CENTRO UNIVERSITÁRIO UNIVATES
CURSO DE ANALISE E DESENVOLVIMENTO DE SISTEMAS

ESTUDO DE CASO DE UM SISTEMA DE ORGANIZAÇÕES DE AÇÕES COM BASE NA FERRAMENTA 5W2H

Lucas Leandro de Moura

Lajeado, Junho de 2017

Lucas Leandro de Moura

ESTUDO DE CASO DE UM SISTEMA DE ORGANIZAÇÕES DE AÇÕES COM BASE NA FERRAMENTA 5W2H

Trabalho apresentado na disciplina Programação Avançada, no curso de tecnologia em Analise e Desenvolvimento de Sistemas, do Centro Universitário UNIVATES.

Professor: Prof. Ms. Juliano Dertzbacher

Lajeado, Março de 2017

1. INTRODUÇÃO

Este documento foi elaborado para uma tarefa do módulo de Programação Avançada do curso de Análise e Desenvolvimento de Sistemas da Univates. Com isso, em um estudo breve sobre uma situação real solicitada, identifiquei a necessidade da elaboração de um plano de um sistema que realiza o controle de ações a serem realizadas pelas corporações utilizando a ferramenta administrativa 5W2H. A ferramenta 5W2H, se posiciona em base de algumas perguntas básicas para a organização das atividades que a corporação deverá adotar, que são: Quem? O que? Onde? Quando? Por que? Como?.

Este método é muito simples, que agiliza todas ações de uma empresa, trazendo uma visão rápida e simples de como as corporações poderão alcançar as suas metas e indicadores de desempenho.

2.1 Análise de Requisitos

Objetivo: Desenvolver um sistema para controlar ações com a ferramenta de gestão estratégica 5W2H.

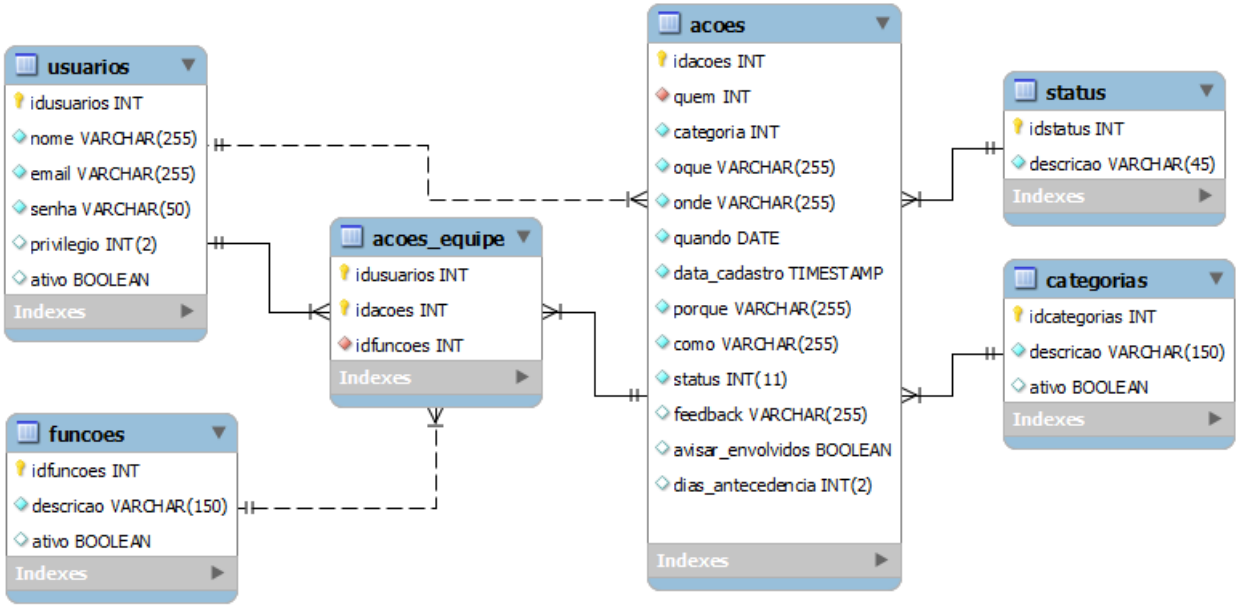
Requisitos funcionais

1. Manter cadastros de *Stakeholders;*
2. Manter cadastro de categorias;
3. Manter cadastro de funções;
4. Gerenciar níveis de permissão dos *Stakeholders*;
5. Gerenciar equipes das ações;
6. Manter o cadastro de ações dentro da grade 5W2H;
7. Permitir gerar relatórios através de filtros: Quem? O que? Onde? Quando? Por que? Como? Concluído?;
8. Avisar os *Stakeholders* de ações que estão atrasadas;
9. Permitir que o *Stakeholder* conclua a ação informando o *feedback* da atividade realizada;

Requisitos Não-funcionais;

1. Utilizar linguagem *Java*;
2. Utilizar banco de dados *PostgreSQL*;
3. Gerar relatório através da biblioteca *IReport*;
4. Validar campos de data;
5. Validar como obrigatório os campos quem, o que, onde, quando, por que, como;
6. Validar o acesso dos usuários através de autenticação;

2.2 Modelo Relacional



2.3 Dump SQL

```
CREATE TABLE acoes (
    quem integer,
    categoria integer NOT NULL,
    oque character varying(255) NOT NULL,
    idacoes integer NOT NULL,
    onde character varying(255) NOT NULL,
    quando date NOT NULL,
    data_cadastro timestamp(0) without time zone DEFAULT now(),
    porque character varying(255) NOT NULL,
    como character varying(255) NOT NULL,
    status integer DEFAULT 0 NOT NULL,
    feedback character varying(255),
    avisa_envolvidos boolean DEFAULT false,
    dias_antecedencia integer
);


ALTER TABLE acoes OWNER TO postgres;
```

```
CREATE TABLE acoes_equipe (
    idusuarios integer NOT NULL,
    idacoes integer NOT NULL,
    idfuncoes integer
);


ALTER TABLE acoes_equipe OWNER TO postgres;


CREATE TABLE categorias (
    idcategorias integer NOT NULL,
    descricao character varying(150) NOT NULL,
    ativo boolean DEFAULT true
);


ALTER TABLE categorias OWNER TO postgres;



CREATE SEQUENCE categorias_idcategorias_seq
    START WITH 1
    INCREMENT BY 1
    NO MINVALUE
    NO MAXVALUE
    CACHE 1;


ALTER TABLE categorias_idcategorias_seq OWNER TO postgres;


ALTER SEQUENCE categorias_idcategorias_seq OWNED BY
categorias.idcategorias;
```

```
CREATE TABLE funcoes (
    idfuncoes integer NOT NULL,
    descricao character varying(150) NOT NULL,
    ativo boolean DEFAULT true
);


ALTER TABLE funcoes OWNER TO postgres;


CREATE SEQUENCE funcoes_idfuncoes_seq
    START WITH 1
    INCREMENT BY 1
    NO MINVALUE
    NO MAXVALUE
    CACHE 1;


ALTER TABLE funcoes_idfuncoes_seq OWNER TO postgres;


ALTER SEQUENCE funcoes_idfuncoes_seq OWNED BY funcoes.idfuncoes;



CREATE TABLE status (
    idstatus integer NOT NULL,
    descricao character varying(45) NOT NULL
);


ALTER TABLE status OWNER TO postgres;


CREATE SEQUENCE status_idstatus_seq
    START WITH 1
    INCREMENT BY 1
    NO MINVALUE
    NO MAXVALUE
    CACHE 1;
```

```
ALTER TABLE status_idstatus_seq OWNER TO postgres;


ALTER SEQUENCE status_idstatus_seq OWNED BY status.idstatus;



CREATE SEQUENCE table_idacoes_seq
    START WITH 1
    INCREMENT BY 1
    NO MINVALUE
    NO MAXVALUE
    CACHE 1;


ALTER TABLE table_idacoes_seq OWNER TO postgres;


ALTER SEQUENCE table_idacoes_seq OWNED BY acoes.idacoes;


CREATE TABLE usuarios (
    idusuarios integer NOT NULL,
    nome character varying(255) NOT NULL,
    email character varying(255) NOT NULL,
    senha character varying(50) NOT NULL,
    privilegio integer DEFAULT 0 NOT NULL,
    ativo boolean DEFAULT true
);


ALTER TABLE usuarios OWNER TO postgres;


CREATE SEQUENCE usuarios_idusuarios_seq
    START WITH 1
    INCREMENT BY 1
    NO MINVALUE
```

```
    NO MAXVALUE
    CACHE 1;


ALTER TABLE usuarios_idusuarios_seq OWNER TO postgres;


ALTER SEQUENCE usuarios_idusuarios_seq OWNED BY usuarios.idusuarios;


ALTER TABLE ONLY acoes ALTER COLUMN idacoes SET DEFAULT nextval('table_idacoes_seq'::regclass);


ALTER TABLE ONLY categorias ALTER COLUMN idcategorias SET DEFAULT nextval('categorias_idcategorias_seq'::regclass);


ALTER TABLE ONLY funcoes ALTER COLUMN idfuncoes SET DEFAULT nextval('funcoes_idfuncoes_seq'::regclass);


ALTER TABLE ONLY status ALTER COLUMN idstatus SET DEFAULT nextval('status_idstatus_seq'::regclass);


ALTER TABLE ONLY usuarios ALTER COLUMN idusuarios SET DEFAULT nextval('usuarios_idusuarios_seq'::regclass);


ALTER TABLE ONLY acoes_equipe
    ADD CONSTRAINT acoes_equipe_pkey PRIMARY KEY (idusuarios, idacoes);
```

```
ALTER TABLE ONLY categorias
   ADD CONSTRAINT categorias_pkey PRIMARY KEY (idcategorias);


ALTER TABLE ONLY funcoes
   ADD CONSTRAINT funcoes_pkey PRIMARY KEY (idfuncoes);


ALTER TABLE ONLY status
   ADD CONSTRAINT status_pkey PRIMARY KEY (idstatus);


ALTER TABLE ONLY acoes
   ADD CONSTRAINT table_pkey PRIMARY KEY (idacoes);


ALTER TABLE ONLY usuarios
   ADD CONSTRAINT usuarios_pkey PRIMARY KEY (idusuarios);


ALTER TABLE ONLY acoes_equipe
    ADD CONSTRAINT acoes_equipe_fk FOREIGN KEY (idacoes) REFERENCES
acoes(idacoes);


ALTER TABLE ONLY acoes_equipe
   ADD CONSTRAINT acoes_equipe_fk1 FOREIGN KEY (idfuncoes) REFERENCES
funcoes(idfuncoes);


ALTER TABLE ONLY acoes
        ADD CONSTRAINT acoes_fk FOREIGN KEY (status) REFERENCES
status(idstatus);
```

ALTER TABLE ONLY acoes

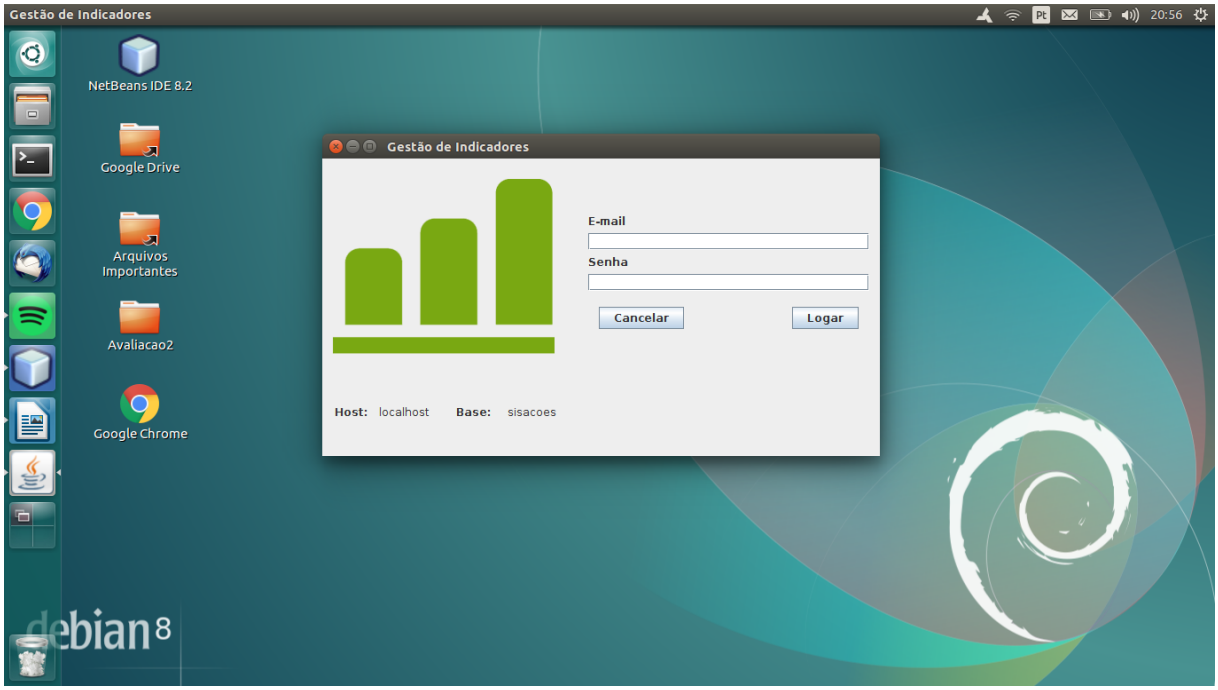
ADD CONSTRAINT acoes_fk1 FOREIGN KEY (categoria) REFERENCES categorias(idcategorias);

ALTER TABLE ONLY acoes

ADD CONSTRAINT acoes_fk2 FOREIGN KEY (quem) REFERENCES usuarios(idusuarios);

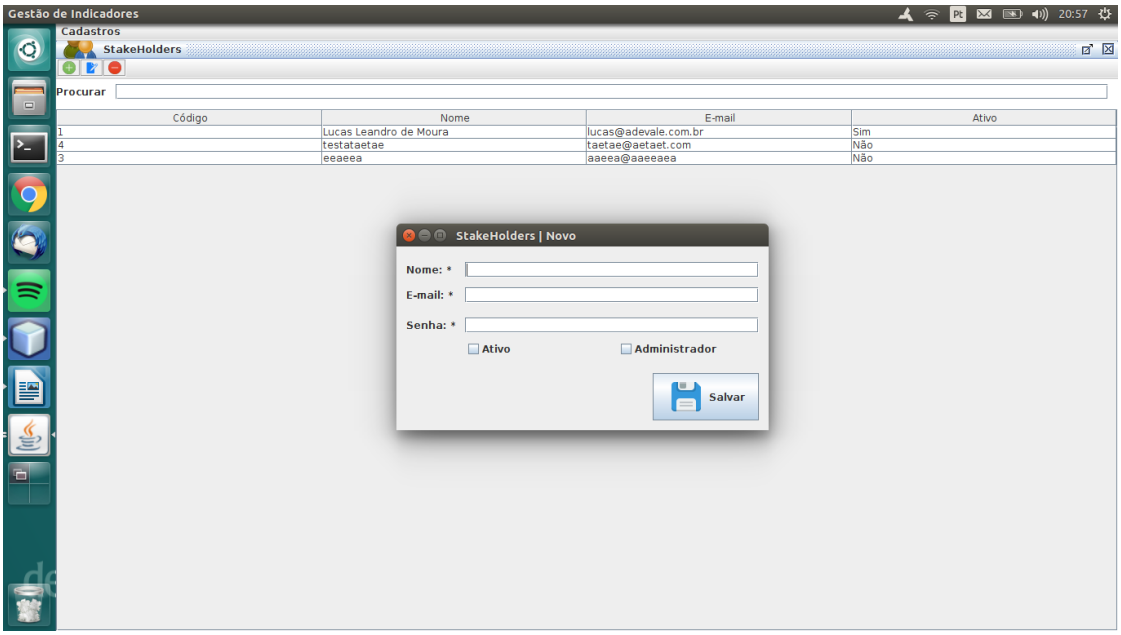
4. Interfaces

a) Tela de configuração de acesso ao banco de dados, instalado na máquina.

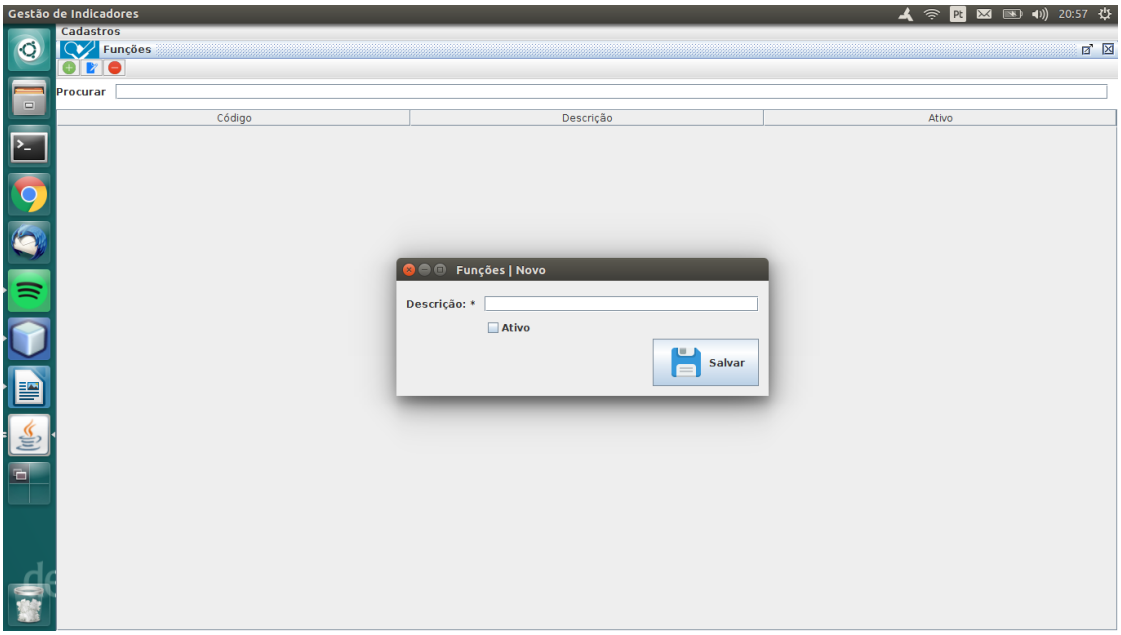


b) Antes de entrar na tela principal, o sistema solicita para que o usuário informe o seu login e senha. De padrão foi inserido o usuário admin@admin.com.br e senha 1234.

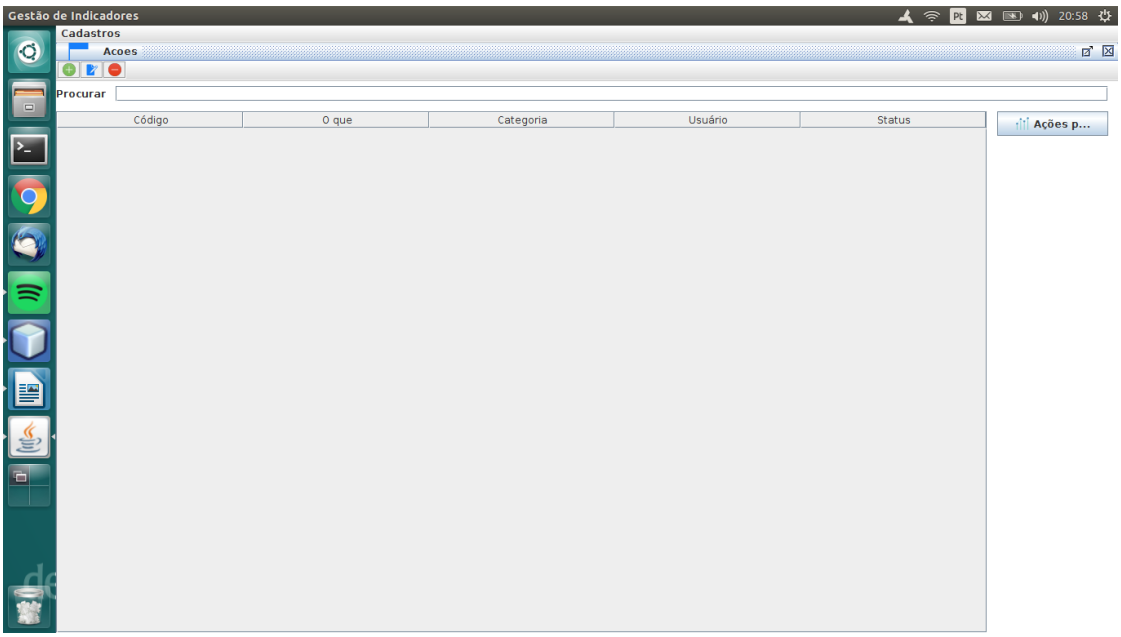
c) Cadastro de stakeholders
Todos os campos obrigatórios, são demarcados pelo *.



d) Cadastro de funções, é responsável por cadastrar funções que serão utilizadas na classificação dos participantes dentro daquele evento.



e) No cadastro de ações, no qual é a tela que é a parte central do sistema. O mesmo poderá manter os registros das atividades, que terão que ser tomadas e controladas.



10. Referencial Bibliográfico

5W2h Planejamento Estratégico, Disponível em: <https://sites.google.com/site/planejaweb/5w2h>. Acessado em: 07 Mar. 2017.

Métodos e técnicas de planejamento: <http://www.valentim.pro.br/data/documents/Metodos_Tecnicas_Planejamento_1.pdf >. Acessado em: 07 Mar. 2017.