**Orientadora: Zirlene Effgen**

**Disciplinas Envolvidas:**

* Banco de Dados II (Cledson Malaquias)
* Análise e Projeto de Sistemas II (André Ribeiro)
* Programação Avançada de Sistemas (André Ribeiro)

# Anexo PI-4-01 - Tema e Escopo do Sistema

|  |  |  |  |
| --- | --- | --- | --- |
| **Data** | **Descrição da(s) Mudança(s) Ocorrida(s)** | **Autor** | **Versão** |
| 20/02/2016 | Documento Original – PI-3 | Bruno | 1.0 |
| 09/09/2016 | Atualização do Documento | Rafael | 1.1 |
| 25/11/2016 | Atualização do Documento | Rafael | 1.2 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Planejamento:** | **Responsabilidades** | | | |
| **Atividades / Componentes** | **Bruno** | **Everaldo** | **Rafael Rufino** | **Walter** |
| Definição Tema | X | X | X | X |
| Definição Escopo | X | X | X | X |
| Revisão Escopo | X |  | X |  |
|  |  |  |  |  |
| *Marque com um x indicando o(s) aluno(s) responsável(eis) por cada atividade.* | | | | |
|  |  |  |  |  |
| **Execução:** | **Cumprimento das Atividades** | | | |
| **Atividades / Componentes** | **Bruno** | **Everaldo** | **Rafael Rufino** | **Walter** |
| Definição Tema | X | X | X | X |
| Definição Escopo | X | X | X |  |
| Revisão Escopo | X |  | X |  |
|  |  |  |  |  |
| *Indique se aluno cumpriu a atividade (Sim, Não, Em parte)* | | | | |
| **Fator de Participação ==>** |  |  |  |  |
| *- Com base nas respostas acima defina o fator de participação do aluno para esta fase.*  *- O fator de participação obtido pelo aluno será multiplicado pela nota obtida na fase compondo assim a nota do aluno naquela fase.* | | | | |

**1 – INTRODUÇÃO**

**1.1 Tema**

Invista Já, Educação Financeira.

**1.2 Escopo**

O Projeto tem a proposta fornecer um software capaz de proporcionar ao usuário um melhor entendimento de suas características econômicas, podendo ser acessado através de plataformas WEB e Mobile.

O sistema irá coletar perguntas e respostas de seus usuários, visando fazer uma mineração e tratamento dos dados para conseguir classificar o perfil econômico. Por meio desse perfil o software vai conseguir trazer ao seu usuário as melhores opções de investimento, com simulações de resultado facil entendimento.

A princípio não será feito investimentos reais, apenas simulações do mesmo.

# Anexo PI-4-02 - Análise Orientada a Objetos

|  |  |  |  |
| --- | --- | --- | --- |
| **Data** | **Descrição da(s) Mudança(s) Ocorrida(s)** | **Autor** | **Versão** |
| 20/02/2016 | Documento Original – PI-3 | Bruno | 1.0 |
| 09/09/2016 | Alteração de Documento | Rafael | 1.1 |
| 12/09/2016 | Especificação dos casos de uso e protótipos incluídos. | Rafael | 1.2 |
| 13/09/2016 | Descrição geral incluída | Bruno | 1.3 |
| 14/09/2016 | Perspectiva do produto e diagrama de classe incluídos | Bruno | 1.4 |
| 21/09/2016 | Diagramas de sequência e estados adicionados | Bruno | 1.5 |
| 15/09/2016 | Requisitos funcionais atualizados | Bruno | 1.6 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Planejamento:** | **Responsabilidades** | | | |
| **Atividades / Componentes** | **Bruno** | **Everaldo** | **Rafael Rufino** | **Walter** |
| Descrição do Projeto | X |  | X |  |
| Levantamento de Requisitos | X |  |  |  |
| Diagramas UC |  | X |  |  |
| Especificação de UC |  | X |  |  |
| Diagramas de Classe |  |  | X |  |
| Diagramas de Sequência |  |  | X |  |
| Diagramas de Estado |  |  | X |  |
| *Marque com um x indicando o(s) aluno(s) responsável(eis) por cada atividade.* | | | | |
|  |  |  |  |  |
| **Execução:** | **Cumprimento das Atividades** | | | |
| **Atividades / Componentes** | **Bruno** | **Everaldo** | **Rafael Rufino** | **Walter** |
| Descrição do Projeto | X |  | X |  |
| Levantamento de Requisitos | X |  |  |  |
| Diagramas UC |  | X |  |  |
| Especificação de UC |  | X |  |  |
| Diagramas de Classe |  |  | X |  |
| Diagramas de Sequência |  |  | X |  |
| Diagramas de Estado |  |  | X |  |
| *Indique se aluno cumpriu a atividade (Sim, Não, Em parte)* | | | | |
| **Fator de Participação ==>** |  |  |  |  |
| *- Com base nas respostas acima defina o fator de participação do aluno para esta fase.*  *- O fator de participação obtido pelo aluno será multiplicado pela nota obtida na fase compondo assim a nota do aluno naquela fase.* | | | | |

**2 – DESCRIÇÃO GERAL DO PRODUTO**

O Invista Já é um sistema multiplataforma que visa analisar o perfil de usuário, buscando identificar suas características econômicas.

Após a identificação o sistema indicará algumas opções de investimento e simulará rendimentos na opção escolhida

**2.1 Perspectiva do Produto**

Observando a sociedade, pode-se constatar que uma parcela relativa da população tem pouco ou quase nenhum conhecimento ou controle financeiro. Sabendo dessa realidade, surge em nosso meio vários sistema que visam tirar proveito dessa vulnerabilidade.

É quase que constantemente noticiado pessoas caindo em golpes de estelionatários, em pirâmides, realmente achando que vão obter algum lucro ou que estão investindo em algo seguro. Ou acabam fazendo investimentos em aplicações nada rentáveis, ou falsos investimentos, que não chegam a cobrir a desvalorização do dinheiro obtendo apenas ganhos ilusórios.

Com base nessa realidade, decidimos em criar um software capaz de ajudar pessoas a terem uma melhor percepção do que é um real investimento, e que poupar dinheiro pode trazer grandes benefícios futuros.

Isso tudo será alcançado através de um estudo de personalidade, que busca traçar o perfil de uma pessoa, para que assim o sistema consiga saber em quais condições de investimento esse usuário se adaptaria melhor.

O “Invista Já” utiliza-se de questionários para estudar a personalidade do usuário. As respostas fornecidas nos questionários são pontuadas e essa pontuação fornece indicadores para a classificação do usuário em um perfil financeiro. Com o perfil classificado, o sistema permite que o usuário faça inúmeras simulações de investimentos e aplicações financeiras, obtendo diversos indicadores que auxiliarão o usuário em tomadas de decisões agindo como vetor de sustentabilidade social

O sistema deve seguir alguns requisitos, a saber:

* O sistema deve permitir a manutenção do cadastro de usuários.
* O sistema deve permitir a manutenção de perguntas e respostas.
* O sistema deve permitir a interação dos usuários com as questões de rastreamento de Persona.
* O sistema deve permitir a exibição de simulações de acordo com o perfil.

**2.2 Requisitos do Sistema**

**Requisitos Funcionais:**

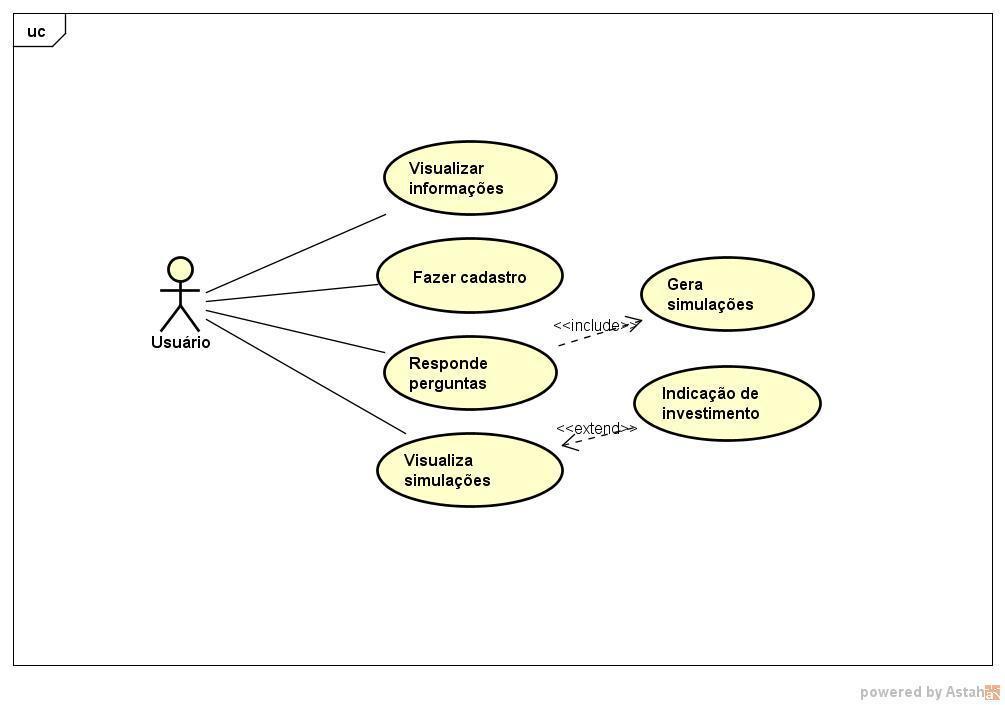
|  |  |  |
| --- | --- | --- |
| Número | Nome | Prioridade |
| RF01 | O sistema deve permitir a visualização de informações sobre investimento | 2 |
| RF02 | O sistema deve permitir o cadastro do usuário | 3 |
| RF03 | O sistema deve permitir o usuário visualizar e responder as questões para traçar o perfil do mesmo. | 1 |
| RF04 | O sistema deve permitir deve traçar um perfil do usuário | 2 |
| RF05 | O sistema deve fazer indicações ao usuário de acordo com o seu perfil | 1 |
| RF06 | O sistema deve permitir ao usuário fazer simulações de investimentos | 1 |
| RF07 | O sistema deve permitir ao administrador o cadastro de perguntas | 2 |

**Requisitos Não Funcionais:**

|  |  |  |
| --- | --- | --- |
| Número | Nome | Prioridade |
| RNF01 | O design do sistema deve ser responsivo, ou seja, adaptável para diferentes dispositivos com foco na mobilidade. | 3 |
| RNF02 | O sistema deve possuir um modulo web | 1 |
| RNF03 | Embora não seja um requisito essencial ao sistema, deve ser considerada por corresponder a um fator de qualidade de software. | 3 |
| RNF04 | O sistema deve persistir e recuperar os dados em banco de dados Oracle. | 1 |
| RNF08 | O sistema deve possuir um módulo Android | 3 |

**2.2.1 Casos de Uso**

|  |  |
| --- | --- |
| **Ator** | **Descrição** |
| Usuário | Responsável pela manutenção das informações no sistema e realizará o cadastro, consultas, etc. |



**2.2.1 Especificação de Casos de Uso**

**UC01 – Visualizar informações**

O caso de uso descreve a visualização das informações sobre investimentos, dicas financeiras.

***Fluxo:* Visualizar informações**

|  |  |
| --- | --- |
| ***Ator: Usuário*** | ***Sistema*** |
| 1 – O usuário acessa o site www.invistaja.com. |  |
|  | 2 – O sistema apresenta uma série de informações sobre investimento |

**UC02 – Fazer cadastro**

O caso de uso descreve o cadastro do usuário no sistema.

***Fluxo: Fazer Cadastro***

|  |  |
| --- | --- |
| ***Ator: Usuário*** | ***Sistema*** |
| 1 – O usuário acessa fazer simulação |  |
|  | 2 – O sistema apresenta a tela de **cadastro de usuário** com os seguintes componentes:   * Nome * Email * Senha * Telefone |
| 3 – O usuário clica no botão **salvar**. |  |
|  | 4 – O sistema então salva os dados do usuário. |
|  | 5 – O caso de uso é finalizado. |

**Regras de Negócio:**

RN02 – A informação do telefone deve ser opcional.

**UC03 – Responder perguntas**

O caso de uso descreve a apresentação das perguntas ao usuário para que se possa traçar um perfil de investimento.

***Fluxo: Responder perguntas***

|  |  |
| --- | --- |
| ***Ator: Usuário*** | ***Sistema*** |
| 1 – Após a realização do cadastro, o usuário visualizar uma série de perguntas. |  |
| 2 – O usuário irá responder as perguntas apresentadas. |  |
|  | 3 – O sistema irá gerar perguntas de acordo com as respostas do usuário. |
| 4 – O usuário então clica em gerar simulações. |  |
|  | 8 – O caso de uso é finalizado. |

**UC04 – Gerar simulações**

Este caso de uso permite o sistema gerar simulações de acordo com as perguntas respondidas.

***Fluxo: Gerar simulações***

|  |  |
| --- | --- |
| ***Ator: Usuário*** | ***Sistema*** |
|  | 1 – Quando o usuário clicar em gerar simulações o sistema irá guardar as informações no banco de dados. |
|  | 2 – O sistema irá pegar as informações e oferecer algumas opções ao usuário para investimento. |
|  | 3 – O caso de uso é finalizado. |

**UC05 – Visualizar simulações**

Este caso de uso permite ao usuário a possibilidade visualizar as informações e realizar a análise das informações geradas.

***Fluxo: Visualizar simulações***

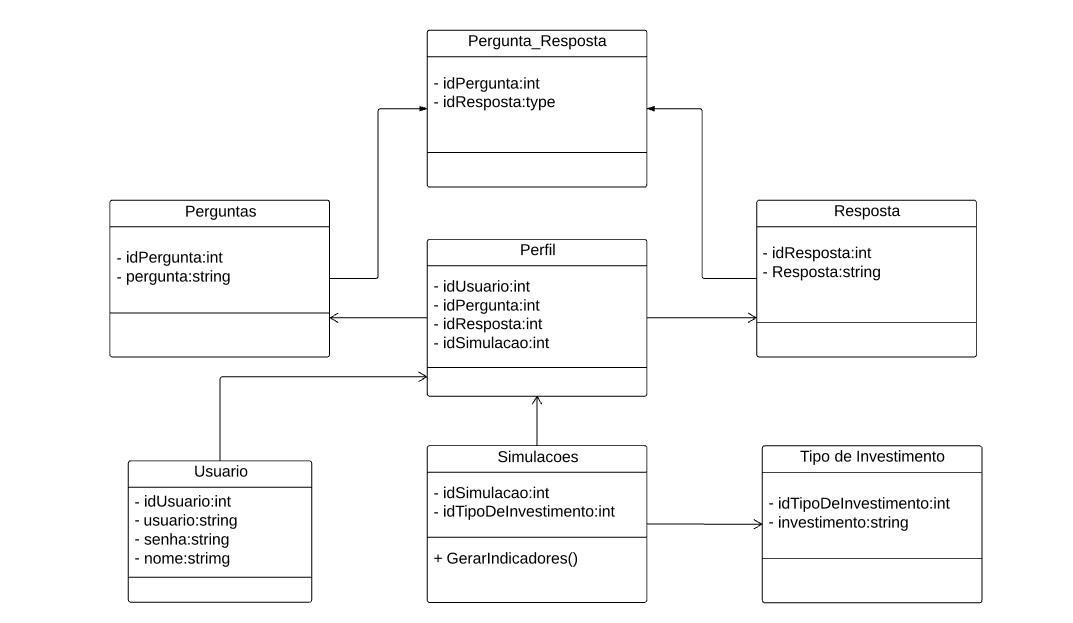
|  |  |
| --- | --- |
| ***Ator: Usuário*** | ***Sistema*** |
| 1 – Após a geração das simulações, o usuário irá visualizar as informações com dicas para investimento. |  |
|  | 2 – O sistema apresenta a tela **Invista já** com os seguintes componentes:   * Melhores opções para você * Escolha o melhor investimento |
| 3 – O usuário seleciona a opção desejada. |  |
|  | 4 – O sistema apresenta as informações que o usuário selecionou. |
|  | 5 - O caso de uso é finalizado. |

**UC06 – Indicação de investimento**

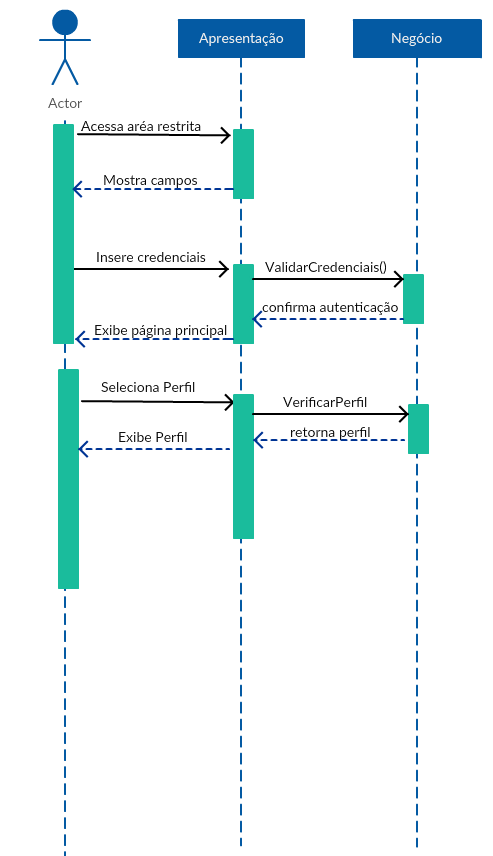
Este caso de uso permite ao usuário a possibilidade visualizar as informações e realizar a análise das informações geradas.

***Fluxo:* Indicação de investimento**

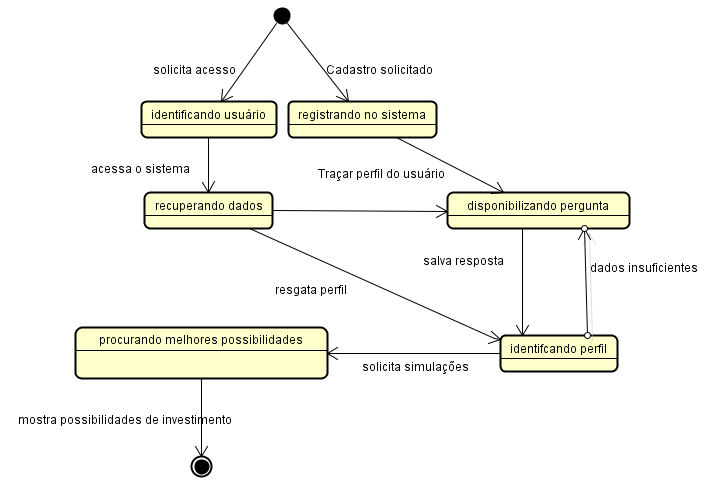
|  |  |
| --- | --- |
| ***Ator: Usuário*** | ***Sistema*** |
|  | 1 – Após a geração das simulações, o usuário irá visualizar as informações com dicas para investimento. |
|  | 2 – O sistema apresenta a tela **Invista já** com os seguintes componentes:   * Melhores opções para você * Escolha o melhor investimento |
| 3 – O usuário seleciona a opção desejada. |  |
|  | 4 – O sistema apresenta as informações que o usuário selecionou. |
|  | 5 - O caso de uso é finalizado. |

**2.2.2 Diagramas de classe de análise**

**2.2.2 Diagramas de sequência**



**2.2.2 Diagramas de Estados**



# Anexo PI-4-03 - Projeto Orientado a Objetos

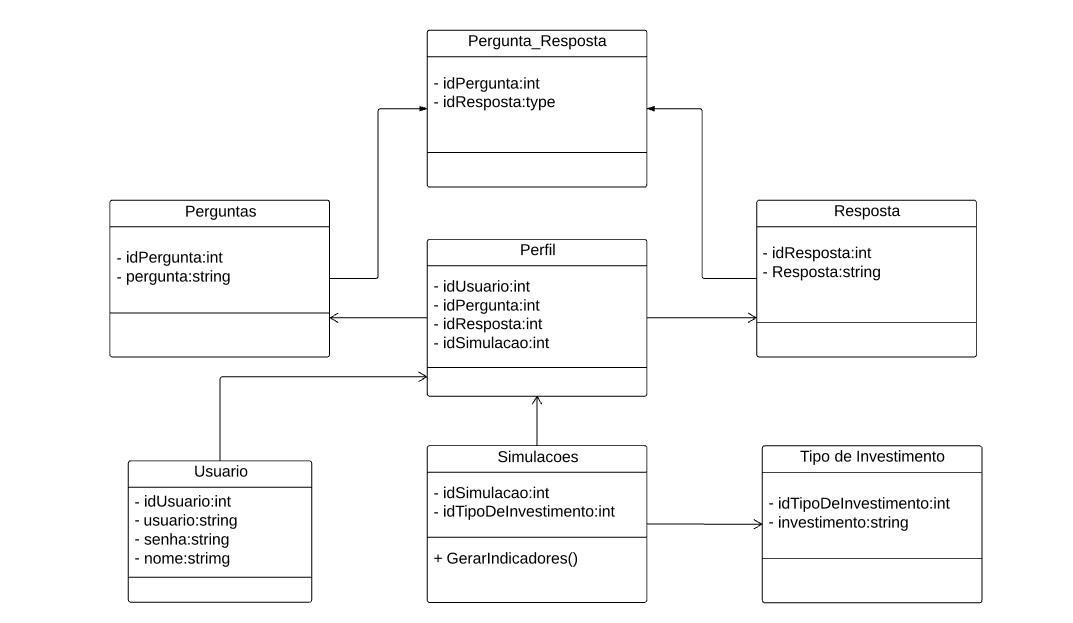
|  |  |  |  |
| --- | --- | --- | --- |
| **Data** | **Descrição da(s) Mudança(s) Ocorrida(s)** | **Autor** | **Versão** |
| 20/02/2016 | Documento Original – PI-3 | Bruno | 1.0 |
| 02/02/2016 | Alteração do Documento | Rafael | 1.1 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Planejamento:** | **Responsabilidades** | | | |
| **Atividades / Componentes** | **Bruno** | **Everaldo** | **Rafael Rufino** | **Walter** |
| Descrição do Projeto | X | X | X |  |
| Projeto de Classes |  |  | X |  |
| Projeto de Arquitetura |  |  | X |  |
| Projeto de componentes |  |  | X |  |
| Implantação do Sistema |  |  | X |  |
| *Marque com um x indicando o(s) aluno(s) responsável(eis) por cada atividade.* | | | | |
|  |  |  |  |  |
| **Execução:** | **Cumprimento das Atividades** | | | |
| **Atividades / Componentes** | **Bruno** | **Everaldo** | **Rafael Rufino** | **Walter** |
| Descrição do Projeto | X | X | X |  |
| Projeto de Classes |  |  | X |  |
| Projeto de Arquitetura |  |  | X |  |
| Projeto de componentes |  |  | X |  |
| Implantação do Sistema |  |  | X |  |
| *Indique se aluno cumpriu a atividade (Sim, Não, Em parte)* | | | | |
| **Fator de Participação ==>** |  |  |  |  |
| *- Com base nas respostas acima defina o fator de participação do aluno para esta fase.*  *- O fator de participação obtido pelo aluno será multiplicado pela nota obtida na fase compondo assim a nota do aluno naquela fase.* | | | | |

**3 – DESCRIÇÃO GERAL DO PROJETO DO PRODUTO**

*Para garantir a manutenibilidade e confiabilidades dos dados o sistema será dividido em camadas utilizando o padrão MVC.*

**3.1 Projeto de Classes/Dados**



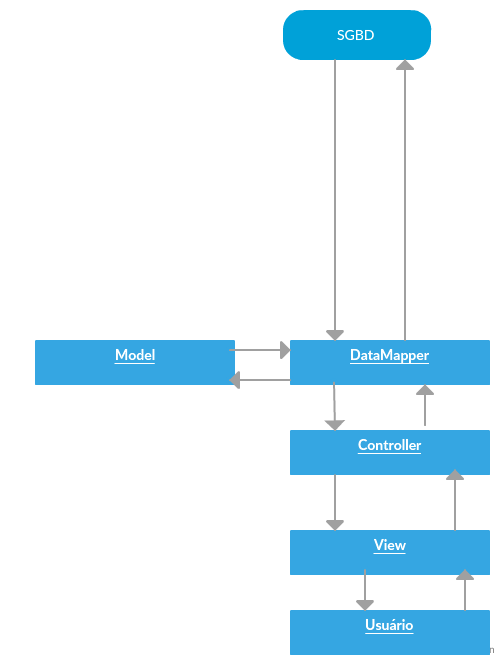
**3.2 Projeto de Arquitetura do Sistema**

*O projeto será estruturado no padrão MVC, o separando em camadas de apresentação e regras de negócio categorizadas como:*

*View: Responsável pela exibição de dados para o usuário usuário.*

*Controller: Responsável pelo controle e manipulação de dados.*

*Model: Responsável pela definição e comportamento dos modelos.*

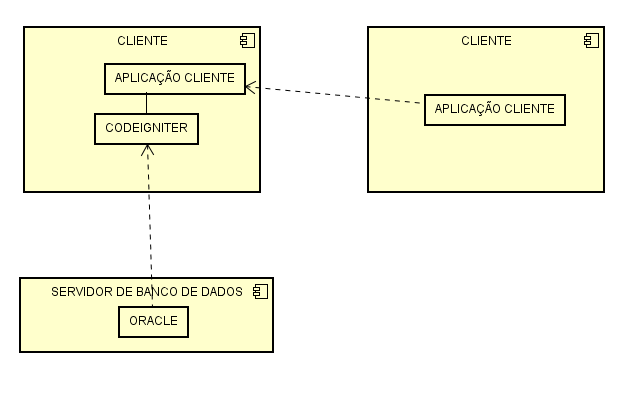


**3.3 Projeto de Componentes**

*O projeto será composto por dois componentes principais, as interfaces responsáveis pela exibição e coletada de dados e banco de dados responsável pela persistência dos dados.*

**3.4 Implantação do Sistema**

*O sistema será implementado utilizando um servidor Web responsável pelo armazenamento e execução da aplicação PHP. Para a persistência de dados a aplicação utiliza o componente CodeIgint Framework como forma de interface entre a aplicação PHP e o banco de dados Oracle.*



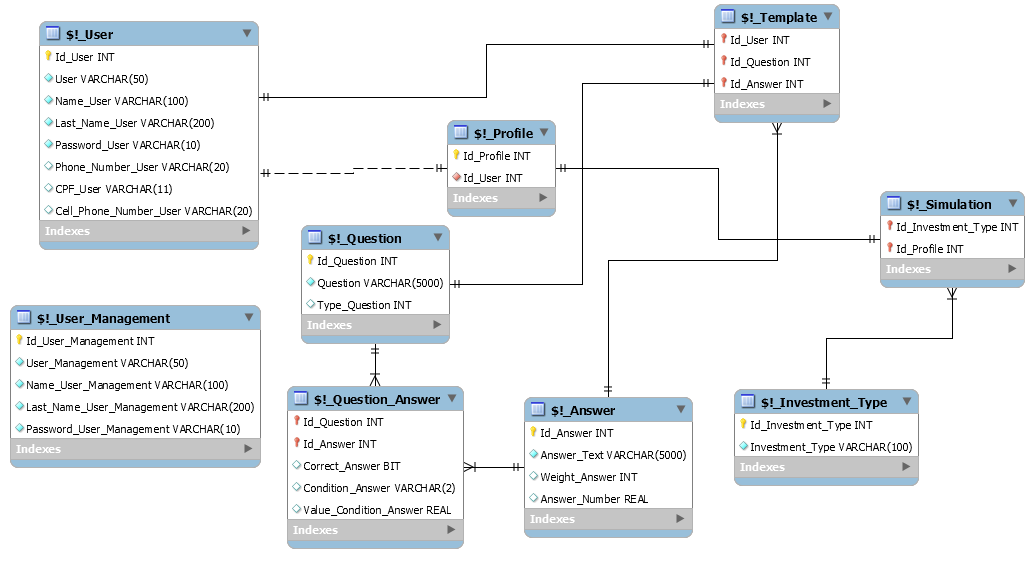
# Anexo PI-4-03 - Banco de Dados

|  |  |  |  |
| --- | --- | --- | --- |
| **Data** | **Descrição da(s) Mudança(s) Ocorrida(s)** | **Autor** | **Versão** |
| 10/09/2016 | Modelagem Conceitual | Rafael | 1.0 |
|  |  |  |  |
|  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Planejamento:** | **Responsabilidades** | | | |
| **Atividades / Componentes** | **Bruno** | **Everaldo** | **Rafael Rufino** | **Walter** |
| 4.1 Modelagem Conceitual | X | X | X | X |
| 4.2 Dicionário de Dados |  |  | X |  |
| 4.3 Política de Segurança |  |  |  | X |
| 4.4 Criação de Tablespaces (Dados, Temporária e Índices) |  | X |  |  |
| 4.5 Criação de usuários |  |  |  | X |
| 4.6 Script para criação do Modelo de Dados |  |  |  | X |
| 4.7 Queryes |  | X |  |  |
| 4.8 Modularização |  |  | X |  |
| *Marque com um x indicando o(s) aluno(s) responsável(eis) por cada atividade.* | | | | |
|  |  |  |  |  |
| **Execução:** | **Cumprimento das Atividades** | | | |
| **Atividades / Componentes** | **Bruno** | **Everaldo** | **Rafael Rufino** | **Walter** |
| 4.1 Modelagem Conceitual | X | X | X | X |
| 4.2 Dicionário de Dados |  |  | X |  |
| 4.3 Política de Segurança |  |  |  | X |
| 4.4 Criação de Tablespaces (Dados, Temporária e Índices) |  | X |  |  |
| 4.5 Criação de usuários |  |  |  | X |
| 4.6 Script para criação do Modelo de Dados |  |  |  | X |
| 4.7 Queryes |  | X |  |  |
| 4.8 Modularização |  |  | X |  |
| *Indique se aluno cumpriu a atividade (Sim, Não, Em parte)* | | | | |
| **Fator de Participação ==>** |  |  |  |  |
| *- Com base nas respostas acima defina o fator de participação do aluno para esta fase.* | | | | |

**4 – PROJETO DE BANCO DE DADOS**

**4.1 Modelagem Conceitual**



**4.2 Dicionário de Dados**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **TABELA: BREW\_USER** | | | | | | | | |
| **-** | | **CAMPO** | **DATATYPE** | | **TAMANHO** | | **DESCRIÇÃO** | |
| PK | | ID\_USER | NUMBER | | - | | ID do usuário | |
|  | | LOGIN | VARCHAR2 | | 50 | | Login de acesso do usuário | |
|  | | NAME\_USER | VARCHAR2 | | 100 | | Nome do usuário | |
|  | | PASSWORD\_USER | VARCHAR2 | | 50 | | senha do usuário | |
|  | | PHONE\_NUMBER\_USER | VARCHAR2 | | 20 | | Numero telefônico | |
| **TABELA: BREW\_USER\_MANAGEMENT** | | | | | | | | | |
| **-** | **CAMPO** | | | **DATATYPE** | | **TAMANHO** | | **DESCRIÇÃO** | |
| PK | ID\_USER\_MANAGEMENT | | | NUMBER | | - | | id identificação | |
|  | USER\_MANAGEMENT | | | VARCHAR2 | | 50 | | usuário de identificação | |
|  | NAME\_USER\_MANAGEMENT | | | VARCHAR2 | | 100 | | nome do usuário | |
|  | PASSWORD\_USER\_MANAGEMENT | | | VARCHAR2 | | 100 | | senha de identificação | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TABELA: BREW\_QUESTION** | | | | |
| **-** | **CAMPO** | **DATATYPE** | **TAMANHO** | **DESCRIÇÃO** |
| PK | ID\_QUESTION | NUMBER | - | id identificação |
|  | QUESTION | VARCHAR2 | 4000 | conteúdo da questão |
|  | QUESTION\_NUMBER | NUMBER | 100 |  |
| **TABELA: BREW\_ANSWER** | | | | |
| **-** | **CAMPO** | **DATATYPE** | **TAMANHO** | **DESCRIÇÃO** |
| PK | ID\_ANSWER | NUMBER | - | id identificação |
|  | ANSWER\_TEXT | VARCHAR2 | 4000 | conteúdo da resposta |
|  | WEIGHT\_ANSWER | NUMBER | 100 |  |
| FK | ID\_QUESTION | NUMBER | 100 | id da questão |
| **TABELA: BREW\_TEMPLATE** | | | | |
| **-** | **CAMPO** | **DATATYPE** | **TAMANHO** | **DESCRIÇÃO** |
| PK ,FK | ID\_QUESTION | NUMBER | - | id questão |
| PK,FK | ID\_ANSWER | NUMBER | 4000 | id resposta |
| PK,FK | ID\_USER | NUMBER | 100 | id do usuário |
| **TABELA: BREW\_PROFILE** | | | | |
| **-** | **CAMPO** | **DATATYPE** | **TAMANHO** | **DESCRIÇÃO** |
| PK | ID\_PROFILE | NUMBER | - | id identificação do perfil |
| FK | ID\_USER | NUMBER |  | id do usuário |
|  | NAME\_PROFILE | VARCHAR2 | 50 | descrição do perfil |
|  | RESUMO | VARCHAR2 | 500 | conteúdo descritivo do perfil |
|  | TYPE | NUMBER |  | tipo de perfil |

**4.3 Política de Segurança**

# 4.3.1 Controle de acesso a nível de Sistema

CREATE PROFILE usertec\_prf\_own limit failed\_login\_attempts 3;   
  
CREATE PROFILE invistaja\_prf\_own limit failed\_login\_attempts 3;   
  
CREATE PROFILE owner\_prf limit failed\_login\_attempts 3;   
  
CREATE PROFILE gerente\_prf limit failed\_login\_attempts 3;   
  
CREATE PROFILE staff\_prf limit failed\_login\_attempts 3;

# 4.3.2 Controle de acesso a nível de dados

CREATE ROLE invistaja\_own\_modelo1;   
  
CREATE ROLE owner\_role\_modelo1;   
  
CREATE ROLE gerente\_role\_modelo1;   
  
CREATE ROLE staff\_role\_modelo1;

**4.4 Criação de Tablespaces (Dados, Temporária e Índices)**

CREATE TABLESPACE invistaja\_tbs\_dados DATAFILE   
'C:\ORACLEXE\APP\ORACLE\ORADATA\XE\INVISTAJA\_TBS\_DADOS.DBF' SIZE 10m;

CREATE TABLESPACE invistaja\_tbs\_temp DATAFILE   
'C:\ORACLEXE\APP\ORACLE\ORADATA\XE\INVISTAJA\_TBS\_TEMP.DBF' SIZE 10m;

GRANT ALTER SESSION, CREATE SESSION, CREATE TABLE, CREATE VIEW, CREATE SEQUENCE,   
CREATE SYNONYM, CREATE PROCEDURE, CREATE TRIGGER, CREATE TYPE, query rewrite,   
CREATE ANY INDEX, CREATE PUBLIC SYNONYM TO owner\_role\_modelo1;   
  
GRANT ALTER SESSION, CREATE SESSION, CREATE TABLE, CREATE VIEW, CREATE SEQUENCE,   
CREATE SYNONYM, CREATE PROCEDURE, CREATE TRIGGER, CREATE TYPE, query rewrite,   
CREATE ANY INDEX, CREATE PUBLIC SYNONYM TO gerente\_role\_modelo1;   
  
GRANT ALTER SESSION, CREATE SESSION, CREATE PUBLIC SYNONYM TO staff\_role\_modelo1;   
  
GRANT ALTER SESSION, CREATE SESSION, CREATE TABLE, CREATE VIEW, CREATE SEQUENCE,   
CREATE SYNONYM, CREATE PROCEDURE, CREATE TRIGGER, CREATE TYPE, query rewrite,   
CREATE ANY INDEX, CREATE PUBLIC SYNONYM TO invistaja\_own\_modelo1;

**4.5 Criação de usuários**

CREATE USER invistaja\_dba IDENTIFIED BY blastoise010 DEFAULT TABLESPACE   
invistaja\_tbs\_dados QUOTA UNLIMITED ON invistaja\_tbs\_dados PROFILE   
invistaja\_prf\_own;   
  
GRANT invistaja\_own\_modelo1 TO invistaja\_dba;   
  
CREATE USER walterjunior IDENTIFIED BY blastoise01 DEFAULT TABLESPACE   
invistaja\_tbs\_dados TEMPORARY TABLESPACE invistaja\_tbs\_temp QUOTA UNLIMITED ON   
invistaja\_tbs\_dados PROFILE invistaja\_prf\_own;   
  
GRANT invistaja\_own\_modelo1 TO walterjunior;   
  
CREATE USER everaldo IDENTIFIED BY everaldo12345 DEFAULT TABLESPACE   
invistaja\_tbs\_dados TEMPORARY TABLESPACE invistaja\_tbs\_temp QUOTA UNLIMITED ON   
invistaja\_tbs\_dados PROFILE owner\_prf;   
  
GRANT owner\_role\_modelo1 TO everaldo;   
  
CREATE USER brunobrandao IDENTIFIED BY brunobrandao12345 DEFAULT TABLESPACE   
invistaja\_tbs\_dados QUOTA UNLIMITED ON invistaja\_tbs\_dados PROFILE gerente\_prf   
GRANT gerente\_role\_modelo1 TO invistaja\_dba;   
  
CREATE USER tiophill IDENTIFIED BY tiophill12345 DEFAULT TABLESPACE   
invistaja\_tbs\_dados QUOTA UNLIMITED ON invistaja\_tbs\_dados PROFILE staff\_prf;   
  
GRANT staff\_role\_modelo1 TO invistaja\_dba;

**4.6 Script para criação do Modelo de Dados**

CREATE TABLE brew\_user   
  (   
     id\_user                *INT* NOT NULL PRIMARY KEY,   
     login                  *VARCHAR*(50) NOT NULL,   
     name\_user              *VARCHAR*(100) NOT NULL   
     password\_user          *VARCHAR*(10) NOT NULL,   
     phone\_number\_user      *VARCHAR*(20) NULL  
  );   
  
CREATE TABLE brew\_question   
  (   
     id\_question   *INT* NOT NULL PRIMARY KEY,   
     question      *VARCHAR*(4000) NOT NULL,   
     type\_question *INT*   
  );   
  
CREATE TABLE brew\_user\_management   
  (   
     id\_user\_management        *INT* NOT NULL PRIMARY KEY,   
     user\_management           *VARCHAR*(50) NOT NULL,   
     name\_user\_management      *VARCHAR*(100) NOT NULL,   
     last\_name\_user\_management *VARCHAR*(200) NOT NULL,   
     password\_user\_management  *VARCHAR*(10) NOT NULL   
  );   
  
CREATE TABLE brew\_answer   
  (   
     id\_answer     *INT* NOT NULL PRIMARY KEY,   
     answer\_text   *VARCHAR*(4000) NOT NULL,   
     weight\_answer *INT* NULL,   
     answer\_number *REAL* NULL   
  );   
  
CREATE TABLE brew\_profile   
  (   
     id\_profile *INT* NOT NULL PRIMARY KEY,   
     id\_user    *INT* NOT NULL,   
     CONSTRAINT x FOREIGN KEY (id\_user) REFERENCES brew\_user (id\_user)   
  );   
  
CREATE TABLE brew\_template   
  (   
     id\_user     *INT* NOT NULL,   
     id\_question *INT* NOT NULL,   
     id\_answer   *INT* NOT NULL,   
     CONSTRAINT dx FOREIGN KEY (id\_user) REFERENCES brew\_user (id\_user),   
     CONSTRAINT fx FOREIGN KEY (id\_question) REFERENCES brew\_question (   
     id\_question),   
     CONSTRAINT mx FOREIGN KEY (id\_answer) REFERENCES brew\_answer (id\_answer)   
  );

**4.7 Queryes**

**4.7.1 Consultas Simples**

*--PRIMEIRA CONSULTA:*   
SELECT \*   
FROM   brew\_question Q   
       INNER JOIN brew\_template T   
               ON T.id\_question = Q.id\_question   
       INNER JOIN brew\_user U   
               ON U.id\_user = T.id\_user   
       INNER JOIN brew\_answer A   
               ON T.id\_answer = A.id\_answer   
  
*--SEGUNDA CONSULTA:*   
SELECT P.id\_user,   
       U.name\_user   
FROM   brew\_profile P   
       INNER JOIN brew\_user U   
               ON U.id\_user = P.id\_user   
       INNER JOIN brew\_template T   
               ON T.id\_user = U.id\_user   
WHERE  T.id\_user = 2;   
  
*--TERCEIRA CONSULTA:*   
SELECT Q.question,   
       U.name\_user   
FROM   brew\_question Q   
       INNER JOIN brew\_template T   
               ON T.id\_question = Q.id\_question   
       INNER JOIN brew\_user U   
               ON U.id\_user = T.id\_user   
       INNER JOIN brew\_answer A   
               ON T.id\_answer = A.id\_answer   
WHERE  A.id\_answer IN ( 1, 2 );   
  
*--QUARTA CONSULTA:*   
SELECT P.id\_user,   
       U.name\_user,   
       U.login,   
       U.cpf\_user,   
       P.id\_profile   
FROM   brew\_profile P   
       INNER JOIN brew\_user U   
               ON U.id\_user = P.id\_user   
       INNER JOIN brew\_template T   
               ON T.id\_user = U.id\_user   
WHERE  T.id\_user IN ( 1, 2, 3, 4 );   
  
*--QUINTA CONSULTA:*   
SELECT Q.question,   
       U.name\_user,   
       T.id\_user   
FROM   brew\_question Q   
       INNER JOIN brew\_template T   
               ON T.id\_question = Q.id\_question   
       LEFT JOIN brew\_user U   
              ON U.id\_user = T.id\_user   
WHERE  U.id\_user = 1; 

**4.7.2 Consultas Complexas**

*--PRIMEIRA CONSULTA:*   
SELECT \*   
FROM   brew\_question Q   
       inner join brew\_template T   
               ON T.id\_question = Q.id\_question   
       inner join brew\_user U   
               ON U.id\_user = T.id\_user   
       inner join brew\_answer A   
               ON T.id\_answer = A.id\_answer   
ORDER  BY T.id\_question;   
  
*--SEGUNDA CONSULTA:*   
SELECT P.id\_user,   
       **Initcap**(U.name\_user)   
FROM   brew\_profile P   
       inner join brew\_user U   
               ON U.id\_user = P.id\_user   
       inner join brew\_template T   
               ON T.id\_user = U.id\_user   
WHERE  T.id\_user = 2;   
  
*--TERCEIRA CONSULTA:*   
SELECT Q.question,   
       U.name\_user,   
       **SUM**(A.weight\_answer)   
FROM   brew\_question Q   
       inner join brew\_template T   
               ON T.id\_question = Q.id\_question   
       inner join brew\_user U   
               ON U.id\_user = T.id\_user   
       inner join brew\_answer A   
               ON T.id\_answer = A.id\_answer   
WHERE  A.id\_answer IN ( 1, 2 )   
GROUP  BY Q.question,   
          U.name\_user;   
  
*--QUARTA CONSULTA:*   
SELECT U.name\_user,   
       U.login,   
       **To\_char**(U.cpf\_user)   
FROM   brew\_profile P   
       inner join brew\_user U   
               ON U.id\_user = P.id\_user   
       inner join brew\_template T   
               ON T.id\_user = U.id\_user   
WHERE  T.id\_user IN ( 4578, 874, 0923, 4894 );   
  
*--QUINTA CONSULTA*   
SELECT brew\_user.name\_user,   
       **Toupper**(brew\_user.login),   
       **To\_char**(brew\_user.cpf\_user)   
FROM   brew\_profile   
       inner join brew\_user   
               ON brew\_user.id\_user = brew\_profile.id\_user   
       inner join brew\_template   
               ON brew\_template.id\_user = brew\_user.id\_user   
WHERE  brew\_user.login = 'ECARDOSO';

**4.7.3 Sub Consultas**

*--PRIMEIRA CONSULTA*   
SELECT brew\_profile.id\_profile,   
       (SELECT brew\_user.name\_user   
        FROM   brew\_user   
        WHERE  brew\_user.id\_user = brew\_profile.id\_user) AS USUARIO   
FROM   brew\_profile;   
  
*--SEGUNDA CONSULTA:*   
SELECT brew\_template.id\_user,   
       brew\_template.id\_question,   
       (SELECT brew\_answer.answer\_text   
        FROM   brew\_answer   
        WHERE  brew\_answer.id\_answer = brew\_template.id\_user) AS QUESTAO   
FROM   brew\_template;   
  
*--TERCEIRO CONSULTA:*   
SELECT brew\_template.id\_user,   
       brew\_template.id\_question   
FROM   brew\_template   
WHERE  brew\_template.id\_user IN (SELECT brew\_user.id\_user   
                                 FROM   brew\_user);

**4.7.4 Consulta com Left ou Right JOIN**

*--PRIMEIRA CONSULTA:*   
SELECT \*   
FROM   brew\_template   
       left join brew\_question   
              ON brew\_question.id\_question = brew\_template.id\_question   
WHERE  brew\_template.id\_answer = 1;   
  
*--SEGUNDA CONSULTA:*   
SELECT brew\_template.id\_answer,   
       brew\_question.id\_question   
FROM   brew\_template   
       right join brew\_question   
               ON brew\_question.id\_question = brew\_template.id\_question   
WHERE  brew\_template.id\_answer IN( 90, 91, 92 );

**4.8 Modularização**

**4.8.1 Procedures**

*CREATE OR replace PROCEDURE* ***Create\_user****(*

*p\_login IN brew\_user.login%TYPE,*

*p\_password IN brew\_user.password\_user%TYPE,*

*p\_name IN brew\_user.name\_user%TYPE,*

*p\_phonenumber IN brew\_user.phone\_number\_user%TYPE)*

*IS*

*BEGIN*

*INSERT INTO brew\_user*

*(login,*

*password\_user,*

*name\_user,*

*phone\_number\_user)*

*VALUES (p\_login,*

*p\_password,*

*p\_name,*

*p\_phonenumber);*

*END create\_user;*

*------------------------------------------*

*CREATE OR replace PROCEDURE* ***Add\_template****(*

*p\_user\_id IN brew\_template.id\_user%TYPE,*

*p\_question\_id IN brew\_template.id\_question%TYPE,*

*p\_answer\_id IN brew\_template.id\_answer%TYPE)*

*IS*

*BEGIN*

*INSERT INTO brew\_template*

*(id\_user,*

*id\_question,*

*id\_answer)*

*VALUES (p\_user\_id,*

*p\_question\_id,*

*p\_answer\_id);*

*END add\_template;*

**4.8.2 Functions**

*CREATE OR replace FUNCTION* ***Gettotal****(id\_user IN brew\_user.id\_user%TYPE)*

*RETURN NUMBER*

*IS*

*acctotal NUMBER(11, 2);*

*BEGIN*

*SELECT* ***SUM****(brew\_answer.weight\_answer)*

*INTO acctotal*

*FROM brew\_answer,*

*brew\_template*

*WHERE brew\_answer.id\_answer = brew\_template.id\_answer*

*AND brew\_template.id\_user = id\_user;*

*RETURN( acctotal );*

*END;*

------------------------------------------------------------------------

CREATE OR replace FUNCTION **Gettotalanswer**(id\_user IN brew\_user.id\_user%*TYPE*)

RETURN *NUMBER*

IS

acccount *NUMBER*(11, 2);

BEGIN

SELECT **Count**(brew\_answer.id\_answer)

INTO acccount

FROM brew\_answer,

brew\_template

WHERE brew\_answer.id\_answer = brew\_template.id\_answer

AND brew\_template.id\_user = id\_user;

RETURN( acccount );

END;

**4.8.3 Triggers**

CREATE OR replace TRIGGER "AUTO\_TRG"

BEFORE INSERT ON brew\_user

FOR EACH ROW

BEGIN

IF inserting THEN

IF :NEW."id\_user" IS NULL THEN

SELECT autoincrement.NEXTVAL

INTO :NEW."ID\_USER"

FROM dual;

END IF;

END IF;

END;

-------------------------

CREATE OR replace TRIGGER "INCREMENTPROFILE"

BEFORE INSERT ON "brew\_profile"

FOR EACH ROW

BEGIN

IF inserting THEN

IF :NEW."id\_user" IS NULL THEN

SELECT autoincrement.NEXTVAL

INTO :NEW."ID\_USER"

FROM dual;

END IF;

END IF;

END;

-------------------------

CREATE OR replace TRIGGER SYSTEM.repcatlogtrig

AFTER UPDATE OR DELETE ON SYSTEM.repcat$\_repcatlog

BEGIN

sys.dbms\_alert.**Signal**('repcatlog\_alert', '');

END;

CREATE OR replace TRIGGER SYSTEM.def$\_propagator\_trig

BEFORE INSERT ON SYSTEM.def$\_propagator

DECLARE

prop\_count *NUMBER*;

BEGIN

SELECT **Count**(\*)

INTO prop\_count

FROM SYSTEM.def$\_propagator;

IF ( prop\_count > 0 ) THEN

*-- Raise duplicate propagator error*

sys.dbms\_sys\_error.**Raise\_system\_error**(-23394);

END IF;

END;