizando desses scripts, foi criado o banco de dados no SGBD Postgre SQL, conforme imagem abaixo:



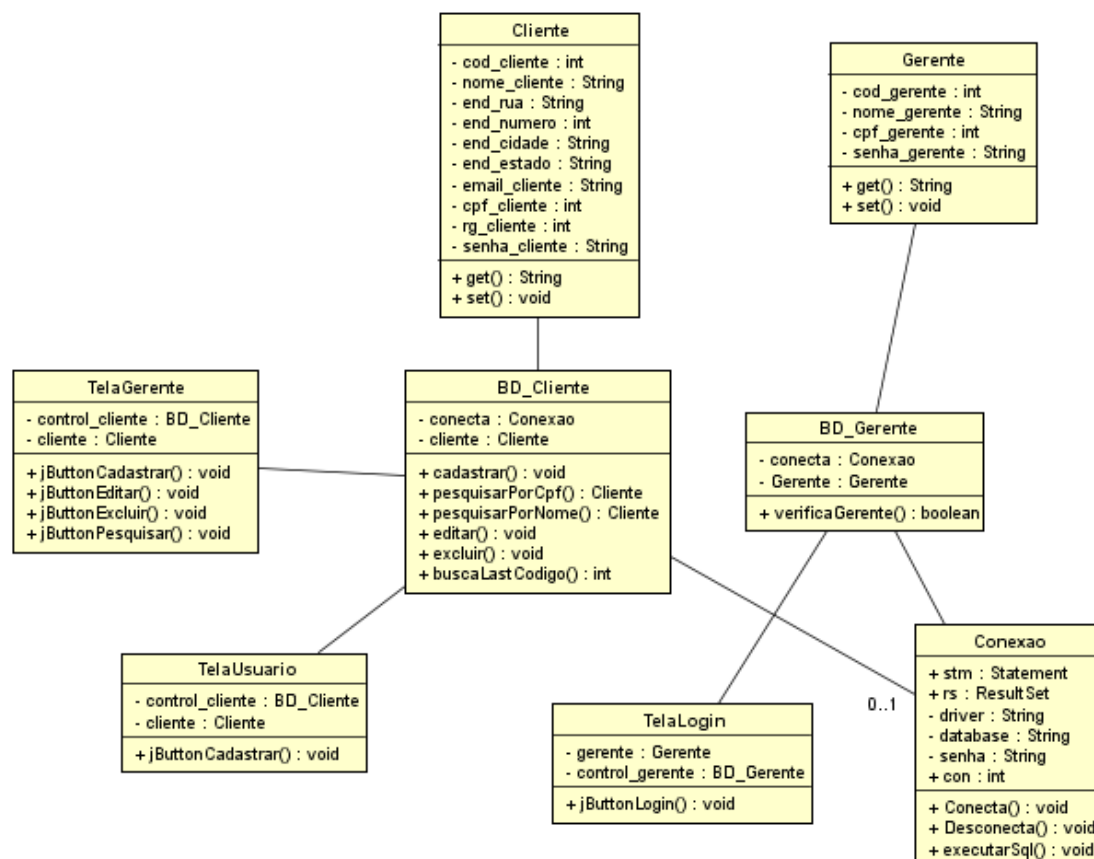
Obs1: Uma mudança no banco de dados ocorreu quando se compara com o modelo relacional; foi criada uma tabela "Arquivos" com duas colunas: cod_arquivo e arquivo. Esse atributo arquivo é do tipo Oid, na qual você gravar qualquer arquivo no banco de dados. Essa tabela será usada para armazenar os comprovantes de residência.

Desenvolvimento e Diagramas de Arquitetura ou UML

O desenvolvimento em Java, foi feito através da IDE Netbeans 8.2. Foi utilizado 3 pacotes distintos, na qual têm as seguintes funções:

- Conexão e troca de informações com o banco de dados (Postgre)
- Classes para tipos (Gerente, Cliente, etc)
- Pacote para interface gráfica

Sendo assim, obtive o seguinte Diagrama de Classe, ou arquitetura do sistema, sendo o mesmo desenhado na ferramenta Astah Community:



Capturas de Tela e maiores descrições:

A classe principal do projeto, e tela inicial ao executar o programa (exibida na imagem abaixo) tem a função de direcionar o usuário para a respectiva tarefa que ele gostaria de executar. Caso seja um novo cadastro (Pessoa Física ou Jurídica) não é necessário colocar Usuário e Senha, já caso queira logar como um Gerente de Conta Itaú, é necessário colocar o usuário (matrícula ou cod_gerente no banco de dados) e a sua senha. Foi utilizado os seguintes dados (cadastrado previamente no banco de dados):

- Usuário: 1
- Senha: admin123



Ao clicar em Pessoa Física/Jurídica e em seguida em Acessar, é direcionado para uma tela na qual o usuário pode se cadastrar (figura abaixo). Não é possível realizar qualquer outra operação, já que essa é uma tarefa para o Gerente de Conta. Também é possível o usuário anexar um comprovante de residência, na qual é colocado no banco de dados (tipo oid no Postgre), assim como voltar para a tela de login ou sair do programa.

Voltar Sair

Como usuário você pode cadastrar sua conta, deve-se preencher todas as informações e anexar um comprovante de residência:

Nome: Jose da Silva

Rua: Av Paulista Número: 222

Cidade: Sao Paulo Estado: SP

CPF: 143312412


Email: jose@itau.com.br

Cadastre uma senha (até 10 dígitos): *****

Anexar comprovante de residência: ...

Cadastrar

Mensagem

 Cadastro de cliente com sucesso!

OK

Na tela do gerente de contas (logado com usuário e senha) é possível cadastrar, editar um cadastro de cliente, excluir ou então pesquisar um determinado cliente (através de seu CPF ou nome).

Voltar Sair

Como Gerente de Conta, você pode cadastrar, pesquisar, editar ou excluir Clientes.

Nome:

Rua: Número:

Cidade: Estado:

CPF: RG:

Email:


Senha:

Anexar comprovante de residência: ...

Cadastrar Pesquisar * Editar Excluir

* Pesquisar por CPF ou Nome

Mensagem

 Login de Gerente com sucesso!

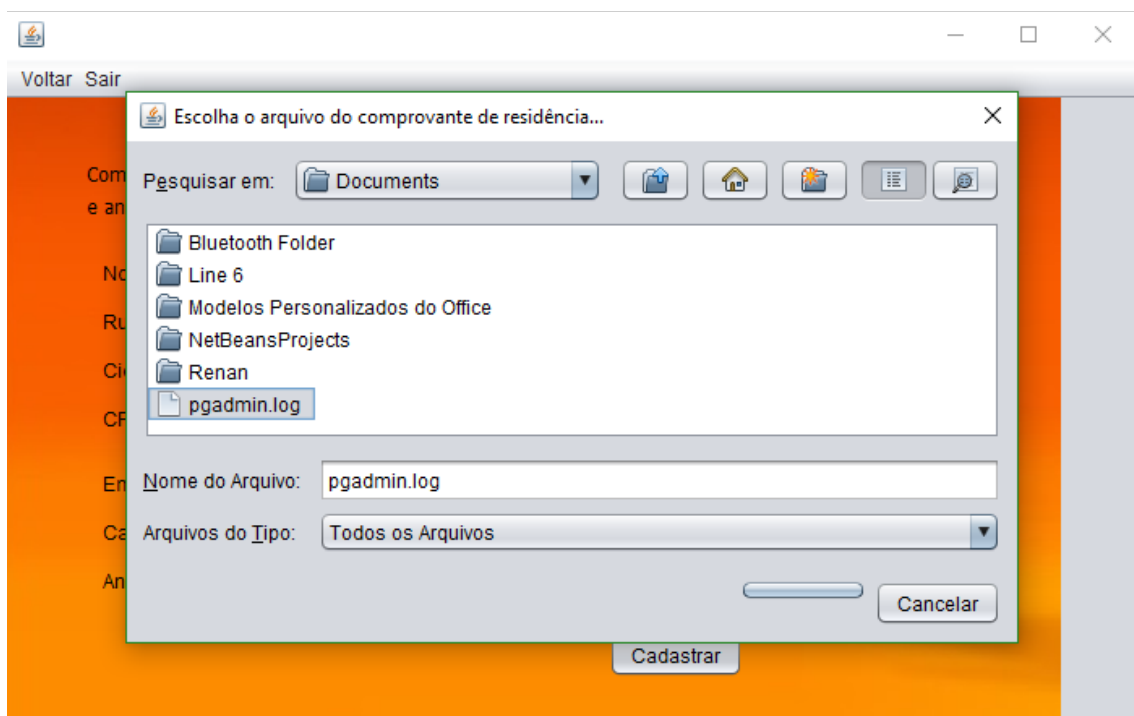
OK

Seleção arquivo de comprovante de residência

Como descrito na seção sobre a modelagem de dados, foi criada uma tabela para armazenar arquivos. Na parte de desenvolvimento foi criado um `JFileChooser`, na qual é chamado através do botão "...". Quando esse botão é acionado, abre uma caixa de seleção de arquivos, e após o arquivo ser selecionado pelo usuário, o diretório dele é gravado em uma `String`, e também é verificado o último `cod_arquivo` do banco de dados, para fazer o código incremental.

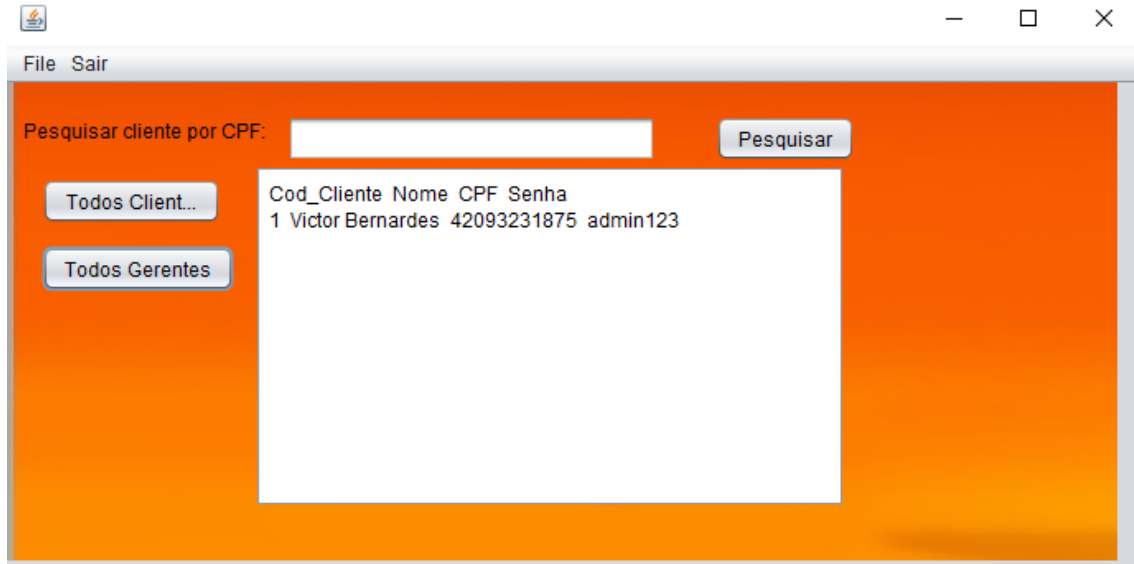
```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {  
    int returnVal = jFileChooser1.showOpenDialog(this);  
    if (returnVal == JFileChooser.APPROVE_OPTION) {  
        File file = jFileChooser1.getSelectedFile();  
  
        String diretorioArquivo = file.getParent() + "/" + file.getName();  
        // What to do with the file, e.g. display it in a TextArea  
        Conexao conectar = new Conexao();  
        int cod_arquivo = conectar.buscaLastCodigo() + 1;  
        String sql = "insert into arquivos values (" + cod_arquivo + ", lo_import('" + diretorioArquivo + "'))";  
        conectar.adicionaArquivo(sql);  
    } else {  
        System.out.println("Acesso cancelado pelo usuário.");  
    }  
}
```

Então é chamado o método `adicionaArquivo`, passando como parâmetro esse script SQL (com o uso do diretório), na qual grava o arquivo no banco de dados. A imagem de um teste pode ser vista abaixo:



Adendos

Foi adicionado uma página extra de “Pesquisa Avançada” na aba da tela do Gerente, na qual é possível além de pesquisar por CPF, pesquisar todos os clientes ou gerentes. (Conforme figura abaixo)



Pensando na continuidade desse projeto, essa área seria responsável por pesquisar SQL mais avançados.

Testes

Como o software desenvolvido nesse curto período de tempo, não seria um sistema muito complexo, foram realizados testes utilizando a técnica da caixa branca, principalmente nos níveis de teste de unidade e integração, como por exemplo:

- Testes com parâmetros incorretos
- Testes simulando erro do usuário (login)
- Testes com informações parecidas no banco de dados (pesquisar

Plano de Trabalho (realizado)

Não houve desvios na execução do sistema, pois o planejamento foi realizado pensando já em todas as etapas do ciclo de vida desse projeto, começando pela modelagem do banco de dados, passando pela modelagem do sistema, e por último a execução. Também foi de grande ajuda os conhecimentos prévios e certo domínio na linguagem de programação na qual foi desenvolvido (Java) e também sobre as metodologias de modelagem de dados e Engenharia de Software.

O projeto poderia se desenvolver mais, caso continuasse (com mais tempo hábil), como por exemplo a vinculação de uma conta corrente ou poupança para o cliente já cadastrado e simulação de internet banking (o usuário verificando o saldo de sua conta).