CURSO SUPERIOR DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS, REDES DE COMPUTADORES E GESTÃO DE TECNOLOGIA DA INFORMAÇÃO

BRAUNER SOUZA DE MELLO
DALMIR DA SILVA

GRAFORM

PORTO ALEGRE,
DEZ de 2013

Sumário

Lista de Figuras

Lista de Tabelas

Resumo

Nos dias atuais, empresas que fornecem serviços de monitoramento de indicadores e índices de satisfação de clientes, precisam de uma solução de software que seja poderosa, e ao mesmo tempo, simples de ser utilizada. Este é o caso da Stringhini Marketing. Este trabalho propõe o desenvolvimento de um sistema de gerenciamento de formulários eletrônicos, direcionado ao escopo de negócio da Stringhini. O sistema, denominado Graform, tem, como principal objetivo, o papel de substituir a forma como a Stringhini gerencia e entrega o seu serviço de questionários eletrônicos, possibilitando que os mesmos deixem de ser criados e editados diretamento na base de dados e passem a ter uma interface web, simples e funcional. Baseando-se em regras aplicadas a cada resposta de uma questão do formulário, o sistema Graform permite controlar a ordem das próximas questões, alterando o fluxo de respostas e customizando o formulário como um todo. Possibilitando assim, coletar informações mais relevantes do que um formulário convencional.

1 Introdução

1.1 A empresa

A Stringhini Marketing é uma empresa de marketing com experiência em gestão da operação comercial e do relacionamento com o cliente através do monitoramento de indicadores chave e índices de satisfação. A empresa dá muita importância à inteligência competitiva e procura transformar informação em estratégia e decisão em resultado, agregando valor ao negócio dos clientes.

1.2 Problema a ser resolvido

Devido ao grande número de informações trabalhadas em uma pesquisa de marketing, há uma grande dificuldade em não se ter uma interface amigável de criação e gerenciamento de pesquisas.

O desenvolvimento dos formulários é diretamente no banco de dados através de scripts, o que gerar um grande risco de inconsistência pela possibilidade de erro humano, a dificuldade de manutenção é muito alta, tornando a personalização de um formulário quase impossível. No atual momento, a empresa não vê condições de adquirir um software no mercado, por julgar que não atende em relação aos custos e necessidades específicas de cada projeto.

1.3 Objetivos do trabalho

O objetivo do projeto é desenvolver uma ferramenta web para criação e gerenciamento de pesquisas, com uma interface amigável e de uso fácil para os clientes. Com o desenvolvimento deste software, a empresa permitirá autonomia aos clientes, pois os mesmos terão condições de desenvolver suas pesquisas conforme sua necessidade, bem como, possibilitará a que sua equipe tenha condições de focar apenas na inteligência do negócio e não mais na operação.

2 Sistema

2.1 Definição

2.1.1 Funcionalidades

O projeto em questão trata-se do desenvolvimento de um sistema de criação e gerenciamento de formulários eletrônicos online. A ferramenta consiste em uma página *web* onde os clientes da Stringhini poderão criar e gerenciar seus próprios formulários, através de uma conta de usuário.

O sistema possui as seguintes características:

1. Provê uma interface web para manipulação dos formulários;
2. Possibilita a criação de dois tipos distintos de formulários:
   (a) Condicional: A sequência das questões a serem respondidas pode variar de acordo com as respostas;
   (b) Contínuo: Possui sequência estática de questões;
3. Usuários acessam os dados concorrentemente;
4. Possui interface amigável, limpa e simples;

2.1.2 Modelagem

Um formulário consiste em um conjunto de questões encadeadas de duas maneiras possíveis: contínua ou condicional.

Da forma contínua, as questões são respondidas pela ordem de seus números. Por exemplo: a questão nº 1 será a primeira a ser respondida, seguida pela questão nº 2, nº 3... e assim sucessivamente.

Quando o formulário for condicional, a sequência de questões não segue, necessariamente, tal ordem. A sequência de questões é dinamicamente escolhida em função das respostas, isto é, respostas diferentes podem determinar diferentes próximas questões.

Grafo

Para modelar o sistema, seguindo as características de condicionalidade citadas acima, os formulários são organizados internamente de uma maneira análoga a uma estrutura de dados chamada *grafo*.

Um grafo é uma estrutura de dados $G(V, A)$, onde V é um conjunto não vazio de objetos denominados vértices e A é um conjunto de pares de V, chamado arestas [12,14]. Analogamente, um formulário é uma estrutura $F(Q, R)$, onde Q é um conjunto de questões e R é um conjunto de pares de Q. R, sendo um conjunto de pares ordenados, torna o grafo orientado.

Para uma dada questão $q \in Q$, a próxima questão Q_k é determinada pelas possíveis respostas A de q e uma regra r. Portanto, cada elemento de R é definido como uma função que mapeia a questão atual, a resposta e uma regra para uma próxima questão:

$$R_n \quad = \quad f(q, A_n, r) : Q_k \tag{1}$$

Visualmente podemos perceber que, dependendo da resposta para cada questão, pode-se percorrer diferentes regiões do formulário (grafo). A próxima figura representa tal formulário, onde q_n são questões e r_n são associações entre respostas e regras para cada questão q.

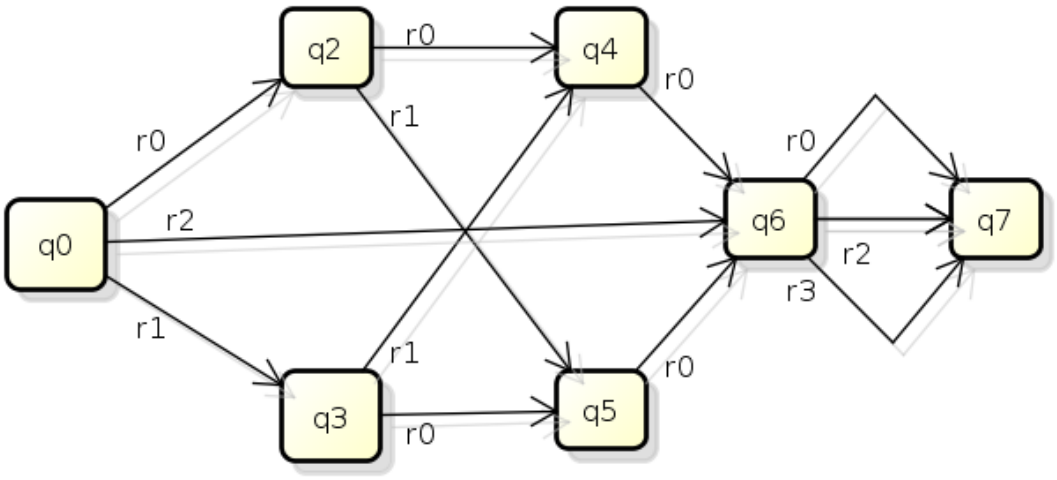

Figura 1: Formulário visto como um grafo.

Para controlar a primeira e última questão de um formulário, cada vértice contém 2 atributos: *inicial* e *final*. Pode haver apenas um vértice inicial. Entretanto, múltiplos vértices podem ser considerados terminais (finais), permitindo que o formulário seja encerrado de forma controlada.

2.2 Tecnologias escolhidas

Devido ao apoio da instituição, onde esse trabalho foi realizado, à projetos e iniciativas *open source*, decidiu-se utilizar, em todas as partes desse trabalho, ferramentas livres e/ou de código fonte aberto. Desde o Sistema Operacional onde ele foi desenvolvido até o programa utilizado para escrever esse documento.

2.2.1 Sistema Operacional

O Sistema Operacional (SO) utilizado é o Linux. Mais precisamente a distribuição Ubuntu [5], de código aberto, construído a partir do núcleo Linux, baseado no Debian.

2.2.2 Versionamento

O sistema desenvolvido utiliza o Git. Git é um sistema de controle de versão distribuído e um sistema de gerenciamento de código fonte, com ênfase em velocidade. O Git foi inicialmente projetado e desenvolvido por Linus Torvalds para o desenvolvimento do kernel Linux, mas foi adotado por muitos outros projetos [15].

2.2.3 Produção de textos científicos

Fugindo do senso comum, utilizou-se o LaTeX para escrever esse documento. LaTeX é utilizado amplamente para a produção de textos matemáticos e científicos devido à sua alta qualidade tipográfica [10].

2.2.4 *Framework* de desenvolvimento

Levando em consideração as características do sistema e a ideia de criar a aplicação utilizando a arquitetura MVC (Model-View-Controler), decidiu-se implementar o sistema Graform usando o *framework* livre Ruby on Rails, projeto de código aberto escrito na linguagem Ruby. Ruby on Rails foi uma extração de David Heinemeier Hansson de um projeto seu, o gerenciador de projetos Basecamp [2, 9, 13]. Foi lançado a público pela primeira vez em julho de 2004.

2.2.5 Linguagem de programação

Por termos optado pelo *framework* Ruby on Rails, a linguagem escolhida foi, obviamente, o Ruby. Uma linguagem dinâmica, open source com foco na simplicidade e na produtividade [6].

2.2.6 Biblioteca *javascript*

Como biblioteca *javascript*, o jQuery [4] foi utilizado, devido a sua simplicidade e capacidade de funcionar em diferentes navegadores.

2.2.7 Servidor web

O servidor web escolhido foi o Nginx, em detrimento do Apache, devido ao fato de o Nginx ser um servidor web rápido, leve, e com inúmeras possibilidades de configuração para melhor performance. Técnicamente, o Nginx consome menos memória que o Apache, pois lida com requisições Web através do conceito de *event-based web server*, já o Apache é baseado no conceito *process-based server* [7].

2.2.8 Servidor de banco de dados

O servidor de banco de dados utilizado é o MySQL. O MySQL é um sistema de gerenciamento de banco de dados (SGBD), que utiliza a linguagem SQL (Linguagem de Consulta Estruturada, do inglês *Structured Query Language*) como interface. É atualmente um dos bancos de dados mais populares, com mais de 10 milhões de instalações pelo mundo [8].

2.3 Análise das soluções existentes

Para o presente trabalho, pesquisamos os principais fornecedores de funcionalidades similares no mercado. Três fornecedores foram identificados como os mais aptos a fornecer os serviços desejados pela Stringhini. São eles:

1. Surveymonkey [11];
2. Eval & Go [1];
3. Google Forms [3].

Dentre os fornecedores acima citados, pesquisamos as funcionalidades mais relevantes providas pelos mesmo. Como segue:

1. Cadastro de opções de respostas;
2. Alterar questões de acordo com o tipo de respostas;
3. Preço acessível;
4. Reaproveitar formulários.

No quadro abaixo, consta o comparativo de funcionalidades entre os fornecedores de serviços de formulários online escolhidos, também os serviços que o projeto Graform propõem desenvolver.

	Surveymonkey	Eval & Go	Google	*Graform*
Cadastro de opções de respostas	X	X	X	*X*
Alterar questões de acordo com o tipo de respostas	X	X		*X*
Preço acessível			X	*X*
Reaproveitar formulários		X		*X*

Tabela 1: Comparativo de funcionalidades.

3 Especificação dos requisitos

3.1 Requisitos Funcionais

As especificações a seguir mostram, detalhadamente, cada requisito funcional.

3.1.1 Cadastrar usuário.

Código	RF. 001
Título	Cadastrar usuário.
Descrição	Cadastrar um usuário no sistema.
Pré-condições	Nenhuma.
Pós-condições	Ir para tela de login.
Cenários	
1. Cenário Principal	Um usuário ainda não cadastrado acessa a área de Sign up

Tabela 2: RF. 001 - Cadastrar usuário.

3.1.2 Autenticar usuário no sistema.

Código	RF. 002
Título	Autenticar usuário no sistema.
Descrição	Autenticar um usuário cadastrado no sistema.
Pré-condições	Usuário estar previamente cadastrado.
Pós-condições	Listar formulários do usuário.
Cenários	
1. Cenário Principal	Usuário acessa a área de Login.
2. Cenário Alternativo	Usuário tenta acessar qualquer área restrita do site.

Tabela 3: RF. 002 - Autenticar usuário no sistema.

3.1.3 Cadastrar formulário.

Código	RF. 003
Título	Cadastrar formulário.
Descrição	Cadastrar um formulário no sistema associado a um usuário.
Pré-condições	Usuário estar logado no sistema.
Pós-condições	Lista de formulários cadastrados no sistema associados ao usuário logado.
Cenários	
1. Cenário Principal	Usuário realiza a inserção de um novo formulário no sistema, associando-o automaticamente a si.

Tabela 4: RF. 003 - Cadastrar formulário.

3.1.4 Editar formulário.

Código	RF. 004
Título	Editar formulário.
Descrição	Editar um formulário existente. Incluindo a adição de novas questões e opções.
Pré-condições	Usuário estar logado no sistema, existir um formulário.
Pós-condições	Continuar na tela de edição.
Cenários	
1. Cenário Principal	Usuário solicita tela de edição de formulário. Nessa tela, é possível adicionar, remover, e editar questões.

Tabela 5: RF. 004 - Editar formulário.

3.1.5 Cadastrar questão.

Código	RF. 005
Título	Cadastrar questão.
Descrição	Cadastrar uma nova questão no sistema associada a um formulário.
Pré-condições	Existir um formulário. Estar na tela de edição desse formulário.
Pós-condições	N/A
Cenários	
1. Cenário Principal	Usuário está na tela de edição de um formulário, solicita a inclusão de uma nova questão ao formulário.

Tabela 6: RF. 005 - Cadastrar questão.

3.1.6 Cadastrar opção.

Código	RF. 006
Título	Cadastrar opção.
Descrição	Um usuário edita um formulário. Para as questões do tipo Múltipla Escolha e Única Escolha, o usuário solicita a inclusão de uma nova opção para uma dada questão.
Pré-condições	Existir uma questão cadastrada no sistema.
Cenários	
1. Cenário Principal	Usuário insere uma nova opção.

Tabela 7: RF. 006 - Cadastrar opção.

3.1.7 Responder um formulário.

Código	RF. 007
Título	Responder um formulário.
Descrição	Responder um fomulário. Pode ser realizado por um usuário anônimo ou um usuário cadastrado no sistema.
Pré-condições	Existir um formulário no sistema.
Pós-condições	Tela de agradecimento.
Cenários	
1. Cenário Principal	Acesso ao link de resposta de um formulário.

Tabela 8: RF. 007 - Responder um formulário.

3.1.8 Visualizar as respostas de um formulário.

Código	RF. 008
Título	Visualizar as respostas de um formulário.
Descrição	O usuário dono do formulário poderá acessar a página de relatório e visualizar todas as respostas de uma dado formulário.
Pré-condições	Existir um formulário cadastrado no sistema.
Pós-condições	N/A
Cenários	
1. Cenário Principal	Usuário acessa a área de relatórios, escolhe um formulário para ser visualizado.
2. Cenário Alternativo	Usuário com um formulário cadastrado no sistema, acessa esse formulário e solicita a visualização do relatório desse formulário.

Tabela 9: RF. 008 - Visualizar as respostas de um formulário.

3.2 Requisitos não Funcionais

As especificações a seguir mostram os requisitos não funcionais do sistema.

3.2.1 Utilizar a nuvem da Amazon.

Código	RNF. 001
Título	Utilizar a nuvem da Amazon.
Descrição	Toda a aplicação deverá rodar usando a infraestrutura de nuvem da Amazon.

Tabela 10: RNF. 001 - Utilizar a nuvem da Amazon.

3.2.2 Utilizar máquina como servidor de aplicação.

Código	RNF. 002
Título	Utilizar máquina como servidor de aplicação.
Descrição	Será utilizado uma instância do tipo micro na nuvem da Amazon como servidor de aplicação.

Tabela 11: RNF. 002 - Utilizar máquina como servidor de aplicação.

3.2.3 Utilizar máquina como servidor de banco de dados.

Código	RNF. 003
Título	Utilizar máquina como servidor de banco de dados.
Descrição	Será utilizado uma instância do tipo micro na nuvem da Amazon como servidor de banco de dados.

Tabela 12: RNF. 003 - Utilizar máquina como servidor de banco de dados.

3.2.4 Utilizar servidor de banco de dados MySQL.

Código	RNF. 004
Título	Utilizar servidor de banco de dados MySQL.
Descrição	Será utilizado o MySQL como servidor de banco de dados.

Tabela 13: RNF. 004 - Utilizar servidor de banco de dados MySQL.

3.2.5 Utilizar o nginx como servidor web.

Código	RNF. 005
Título	Utilizar o nginx como servidor web.
Descrição	Será utilizado o nginx como servidor web.

Tabela 14: RNF. 005 - Utilizar o nginx como servidor web.

3.2.6 Utilizar a linguagem Ruby.

Código	RNF. 006
Título	Utilizar a linguagem Ruby.
Descrição	Será utilizado a linguagem Ruby para o desenvolvimento da aplicação.

Tabela 15: RNF. 006 - Utilizar a linguagem Ruby.

3.2.7 Utilizar o framework Ruby on Rails.

Código	RNF. 007
Título	Utilizar o framework Ruby on Rails.
Descrição	Será utilizado o framework Ruby on Rails para o desenvolvimento da aplicação.

Tabela 16: RNF. 007 - Utilizar o framework Ruby on Rails.

3.3 Casos de uso

3.3.1 Cadastrar usuário.

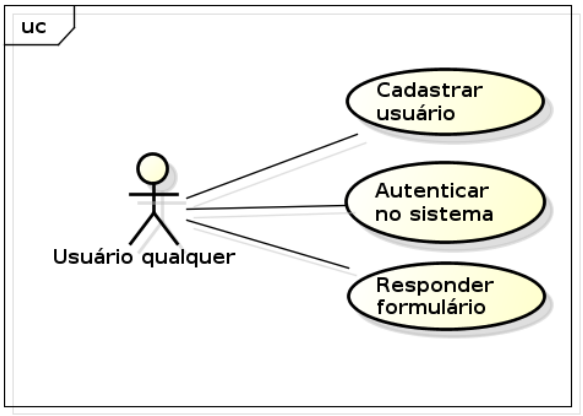

Figura 2: Cadastrar usuário

Código:	UC01.
Nome do UC:	Cadastrar usuário.
Objetivo:	Registrar um novo usuário no sistema.
Ativação:	Página principal / *Sign up*
Cenário 1 - Cadastrar um usuário:	
Ação	Reação
Um usuário ainda não cadastrado clica no *link Sign up*	O sistema exibe um formulário para o usuário entrar com as informações.
O usuário entra com os dados. E clica no botão 'Registrar'.	O sistema valida as informações. Insere o novo usuário e redireciona para a página de login. Em caso de falha, exibe mensagem de erro na tela.
Cenário 2 - Visualizar usuário:	
Ação	Reação
O usuário já cadastrado e logado no sistema, clicar no link *Minha conta*.	O sistema exibe as informações do usuário.

Tabela 17: UC01 - Cadastrar usuário.

3.3.2 Autenticar usuário no sistema.

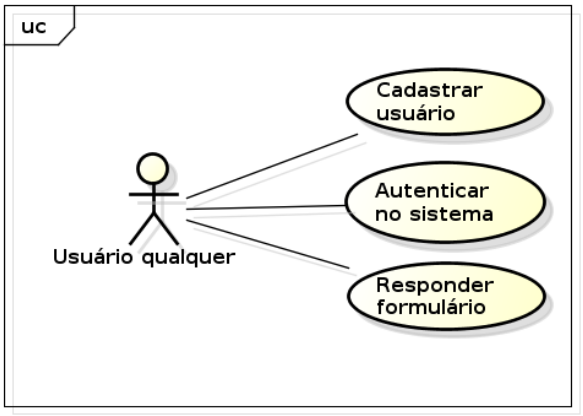

Figura 3: Autenticar usuário no sistema

Código:	UC02.
Nome do UC:	Autenticar usuário no sistema.
Objetivo:	Autenticar um usuário existente no sistema.
Ativação:	Página principal / *Login*
Cenário 1 - Autenticar um usuário:	
Ação	Reação
Um usuário clica no *link Login*	O sistema exibe um formulário para o usuário entrar com as informações.
O usuário entra com os dados. E clica no botão *Login*.	O sistema valida as informações. Autentica o usuário e redireciona para a página de lista de formulários. Em caso de falha, exibe mensagem de erro na tela.

Tabela 18: UC02 - Autenticar usuário no sistema.

3.3.3 Cadastrar formulário.

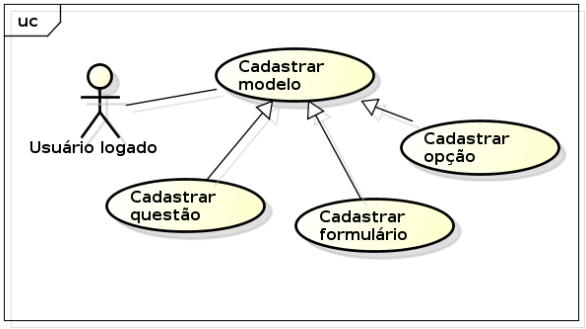

Figura 4: Cadastrar formulário

Código:	UC03.
Nome do UC:	Cadastrar formulário.
Objetivo:	Cadastrar um novo formulário no sistema..
Ativação:	Página principal / Meus formulários / Novo formulário
Cenário 1 - Cadastrar um formulário:	
Ação	Reação
Um usuário clica no botão *Novo formulário*	O sistema exibe um formulário para o usuário entrar com as informações do novo formulário.
O usuário entra com os dados. E clica no botão *Salvar*.	O sistema valida as informações, salva o formulário e redireciona para a página de edição do formulário. Em caso de falha, exibe mensagem de erro na tela.

Tabela 19: UC03 - Cadastrar formulário.

3.3.4 Editar formulário.

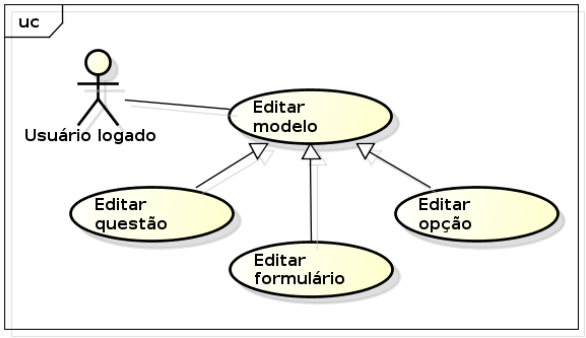

Figura 5: Editar formulário

Código:	UC04.
Nome do UC:	Editar formulário.
Objetivo:	Editar um formulário existente no sistema..
Ativação:	Página principal / Formulário / Editor
Cenário 1 - Editar um formulário:	
Ação	Reação
Um usuário clica no botão *Editor*	O sistema exibe as informações do formulário e uma lista de tipos de questão a ser adicionadas ao formulário.
O usuário clica no botão *Adicionar*, de um tipo específico de questão.	O sistema insere na tela um formulário para o usuário entrar com os dados da questão. Esse formulário deverá ser adicionado no final do formulário.
O usuário entra com o nome da questão e clica em salvar.	O sistema salva a questão, e exibe o feedback para o usuário. Sem recarregar a página.
O usuário clica no botão *Adicionar* uma opção.	O sistema insere na tela uma opção para o usuário entrar com os dados.
O usuário entra com o valor da opção e clica em salvar.	O sistema salva a opção, e exibe o feedback para o usuário. Sem recarregar a página.

Tabela 20: UC04 - Editar formulário.

3.3.5 Responder um formulário.

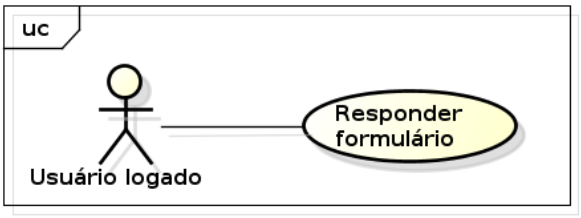

Figura 6: Responder um formulário

Código:	UC05.
Nome do UC:	Responder um formulário.
Objetivo:	Coletar as respostas para um dado formulário.
Ativação:	Página principal / Formulário / Nova resposta
Cenário 1 - Responder a um formulário do tipo contínuo:	
Ação	Reação
Um usuário clica no *link* para responder o formulário	O sistema. exibe as informações do formulário e todas as questões desse formulário.
O usuário responde cada pergunta e clica no botão *Salvar* no final da lista de questões.	O sistema valida as respostas, salva todas e mostra a tela de agradecimento. Exibe mensagem de erro caso a validação não passe ou algum erro aconteça.
Cenário 2 - Responder a um formulário do tipo condicional:	
Ação	Reação
Um usuário clica no *link* para responder o formulário	O sistema. exibe as informações do formulário e a primeira questão do formulário.
O usuário responde a questão e clica em *Próxima questão.*	O sistema valida a resposta e exibe a próxima questão. Exibe mensagem de erro caso ocorra.

Tabela 21: UC05 - Responder um formulário.

3.3.6 Visualizar relatório.

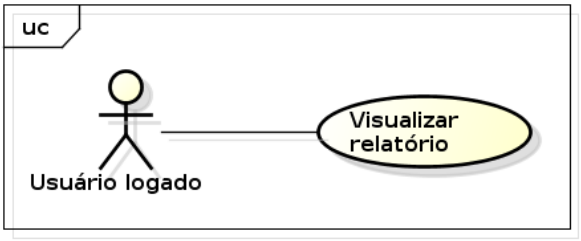

Figura 7: Visualizar relatório

Código:	UC06.
Nome do UC:	Visualizar relatório.
Objetivo:	Relatar todas as respostas para uma dado formulário.
Ativação:	Página principal / Formulário / Relatório
Cenário 1 - Responder a um formulário do tipo contínuo:	
Ação	Reação
Um usuário logado, com um formulário, clica no *link Relatório*.	O sistema exibe as respostas do formulário.

Tabela 22: UC06 - Visualizar relatório.

4 Diagramas

4.1 Diagrama de classes

O seguinte diagrama mostra as classes do sistema e suas cardinalidades.

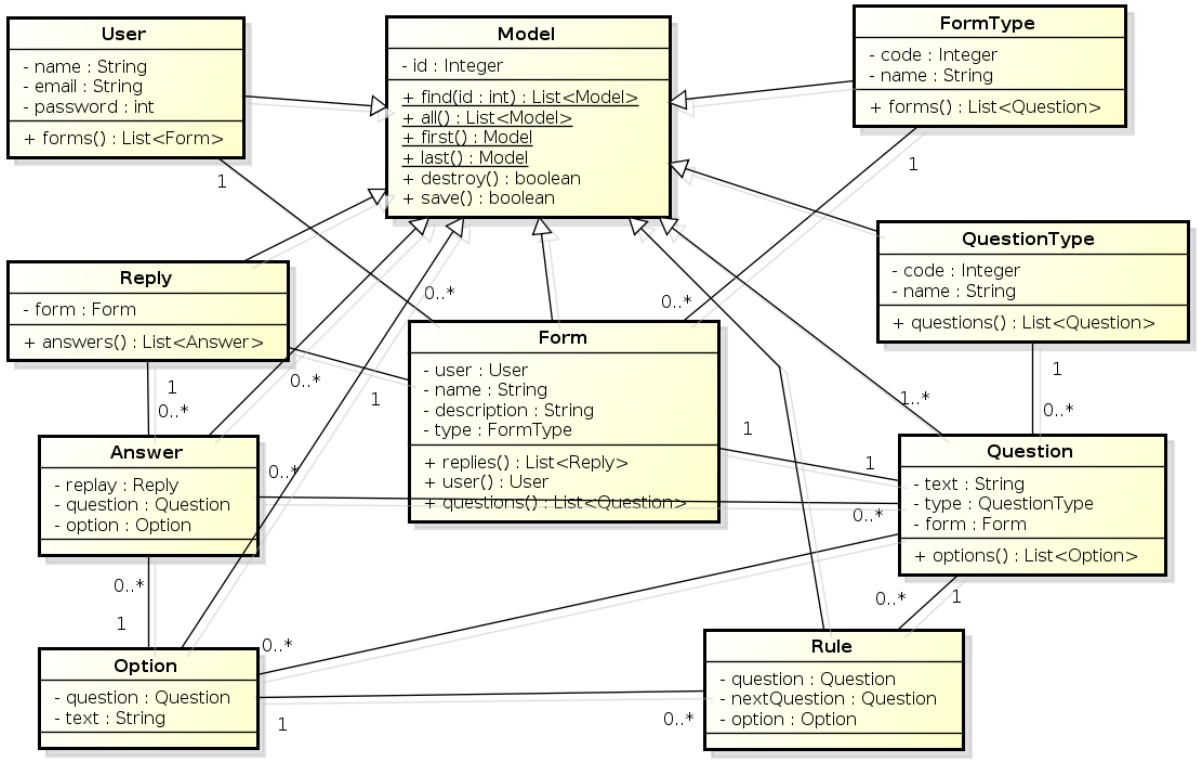

Figura 8: Diagrama de classes

4.2 Diagramas de sequência

4.2.1 Diagrama de sequência da interface REST da aplicação.

O próximo diagrama descreve a sequência de comunicação entre o cliente - navegador, e o serviço REST da aplicação.

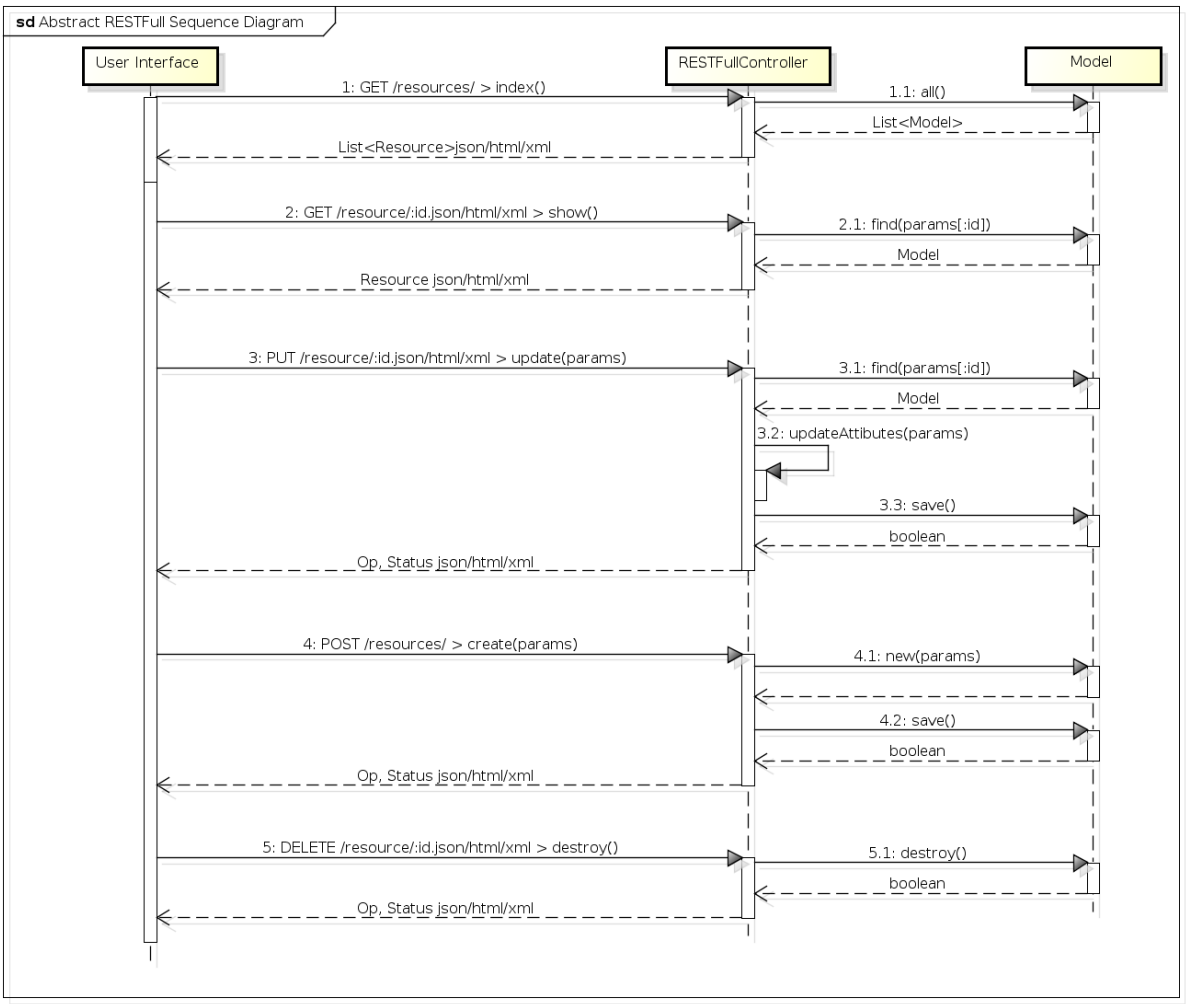

Figura 9: Diagrama de sequência REST

4.2.2 Diagrama de sequência da edição de formulário da aplicação.

O próximo diagrama descreve a sequência de comunicação para edição do formulário.

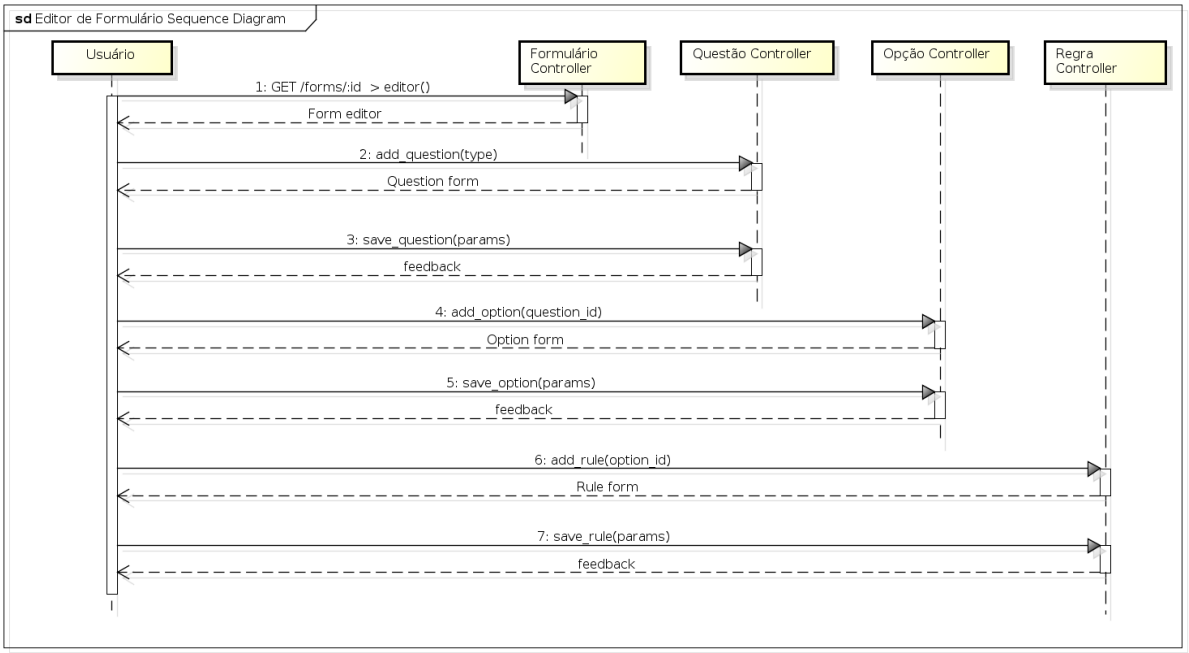

Figura 10: Diagrama de sequência da edição de formuário.

4.3 Diagrama entidade relacionamento

O próximo modelo diagramático descreve, de forma abstrata, o modelo de dados do sistema.

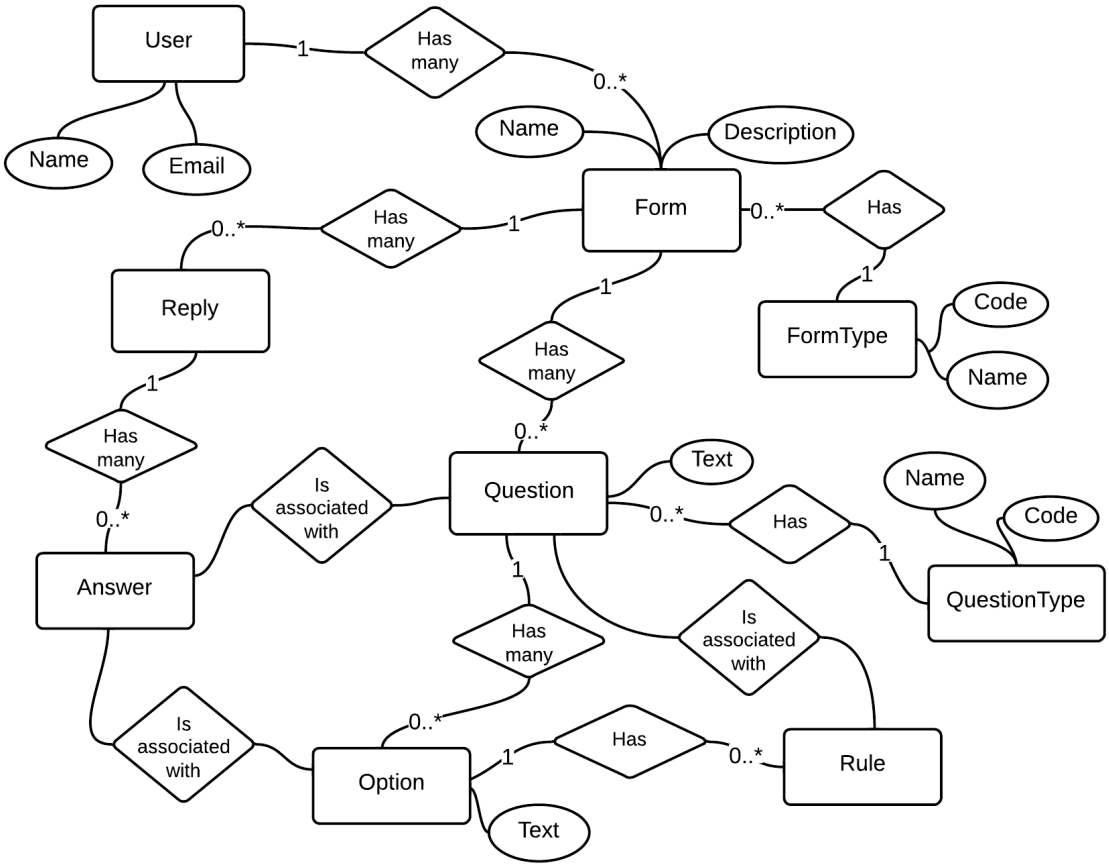

Figura 11: Diagrama entidade relacionamento.

5 Sistema implementado

As próximas seções descrevem e apresentam as principais telas do sistema implementado.

5.1 Tela de cadastro de novo usuário.

A tela que segue é a de criação de um novo usuário. Contém os dados básicos necessários e um campo extra para repetir a senha, com objetivo de garantir que o usuário inseriu a senha que realmente deseja.

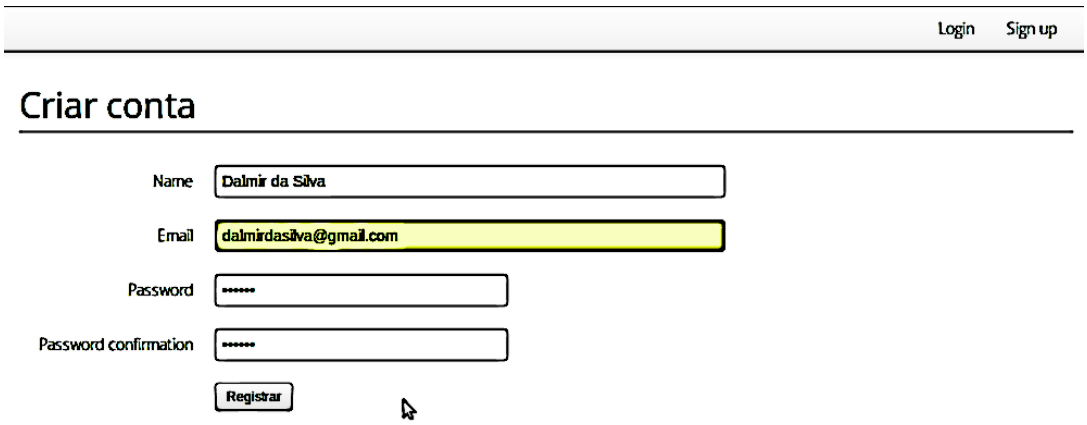

Figura 12: Tela de cadastro de novo usuário.

5.2 Tela de autenticação do usuário.

Nessa tela, um usuário já cadastrado pode se autenticar no sistema.

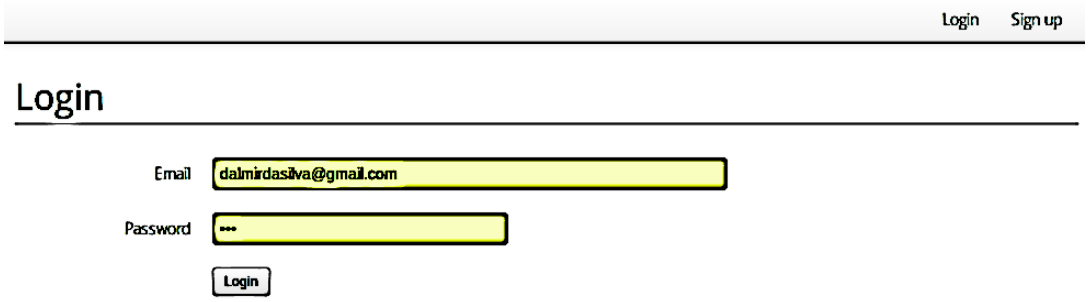

Figura 13: Tela de autenticação do usuário.

5.3 Tela de criação de um novo formulário.

Na tela seguinte, um usuário autenticado insere um novo formulário no sistema. Note que nessa tela apenas os dados básicos do formulário podem ser inseridos, as perguntas e opções são inseridas na tela de edição de formulário.

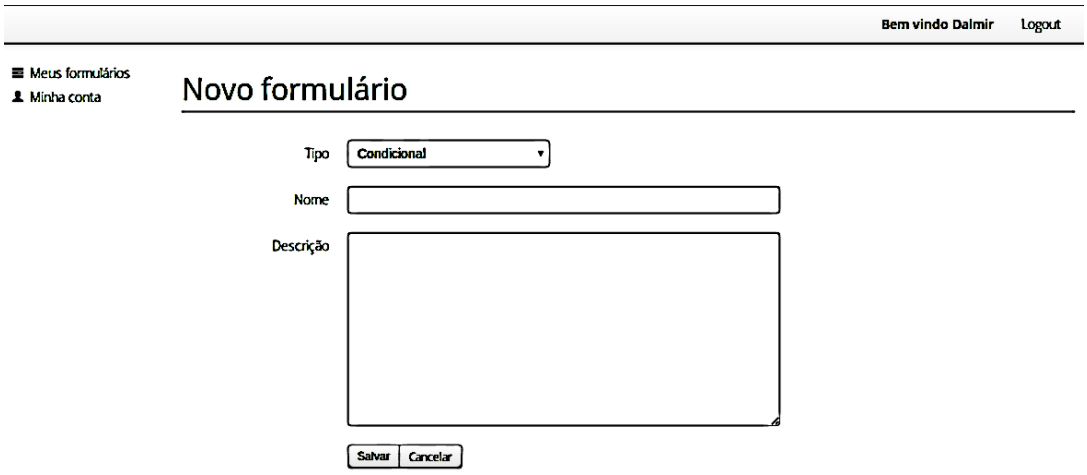

Figura 14: Tela de criação de um novo formulário.

5.4 Tela de edição do formulário.

Essa tela é a principal tela do sistema. Nela, pode-se cadastrar e editar perguntas de um formulário. Perguntas podem ser de diferentes tipos. Na coluna à direita, são listados os diferentes tipos de perguntas. Para adicionar uma nova pergunta ao formulário, basta clicar no botão correspondente à pergunta desejada.

Após adicionar uma pergunta, pode-se adicionar opções de resposta para ela - caso tal pergunta não seja de texto livre. Opções podem ser removidas ou editadas. Para cada opção, pode-se adicionar uma regra. Uma regra direciona o fluxo do questionário. Portanto, na regra pode-se definir o número da próxima pergunta.

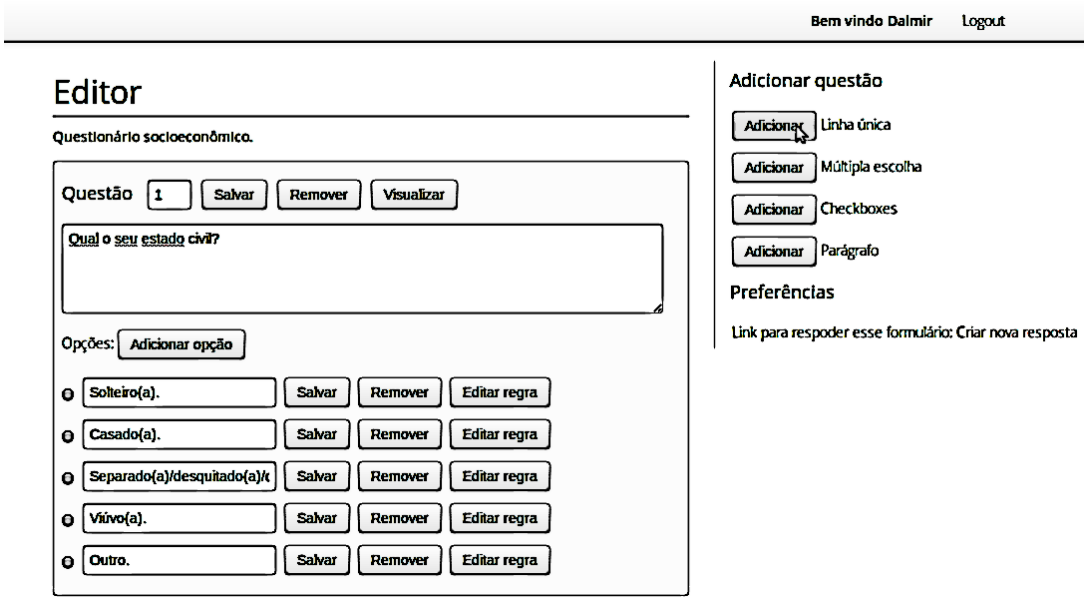

Figura 15: Tela de edição do formulário.

5.5 Tela de resposta à um formulário.

Essa é a tela de resposta para um determinado formulário. Como formulários podem ser contínuos ou condicionais, essa tela pode exibir as questões do formulário de duas maneiras distintas. A figura a seguir, apresenta a tela de resposta de um questionário do tipo condicional. Nela, a primeira pergunta do formulário é exibida e, dependendo da resposta, outras novas perguntas são exibidas secessivamente, se opondo ao modo de exibição de um formulário contínuo, onde todas as perguntas são exibidas de uma só vez.

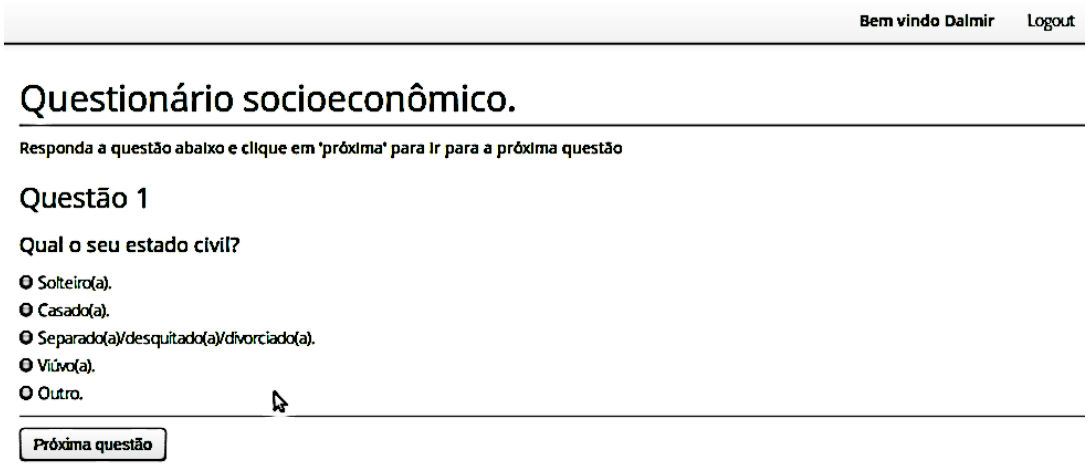

Figura 16: Tela de resposta à um formulário.

5.6 Tela de visualização de relatório.

A próxima tela, é responsável pela visualização do relatório de respostas de um determinado formulário.

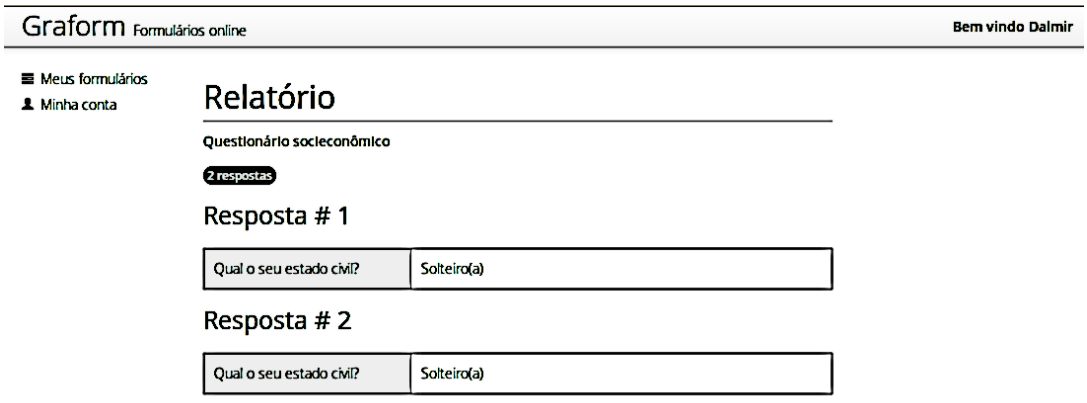

Figura 17: Tela de visualização de relatório.

6 Testes do sistema

6.1 Desenvolvimento orientado a testes (*Test Driven Development*)

Como padrão de testes do projeto, utilizou-se *Test Driven Development* (TDD). TDD é um padrão de testes no qual os testes são criados antes da implementação. Sendo assim, os testes orientam a codificação (como o nome já diz). Primeiro cria-se um teste, depois criamos uma implementação para fazer o teste passar [13].

6.1.1 *Rspec*

TDD é uma proposta de desenvolvimento de software, e não uma ferramenta de teste. Para isso, utilizou-se o *Rspec*. *Rspec* é um conjunto de ferramentas para testar a aplicação, direcionada para o *framework* Ruby on Rails.

Como o Ruby on Rails é um *framework* MVC, o *Rspec* provê um feramental para testar cada camada da aplicação.

6.2 Testes dos modelos

6.2.1 Usuário

No próximo trecho de código, podemos ver um exemplo de teste do modelo Usuário:

```
require 'spec_helper'

describe User do

  before :each do
    @user = User.new
  end

  describe "associations" do

    it 'should be associated with forms' do
      @user.should respond_to(:forms)
    end
  end

  describe "attributes" do

    it "should have name" do
      @user.should respond_to(:name)
    end

    it "should have email" do
      @user.should respond_to(:email)
    end

    it "should have password" do
      @user.should respond_to(:password)
    end
  end
end
```

Neste teste, procura-se testar as relações do modelo com as outras classes do sistemas, além de seus atributos.

Quando rodamos esse teste usando o *rspec*, temos a seguinte saída como resultado, mostrando que todos os testes passaram:

```
rspec --format nested spec spec/user

User
  associations
    should be associated with forms
  attributes
    should have name
    should have email
    should have password

Finished in 0.03836 seconds
4 examples, 0 failures
```

6.2.2 Fomulário

A seguir, vemos os testes do modelo Formulário. Esse modelo possui, além dos testes de relacionamento e atributos, um método especial: *max_question_number*.
Esse método é responsável por retornar a questão desse formulário com maior número.

```
require 'spec_helper'

describe Form do

  before :each do
    @form = Form.new
  end

  describe "associations" do

    it 'should be associated with questions' do
      @form.should respond_to(:questions)
    end

    it 'should be associated with replies' do
      @form.should respond_to(:replies)
    end
  end

  describe "attributes" do

    it "should have a name" do
      @form.should respond_to(:name)
    end

    it "should have a description" do
      @form.should respond_to(:description)
    end
  end

  describe "#max_question_number" do

    it "should return the higest number of the existing questions" do
      @form.id = 1
      q0 = Question.new(form_id: @form.id, number: 0)
      q1 = Question.new(form_id: @form.id, number: 5)
      q2 = Question.new(form_id: @form.id, number: 4)
      @form.max_question_number.should == q1
    end
  end
end
```

O resultado do teste:

```
rspec --format nested spec spec/form

Form
  associations
    should be associated with questions
    should be associated with replies
  attributes
    should have a name
    should have a description
  #max_question_number
    should return the higest number of the existing questions


Finished in 1.03234 seconds
5 examples, 0 failures
```

A seguir, vemos os resultados de todos os testes rodados de uma só vez:

```
rspec --format nested

User
  associations
    should be associated with forms
  attributes
    should have name
    should have email
    should have password

Form
  associations
    should be associated with questions
    should be associated with replies
  attributes
    should have a name
    should have a description
  #max_question_number
    should return the higest number of the existing questions

FormType
  associations
    should be associated with forms
  attributes
    should have a name
    should have a code

Reply
  associations
    should be associated with answers
    should be associated with form

Question
  associations
    should be associated with rules
    should be associated with options
  attributes
    should have a text
    should have a form
    should have a type
  #next_question
    when the form is continuous
      should get the next based on number
    when the form is conditions
      should get the next based on answer and the rule
```

7 Conclusão

Encontramos, ao longo do desenvolvimento desse projeto, inúmeras circunstâncias onde as técnicas de desenvolvimento de software, estudadas durante o curso, foram fundamentais para prover a melhor solução para cada problema encontrado.

Técnicas, como TDD, propiciaram um desenvolvimento ágil e com boa taxa de cobertura de código. Assim como o uso do padrão MVC ajudou no objetivo de atingir baixo acoplamento e alta coesão das classes do sistema.

A utilização de uma aplicação web, possível de ser acessada por qualquer computador, sem a necessidade de instalação de um software *desktop*, aumentou a produtividade da empresa como um todo. Também, houve uma redução de incidências de erros humanos, devido ao fato de não ser mais necessário o uso de instruções SQL diretamente na base de dados para a criação e gerenciamento dos formulários.

Referências

[1] Rdp Benjamin Franklin. Create, publish, and analyze your surveys & questionnaires in a few minutes! (http://www.evalandgo.com/), 2013.

[2] David Heinemeier Hansson. Rails - web development that doesn't hurt (http://rubyonrails.org/), 2013.

[3] Google Inc. Formulários - faça enquetes ou escale equipes rapidamente com um simples formulário on-line (http://www.google.com/), 2013.

[4] The jQuery Foundation. jquery - write less, do more (http://jquery.com/), 2005.

[5] Canonical Ltd. Ubuntu is the openstack favourite (http://www.ubuntu.com/), 2013.

[6] Matz. A programmer's best friend (http://www.ruby-lang.org/), 2001.

[7] Nginx. Make your web site fast and reliable (http://pt.wikipedia.org/wiki/nginx), 2013.

[8] Oracle. The world's most popular open source database (http://www.mysql.com/), 2013.

[9] R. THOMAS SAM. *Agile Web Development with Rails*. Pragmatic Bookshelf, 2008.

[10] Toni Santo-Regis. Latex - a document preparation system (http://www.latex-project.org/), 2013.

[11] Surveymonkey. Crie seus próprios questionários online (http://pt.surveymonkey.com), 2013.

[12] A. TENENBAUM. *Estruturas de Dados em C*. Makron Books, 1997.

[13] R. URUBATAN. *Ruby on Rails: Desenvolvimento Fácil e Rápido de Aplicações Web*. São Paulo, 2009.

[14] M. VIANNA. *Estrutura de Dados: Conceitos e Técnicas de Implementações*. 2000.

[15] Wikipedia. Git (http://en.wikipedia.org/wiki/git_(software)), 2013.