**Processo Seletivo – Itaú Unibanco**

**Relatório Técnico**

Camila Souza

24 de Janeiro de 2017

**Índice**

**1. Plano de trabalho**3

1.1 Previsto3

1.2 Realizado3

**2. Tecnologias utilizadas4**

**3. Diagrama de arquitetura4**

**4. Modelagem da base de dados5**

**5. Github5**

**6. Referências utilizadas5**

**7. Técnica de testes5**

7.1 Testes funcionais6

7.1.1 Ferramentas6

7.2 Teste unitário6

7.2.1 Ferramentas6

**1. Plano de Trabalho**

1.1 Previsto

Iniciar o planejamento da aplicação com um profissional de UX (*User Experience*) para esquematizar as telas de uma tal forma mais amigável, com uma forma simples e bem estruturada.

A aplicação deverá ser desenvolvida com uma arquitetura de alta disponibilidade e escalabilidade para nunca sair do ar e que, se houver um ou mil clientes, a aplicação deve se comportar da mesma forma.

A aplicação deverá conter uma camada de segurança, um controle de acesso de autenticação e autorização, onde o perfil de cliente terá acesso nas funcionalidades de inclusão, consulta e edição das informações e o perfil de gerente terá todas as funcionalidades do cliente mais a de exclusão.

O cliente poderá abrir várias contas vinculadas à ele e/ou o cliente poderá abrir uma conta conjunta vinculada a uma conta corrente. Na inclusão deverá ser disponibilizado os campos de informações pessoais e possibilitar que o cliente envie uma imagem de comprovante de residência.

No momento da inclusão e edição das informações do cliente, deve ser feito a validação dos campos obrigatórios para poder realizar a gravação dos dados.

Após a aplicação ser desenvolvida, deverá ser realizado *unit tests* e testes funcionais para garantir que a aplicação está atendendo corretamente o que foi solicitado.

1.2 Realizado

A aplicação foi desenvolvida focando em atender no mínimo as 4 funcionalidades de incluir, editar, excluir e consultar.

A aplicação foi desenvolvida com uma arquitetura web em 3 camadas:

1. Visual: As telas da aplicação;
2. Negócio: As regras de negócio;
3. Dados: Conexão com o banco de dados.

Foi desenvolvida uma aplicação simples com poucas informações de dados pessoais, não sendo possível a inclusão de imagem de comprovante de residência. O cliente também só pode ter vinculado apenas uma conta a ele.

Pelo curto prazo que tive para o desenvolvimento e não ter muita habilidade para o desenvolvimento de uma aplicação inteira (do visual até aos dados), não foi possível desenvolver conforme foi previsto.

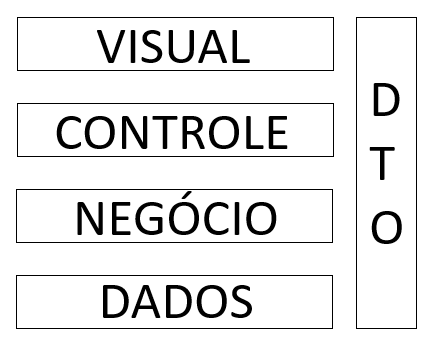
**2. Tecnologias Utilizadas**

A arquitetura foi desenhada para web em 3 camadas (Visual, Negócio e Dados).

Foi utilizado a linguagem Java – versão JDK 7.0 e IDE Eclipe versão Mars.

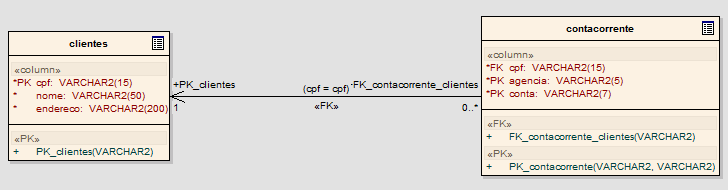
A aplicação suporta qualquer sistema operacional.

**3. Diagrama de Arquitetura**



1. Visual: São as telas da aplicação (HTML)
2. Controle: Direciona uma requisição depois que ela é processada (Servlet)
3. Negócio: Regras de negócio (BO)
4. Dados: As informações no banco de dados (DAO)
5. DTO: Objeto de transferência entre as camadas

**4. Modelagem da base de dados**



**5. Github**

Perfil: Camila-2017

Repositório compartilhado: ContaOnlineItau

Link: <https://github.com/Camila-2017/ContaOnlineItau>

**6. Referencias utilizadas**

<http://www.ic.unicamp.br/~cortes/mc726/cap3.pdf>

<http://www.linhadecodigo.com.br/artigo/1444/quais-sao-as-reais-caracteristicas-da-qualidade-da-nbr-iso_iec-9126-1.aspx>

<http://www.devmedia.com.br/testes-de-unidade-com-junit/4637>

<http://testesdesoftware.com/testes-funcionais/>

<http://www.seleniumhq.org/>

<https://www.caelum.com.br/apostila-java-web/servlets/>

**7. Técnicas de teste**

7.1 Testes Funcionais

Também conhecido como teste da caixa-preta, responsável em verificar se a aplicação está atendendo todas as funcionalidades conforme foram solicitadas. Avalia o comportamento, confiabilidade, eficiência, usabilidade e o design da tela.

7.1.1.Ferramentas

Selenium: é uma ferramenta de automatização de testes funcionais, podendo criar tarefas repetitivas em uma página web.

7.2 Testes Unitários

Responsável em avaliar os módulos separadamente para encontrar possíveis problemas e/ou erros para que a aplicação não venha apresentar problemas ou baixo desempenho.

7.2.1.Ferramentas

Junit: é um *framework* que facilita a criação de código para a automatização de testes com apresentação de resultados.