

Monografia em Sistemas de Informação

Relatório técnico



Autor : João Paulo Martins Castanheira
Matrícula : 2010054584
Orientador : Clarindo Isaias Pereira da Silva e Pádua

Sistema Gerenciador de Provas e Concursos Públicos

Índice:

Introdução	3
Estudo de Viabilidade	5
Motivação	7
Estruturação	8
Implementação	11
Conclusão	25

Introdução

Esta monografia teve por objetivo, construir um sistema levando em consideração todas as etapas de desenvolvimento de software detalhados nos tópicos seguintes. O propósito do software é prover uma interface que dê subsídios para o usuário, cadastrar, editar e visualizar provas de concursos profissionais que realizou pretende realizar, além disto também irá fornecer ferramentas para gerenciamento de compromissos, relacionados ou não com os concursos. O sistema deve prover uma interface amigável e de fácil uso para que o usuário gerencie seus concursos vigentes e passados, bem como provendo as funcionalidades necessárias para alertar o usuário quando uma prova e/ou compromisso se aproxima. Mais funcionalidades serão estudadas nas próximas etapas de desenvolvimento do software, podendo haver a possibilidade de uma interface que exponha estatisticamente dados acerca do tópico.

O problema é importante, por dois motivos. Primeiramente é um projeto que me permitiu exercer na prática conceitos multidisciplinares de desenvolvimento de software. Abrangendo por exemplo conceitos de engenharia de software, usabilidade, interação humano computador, bancos de dados e várias outras. Por outro lado, é importante para mim, especificamente, dado que é uma demanda que me foi atribuída. Portanto este projeto é uma interseção acadêmica/profissional, que futuramente pretendo comercializar, mesmo que para poucas pessoas.

O software não será um protótipo, mas uma versão final para desktops, plataforma que julguei mais adequada neste primeiro momento. A evolução deste sistema, por exemplo, para outras plataformas, não será tratada nesta monografia.

1. Referencial Teórico:

A proposta formulada nesta monografia, não faz parte de um universo genérico de soluções, nem mesmo de ampla utilização. É um trabalho voltado para resolução de um problema específico, para um público específico. No mercado podemos observar, dezenas de ferramentas com propósitos semelhantes, como o Google Keep, Evernote, Google Agenda etc. Porém percebo uma tendência de generalização de funcionalidades, para atingir o maior número de usuários possível. Portanto, mesmo ainda estando em um estágio inicial na pesquisa de

trabalhos relacionados, creio que ferramentas semelhantes, sejam desenvolvidas sob demanda e de forma personalizada para o cliente.

Tendo dito isso, justifico a execução de um trabalho de natureza tão específica, observando a riqueza de oportunidades que este me possibilita praticar os conhecimentos adquiridos no curso de Sistemas de Informação. Afinal de contas, esta monografia se propõe a criar um Sistema de Informação. Todas as áreas de conhecimentos adquiridas no curso são de certa forma aplicáveis a este projeto. Porém existe uma perspectiva de ênfase na área de Engenharia de Software em toda sua formalidade.

Minha principal referência para conduzir o trabalho, será o livro “Engenharia de Software” do autor *Ian Sommerville*. O desenvolvimento de software é tratado com detalhes na obra, citando técnicas variadas para cada tipo de sistema, com etapas formais e bem definidas. Fica claro que o desenvolvimento de um sistema de informação é muito mais que escrever códigos. Utilizarei uma metodologia de desenvolvimento ágil que se chama Scrum. É uma metodologia de desenvolvimento ágil que trabalha com ciclos de desenvolvimento e com metas bem definidas em intervalos curtos de tempo, focando no cliente ao invés de processos, com documentação não tão rígida, mas um software flexível que aceite mudanças. Acredito que o modelo possa me ajudar a manter o controle sobre a previsibilidade de conclusão das etapas previstas.

2. Histórico de Desenvolvimento:

- 10/04 - Levantamento de requisitos
- 25/04 - Criação de um diagrama de classes
- 30/04 - Modelagem do banco de dados
- 07/05 - Desenvolvimento da interface
- 15/05 - Desenvolvimento das classes e regras de negócios
- 07/06 - Homologação do sistema
- 10/06 - Liberação da versão Beta
- 12/06 - Possíveis correções e melhorias
- 20/06 - Entrega do produto para o cliente
- 30/06 - Apresentação de resultados (pôster)
- 05/07 - Entrega de relatório técnico

Estudo de viabilidade feito antes de iniciar o projeto

O projeto de monografia para o Bacharelado de Sistemas de Informação, é feito considerando uma disciplina de 6 créditos ou 90 horas. Seguindo o cronograma da disciplina, procedi em direção a receber todas as orientações, planejar um projeto, procurar e conversar com um orientador capacitado para o perfil do projeto e validá-lo.

Após estas etapas iniciais o projeto se iniciou de fato no dia 10/04/2017 e dado o cronograma deve ser encerrado no dia 20/06/2017. Considerando como o encerramento um release de executável com o produto final para o cliente em questão. Portanto o projeto teve que se concretizar em um intervalo de 70 dias.

Para analisar a viabilidade temos que prever o escopo do projeto junto ao cliente, analisar restrições como, tecnologia atual, cronograma, orçamento etc. Como destacado na proposta da monografia anteriormente, os requisitos funcionais básicos já estão definidos, tendo isso em mente a expectativa é de que quaisquer requisitos funcionais incrementais, sejam minoritários no montante de horas destinados para codificação do sistema.

Requisitos básicos iniciais

1. Log in
2. Cadastrar de usuário
3. Cadastrar concurso
4. Visualizar concursos cadastrados.
 - Exibição do edital do concurso em PDF dentro da ferramenta.
 - Layout customizável
 - Tela de tamanho dimensionável.
 - Filtrar entre concursos vigentes/passados
 - Ordenar arbitrariamente a exibição dos concursos.
5. Armazenar/exibir dados estatísticos de salários e valores de inscrição.
6. Enviar alerta por e-mail e/ou sms quando o concurso estiver se aproximando.

O projeto foi feito única e exclusivamente por mim, portanto terá custo zero e irei utilizar de duas tecnologias diferentes para integrar sistema e banco de dados. A linguagem de programação escolhida foi a linguagem

FoxPro 9, construída pela MicroSoft. Possuo relativa intimidade com a linguagem por trabalhar com ela há 5 anos e se trata de uma linguagem com recursos que suportam o desenvolvimento orientado a objetos que era uma de minhas prioridades. Dito isso acredito ser sensata a escolha, pois um dos recursos menos abundantes que tenho à disposição é o tempo. O banco de dados, por ser relativamente simples foi modelado usando a ferramenta MySql, gratuita e fácil de lidar.

Cada uma das telas se comunicam com as classes que foram desenvolvidas. As interfaces foram desenvolvidas em aproximadamente oito horas de trabalho. Antes de incluir os requisitos extras, as classes que considerei desenvolver eram:

1. Pessoa
2. Concurso
3. Email
4. PDF
5. SMS
6. VISUALIZADOR (suporta a interface de visualização)

Algumas das classes são mais complexas que outras, porém designei um tempo de 10 horas de desenvolvimento para cada classe, somando assim 68 horas totais se considerarmos o desenvolvimento das interfaces.

Após a etapa de desenvolvimento, o software deve ser testado e corrigido, quando eventuais erros forem encontrados. Minha estimativa é que 50% do tempo de desenvolvimento seja destinado aos testes e correções do sistema, sendo assim previ que seriam gastas 34 horas nos testes e correções.

Portanto o sistema como um todo acarretaria em 102 horas de desenvolvimento para serem executados em 70 dias, o que pode ser considerado um sistema viável. Porém não estava considerando a modelagem e criação do banco de dados, construção das especificações de requisitos de usuário e sistema nem o projeto de software nesta estimativa. Entretanto estas etapas desconsideradas na estimativa não inviabilizaram a concretização do sistema, por se tratarem de etapas consideravelmente mais rápidas que o próprio desenvolvimento de software.

A conclusão deste documento é formalizar a declaração deste projeto como viável e sem nenhuma restrição dados os requisitos funcionais levantados anteriormente.

Requisitos incrementais incluídos a posteriori

Após o início da codificação do sistema, um cliente requisitou uma funcionalidade incremental no sistema. A funcionalidade requisitada era de relativa simplicidade e por isso não hesitei em incluí-la, pois ao meu ver enriqueceria o trabalho.

O cliente requisitou que o sistema incluísse um cadastro de tarefas, não necessariamente relacionadas aos concursos e uma forma de visualizá-las e gerenciá-las. Entretanto era importante que as tarefas fossem classificadas conforme a sua urgência e prioridade, como também integrar com a ferramenta de envio de e-mail já presente no levantamento de requisitos inicial. Irei descrever neste documento como estas funcionalidades foram inseridas no software.

Outro requisito também foi incluído na especificação. Esta funcionalidade especificamente, foi incluída por mim mesmo, pois julguei interessante que a ferramenta pudesse suportar múltiplos usuários. Desta forma, imaginando que a aplicação poderia estar instalada em diferentes terminais, como em uma biblioteca de uma universidade ou qualquer outra máquina. A portabilidade desta aplicação se daria na migração do banco de dados para as nuvens, porém este passo demandaria que eu criasse ou alugasse um servidor que ficaria disponível para todos os usuários. Portanto esta etapa não faz parte do escopo deste trabalho.

Motivação

Além de se um aluno do curso de Sistemas de Informação da UFMG, eu também sou um desenvolvedor contratado em uma software house de Belo Horizonte. O meu cargo na empresa me permite observar o trabalho de outros profissionais envolvidos no processo de criação e manutenção de um sistema de software.

Porém cumpro um papel com prerrogativas bem limitadas, onde eu recebo um documento pronto, com o levantamento de requisitos e especificação já feitas. Nesta parte do processo o cliente já foi consultado, os analistas de software já desenvolveram o seu papel, o administrador do

banco de dados já criou os subsídios necessários para que a customização seja compatível com a estrutura do banco existente e me resta apenas codificar as funcionalidades requisitadas e repassar o produto final para uma equipe de QA (quality assurance) que autorizam ou não a incorporação das minhas customizações no produto final, com base em testes automatizados e manuais.

Tendo dito isto, a escolha de desenvolver um sistema de software do zero, me motivou. Esta me pareceu uma excelente oportunidade de eu entender as dificuldades e necessidades que outros profissionais da área de TI, senão programadores, encontram nas tarefas de seu dia a dia. O que de fato se concretizou, encontrei vários problemas que normalmente não estou habituado a lidar.

Outro aspecto interessante, é que apesar de eu não trabalhar diretamente com as tarefas supracitadas, ao longo do curso de Sistemas de Informação, todos esses aspectos relacionados a construção de software foram abordados em algum momento do curso. O que apenas aumentou a motivação de desenvolver um projeto como este, pois assim poderia colocar em prática, conhecimento que muitas vezes nos foi passado apenas na teoria.

Por fim, eu não gostaria de desenvolver um software obsoleto sem nenhum cliente. Então apesar de concursos públicos, não serem uma área de interesse minha, eu escolhi este tema pois existiam clientes dispostos a utilizar o sistema uma vez que ele estivesse pronto.

Estruturação

Banco de dados – Estrutura Básica

Cadastro de Usuário:

Name	Type	Width
id	Integer (AutoInc)	4
nome	Varchar	200
usuario	Varchar	200
senha	Varchar	200
email	Varchar (binary)	200

Cadastro de Concursos:

Name	Type	Width
id	Integer (AutoInc)	4
sel	Logical	1
del	Logical	1
concurso	Varchar	254
dataprova	Date	8
cargo	Varchar	254
endereco	Varchar	254
compl	Varchar	254
horaprova	Character	5
salario	Float	15
homol	Date	8
inscricao	Float	15
vencimento	Date	8
banco	Varchar	254
obs	Varchar	254
link	Varchar	254
pdf	Varchar	254
alerta	Integer	4
idusuario	Integer	4

Cadastro de Tarefas:

Name	Type	Width
id	Integer (AutoInc)	4
check	Integer	4
descricao	Varchar	254
data	Date	8
hora	Character	5
prioridade	Character	20

Informações complementares acerca da estrutura do banco de dados

O banco de dados ainda armazena dados acerca das dimensões editáveis de todas as telas do sistema. Vou omitir suas estruturas por não considerar que agregue valor. O funcionamento básico gira em torno de uma tabela com campos vazios, a princípio. No entanto é a partir da primeira manipulação de qualquer uma das dimensões da tela, seja o tamanho da tela em si ou a disposição das colunas nas grades das telas de visualização, que o programa verifica que houve alteração e armazena todos os atributos da interface, para que ao reabrir as mesmas configurações se mantenham.

Classes

A lista do diagrama completo das classes com todos os seus atributos e funções, estenderia muito este documento, portanto irei disponibilizar um link com todos os arquivos envolvidos no desenvolvimento do software no *GitHub* e abaixo irei listar e resumidamente descrever as principais funcionalidades das classes mais importantes.

Lista das principais classes:

- Cadastro de Concurso:
 - Documento em que seus atributos são análogos aos campos do cadastro ao qual se refere. Suporta a inclusão, edição e exclusão dos registros, por traz da interface do cadastro, que apenas se comunica com a classe através de suas funções públicas.
 - Cada instância de um objeto desta classe é um concurso em particular.
- Visualização de Concursos:
 - Classe por traz da interface de visualização, suportando todas as interações entre a tela e o banco de dados, intermediando os acessos.
 - Suporta também a organização dinâmica dos objetos nas telas, provenientes de manipulações dos usuários.
- Tarefa:
 - Análoga a classe de concursos.
- Visualização de Tarefas:
 - Análoga a classe de visualização de concursos.
- PDF (programa myviewer.prg)
 - Instanciada na interface de visualização de concursos, suporta todas as operações do gerenciador de PDFs da aplicação.
 - Zoom, salvar como, imprimir, abrir com etc.
- Classe Pessoa:
 - Pessoa qualquer, com todos os dados cadastrais de um usuário. Ainda não houve a necessidade da criação de uma subclasse, que diferencie um usuário final de um outro usuário qualquer, pois o grupo de clientes se limita à apenas um perfil.
- E-mail:

- Classe que envia todos os e-mails aos usuários relativos aos alertas cadastrados, tanto para o concurso, quanto para as tarefas.

Informações complementares acerca das classes:

Como a arquitetura do software é do tipo MVC, todas as telas do sistema são suportadas por uma classe. As interfaces gráficas não contemplam nenhuma regra de negócio tampouco acesso direto ao banco de dados. Portanto existem outras classes que ajudam a manipular estas telas, cujas implementações possam ser simples demais para descrever aqui, porém fundamentais para não ferir a arquitetura proposta.

Link para acesso aos programas no GitHub:

[Repositório Git](#)

Implementação

Lista de funcionalidades

Login

- Objetivos
- Protótipos
- Regras de Interação

Cadastro de Usuário

- Objetivos
- Protótipos
- Regras de Interação
- Regras de Negócio

Cadastro de Concursos

- Objetivos
- Protótipos
- Regras de Interação
- Regras de Negócio

Visualização de Concursos

Objetivos
Protótipos
Regras de Interação

Cadastro de Tarefas

Objetivos
Protótipos
Regras de Interação
Regras de Negócio

Visualização de Tarefas

Objetivos
Protótipos
Regras de Interação
Regras de Negócio

Declaração da Visão de Negócio

O problema do cliente	<i>O cliente não possui um modo informatizado que o permita organizar e gerenciar os seus concursos públicos, tampouco contemplar os concursos realizados no passado. Desta forma acaba tendo que fazê-lo de diversos meios diferentes, tanto físicos quanto digitais. Nenhuma das ferramentas utilizadas é julgada satisfatória, por não ter recursos específicos para a realização das tarefas necessárias.</i>
Afeta	<i>Todo o gerenciamento de informações sobre os concursos realizados e a realizar do cliente.</i>
Impacto	<i>Não deixar que surja a necessidade de armazenar qualquer tipo de informação fora da plataforma.</i>
Solução a ser adotada	<i>O propósito do software será prover uma interface que dê subsídios para o usuário, cadastrar, editar e visualizar provas de concursos profissionais que prestou e/ou pretende realizar. O sistema deve prover uma interface amigável e de fácil uso para que o usuário gerencie seus concursos vigentes e passados, bem como prover as funcionalidades necessárias para alertar o usuário quando</i>

	<i>uma prova se aproxima. Os requisitos menos importantes e não funcionais adicionais serão explorados em detalhes ao decorrer deste documento.</i>
--	---

Lista de funcionalidades

1. Login
2. Cadastro de Usuários
3. Gerenciador de Concursos – Cadastro de Concursos
4. Gerenciador de Concursos – Visualização de Concursos
5. Gerenciador de Concursos – Cadastro de Tarefas
6. Gerenciados de Concursos – Visualização de Tarefas
7. Gerenciador de Concursos – Envio de e-mail
 - 7.1. Funcionalidade implícita

Legenda:

- **RN:** regra de negócio
- **RI:** regra de interação

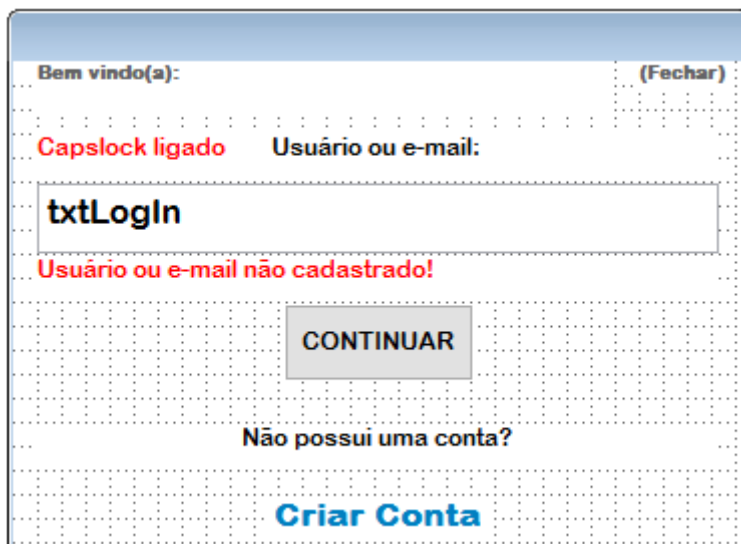
1. Login

Objetivos:

Deve ser exibida uma tela de conferência de entrada de usuário. Desta forma além de proteger os dados pessoais do usuário contra terceiros, também permitirá que múltiplos perfis se cadastrem na plataforma, sem compartilhamento de informações.

Protótipos:

Imagem 1: tela de login.



Bem vindo(a): (Fechar)

Capslock ligado Usuário ou e-mail:

txtLogin

Usuário ou e-mail não cadastrado!

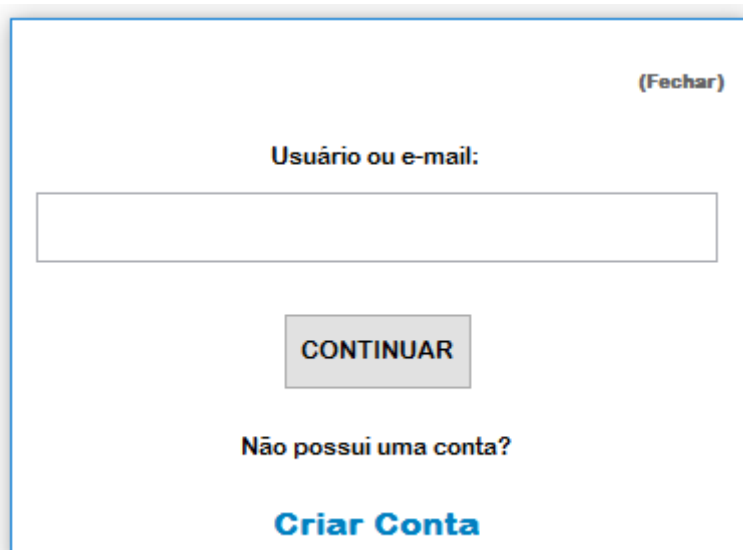
CONTINUAR

Não possui uma conta?

[Criar Conta](#)

Observação: As regras RN001, RN002 e RN003 se dão com a tela no estado da imagem 2.

Imagem 2: tela de login para inserir e-mail ou nome do usuário.



(Fechar)

Usuário ou e-mail:

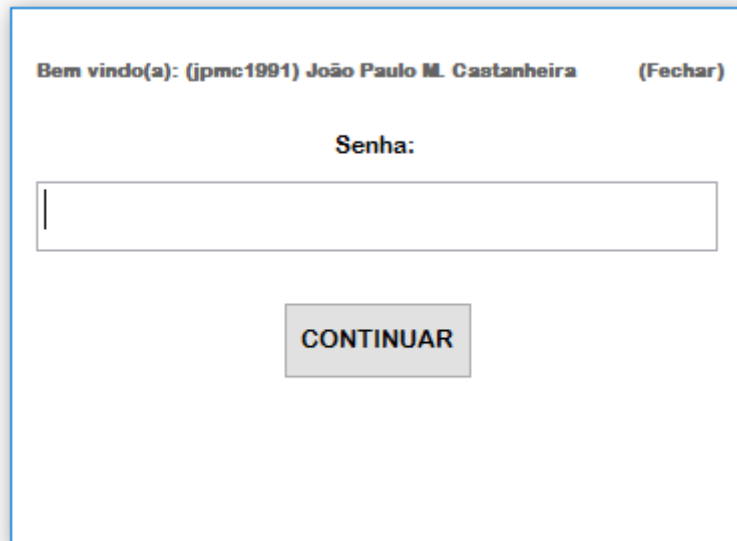
CONTINUAR

Não possui uma conta?

[Criar Conta](#)

Observação: As regras RN004, RN004 e RN005 se dão com a tela no estado da imagem 2.

Imagem 2: tela de login para inserir senha do usuário.

A screenshot of a login form. At the top, it says 'Bem vindo(a): (jpmc1991) João Paulo M. Castanheira' followed by a '(Fechar)' link. Below this is the label 'Senha:' and a text input field. At the bottom of the form is a button labeled 'CONTINUAR'.

Regras de interação:

Observação: alguns campos na tela podem estar **ocultos**, dependendo do **estado** em que o processo de login se encontra.

RI001: O campo designado para a digitação, deve aceitar tanto o e-mail cadastrado quanto o nome do usuário (username) escolhido pelo cliente.

Observação: o cadastro do usuário será explorado posteriormente neste documento.

RI002: Caso o e-mail ou nome do usuário digitado não seja válido, uma mensagem deve se apresentar, alertando que o e-mail/nome informado é inválido.

RI003: Após preencher o campo de login, para prosseguir basta apertar enter ou clicar no botão continuar.

RI004: caso o usuário esteja inserindo a senha e o comando “*caps lock*” esteja ativado, um alerta deve ser mostrado na tela, até que se digite algo com o mesmo, desativado.

RI005: Os caracteres digitados ao informar a senha devem ser ocultados, mostrando apenas asteriscos ao invés de letras e números.

RI006: Se a senha digitada for inválida, um alerta deve ser mostrado na tela. Caso contrário o programa segue sua execução normal.

2. Cadastro de Usuário

Objetivos:

A tela de cadastro de usuário, deve fornecer uma maneira simples de o cliente novato se inscrever na aplicação. Desta forma ele terá que fornecer ao programa algumas informações pessoais para que o sistema interaja com ele na melhor forma possível.

Protótipos:

Imagem 4: tela de cadastro de usuário.

Imagem 4: Protótipo de tela de cadastro de usuário. A interface possui um título "Cadastro de Usuários" no topo. Abaixo dele, há quatro campos de entrada rotulados "Nome:", "E-mail:", "Usuário:" e "Senha:". Os campos "Nome:" e "E-mail:" são mais longos que os outros. No canto inferior direito, há dois botões: "Salvar" e "Cancelar".

Imagem 5: tela de cadastro de usuário, expondo a localização onde os alertas serão emitidos.

Imagem 5: Protótipo de tela de cadastro de usuário com uma grade de fundo. A interface possui um título "Cadastro de Usuários" no topo. Abaixo dele, há quatro campos de entrada rotulados "Nome Completo:", "E-mail:", "Usuário:" e "Senha:". Os campos "Nome Completo:" e "E-mail:" são mais longos que os outros. No canto inferior direito, há dois botões: "Salvar" e "Cancelar". Abaixo dos campos, há uma área rotulada "Mensagem de erro" em vermelho.

Regras de interação:

RI007: todos os campos têm preenchimento obrigatório.

RI008: o usuário deve poder preencher os campos como quiser. Ao clicar em “*Salvar*” uma mensagem de erros deve ser exibida para o primeiro erro encontrado.

Observação: os campos serão validados nesta ordem: Nome Completo, E-mail, Usuário e Senha. Caso exista mais de uma inconsistência, apenas uma será mostrada na ordem sugerida.

Regras de negócio:

RN001: o campo e-mail deve ter validações simples, para garantir que não seja um endereço inválido. Como por exemplo:

- Não poderá conter espaços.
- Deve obrigatoriamente conter um “@”.
- Deve ser maior que quatro caracteres.
- Deve ter pelo menos um ponto final após o “@”.
- Não pode ser igual a um e-mail já cadastrado
- Etc

RN002: o campo “*Usuário*” não pode coincidir com um nome já cadastrado.

3. Cadastro de Concursos

Objetivos:

A tela de cadastro de concursos deve prover uma interface de cadastro, alteração e exclusão de provas. Para alterar um concurso existente o usuário deve ser capaz de selecionar qualquer prova cadastrada. Os campos necessários para o cadastro são exibidos na imagem a seguir.

Protótipos:

Imagem 6: tela de cadastro de concursos

Cadastro de Concursos

Concurso:

Cargo:

Data da Prova:

Endereço:

Complemento:

Horário da Prova:

Salário:

Homologação: Antecedência do Alerta em dias:

Valor da Inscrição:

Vencimento Boleto:

Banco:

Observações:

Link para o edital:

Diretório do PDF: Arquivo não cadastrado.

Observação: os botões ocultos são da esquerda para direita, salvar, editar e cancelar edição respectivamente. O ocultamento dos botão se dão em função do estado de interação que a tela se encontra.

Imagem 7: tela de seleção de concursos para edição, ao clicar em “Selecionar”.

Sel.	Del.	Concurso
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	abi
<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/>	<input type="checkbox"/>	

Regras de interação:

RI009: no cadastro, nenhum campo é obrigatório. É apenas obrigatório que se informe pelo menos um campo.

RI010: para editar um concurso, o usuário deve clicar no botão “Selecionar” que a tela indicada na imagem 7 irá se apresentar.

RI011: todos os concursos já cadastrados irão aparecer nesta grade, e para selecionar um deles, o usuário deve clicar no respectivo campo da coluna “Sel.” (selecionar).

RI012: para deletar um concurso, o usuário deve clicar no respectivo campo da coluna “Del.” (deletar).

Observação: ao clicar em deletar, uma tela de confirmação deve ser apresentada para garantir a correção da ação.

RI013: na tela de cadastro, ao clicar no botão de título “Diretório do PDF”, deve ser aberta uma tela para que o programa guarde o local do arquivo localizado no computador do usuário.

Regras de negócio:

4. Visualização de Concursos

A tela de visualização deve reunir todos os concursos já cadastrados pelo usuário, em uma tela de exibição. Esta tela deve oferecer alguns recursos para que o usuário edite como visualizar seus concursos. Recursos como, edição no tamanho dos campos na grade, redimensionamento da tela, ordenação da grade por todos os campos em ordem crescente etc.

Protótipos:

Imagem 8: tela de visualização de concursos.

[illegible]

Regras de interação:

RI014: ao clicar em qualquer cabeçalho das colunas, a visualização deve alternar entre ordenação crescente e ordenação decrescente respeitando a respectiva coluna.

RI015: no grupo de opções “*Visualizar*” deve-se fazer ao clicar em:

- Todos
 - Exibir todos os concursos cadastrados.
- Vigentes
 - Exibir somente os concursos, cuja data da prova tenha data igual ou superior à data atual.
- Passados:
 - Somente os concursos com datas menores que a data atual.

Observação: as datas não levam em consideração a hora da prova.

RI016: os campos estatísticos sobre os salários e os valores de inscrição, não devem levar em consideração cadastros com os respectivos campos vazios. Fazendo os cálculos apenas com os campos preenchidos.

RI017: quando o número de concursos exibidos se alterar, em decorrência das regras descritas no campo **RI015**, os valores estatísticos devem ser recalculados, de forma a contemplar apenas o que é exibido na tela.

5. Cadastro de Tarefa

Objetivos:

A tela de cadastro de tarefas tem o propósito de armazenar no banco de dados todas as tarefas que o usuário cadastrar.

Protótipos:

Imagem 9: tela de cadastro de tarefas

Cadastro de Tarefas

Descrição	Tirar xerox do CPF e carteira de identidade		
Data	25/06/2017		
Hora	12:00		
Prioridade	1 - Máxima		

Salvar Cancelar

RN004: O campo descrição não deve ter nenhuma restrição e tem por objetivo apenas identificar de qual tarefa o cadastro se refere.

RN005: O campo data não pode aceitar datas inferiores à data atual.

RN006: A hora deve apenas ser uma hora válida, dentro das 24 horas.

RN007: O campo prioridade deve ser uma lista para seleção exclusiva, não podendo inserir um valor que não esteja previamente cadastrado na lista. As prioridades cadastradas são:

- 1) Máxima
- 2) Prioritária
- 3) Normal
- 4) Não prioritária
- 5) Mínima

6. Visualização de Tarefas

Objetivos:

O objetivo da tela é visualizar e gerenciar as tarefas do usuário, de forma quase análoga à tela de visualização de concursos incluindo até mesmo o envio de tarefas por e-mail. Nesta tela, porém, além de visualizar o usuário deve poder marcar tarefas como feitas e/ou deletá-las em duas operações distintas.

Protótipos:

Imagem 10: tela de visualização de tarefas

Del.	Check Tarefa	Data	Hora	Prioridade
<input checked="" type="checkbox"/>	Ligar para André da Micro Universo	07/01/2017	09:15	1 - Máxima
<input type="checkbox"/>	Tirar xerox do CPF e carteira de identidade	25/06/2017	12:00	1 - Máxima
<input type="checkbox"/>	Tirar xerox da certidão de nascimento	25/06/2017	16:00	2 - Prioritária
<input type="checkbox"/>	Comprar uma lixeira para viagens para carro.	26/06/2017	12:00	4 - Não Prioritária
<input type="checkbox"/>	Começar a estudar Programação Orientada a Objetos	27/06/2017	08:00	2 - Prioritária
<input checked="" type="checkbox"/>	Tirar segunda via do CPF	28/06/2017	11:20	2 - Prioritária
<input type="checkbox"/>	Comprar um estojo novo	30/06/2017	12:00	5 - Mínima
<input type="checkbox"/>	Estudar .Net	03/07/2017	08:00	2 - Prioritária
<input type="checkbox"/>	Comprar apostila de programação em C	10/07/2017	10:00	3 - Normal
<input type="checkbox"/>	Pagar o boleto do concurso	02/08/2017	08:00	2 - Prioritária
<input type="checkbox"/>	Pagar o boleto do concurso	17/08/2017	08:00	4 - Não Prioritária
<input type="checkbox"/>	Ligar para a prefeitura de Ouro Preto caso a sala não tenha sido divulgada	01/12/2017	10:00	1 - Máxima
<input type="checkbox"/>	Conferir homologação do concurso da Micr Universo	01/12/2017	13:00	1 - Máxima
<input type="checkbox"/>	Estudar a Matéria de C#	11/12/2017	11:00	2 - Prioritária
<input checked="" type="checkbox"/>	Tirar xerox da certidão de nascimento	01/01/2018	12:00	3 - Normal
<input type="checkbox"/>	Comprar uma mochila	02/03/2018	11:11	4 - Não Prioritária
<input type="checkbox"/>	Comprar uma lancheira	02/03/2018	10:15	5 - Mínima
<input type="checkbox"/>	Pagar Boleto	03/04/2018	10:00	1 - Máxima
<input type="checkbox"/>	Comprar presente dia das mães	01/05/2018	08:00	2 - Prioritária
<input type="checkbox"/>	Pagar Boleto	04/05/2018	10:00	1 - Máxima
<input type="checkbox"/>	Conferir Homologação Concurso UFRJ	03/06/2018	10:11	2 - Prioritária
<input type="checkbox"/>	Cortar o cabelo	04/06/2018	12:00	3 - Normal
<input type="checkbox"/>	Revisão do carro antes de viajar para o Rio de Janeiro	07/07/2018	09:00	3 - Normal

RI018: a operação de deleção deve ser precedida com uma tela pop-up de diálogo, para confirmar a operação.

RN008: marcar as tarefas como feitas, não as deleta automaticamente.

RI019: a tela deve distinguir as tarefas mais prioritárias das menos prioritárias com uma gradação de cores quentes e frias, sendo mais quentes, mais prioritárias.

RI020: todas as colunas da tela, de forma análoga à visualização de concursos, devem poder ser editadas de forma ascendente e descendente.

RN009: todas as ordenações da tela de visualização de tarefas, tem como segundo critério de ordenação a prioridade da tarefa. Exceto quando a coluna sendo ordenada primariamente, seja a própria coluna de prioridade.

7. Envio de e-mail – Funcionalidade implícita

RN010: conforme solicitado pelas telas de visualização anteriores, a classe de e-mail irá funcionar toda vez que o sistema é aberto. Neste momento a instância do objeto deve verificar se existem concursos dentro do período de alerta para cada um dos concursos vigente. Caso exista um concurso o seguinte e-mail deve ser enviado para o endereço cadastrado do usuário:

- **Assunto:**

Alerta de concurso próximo

- **Corpo:**

Você tem um alerta programado de concurso vigente.

Nome do Concurso: Teste

Data da Prova: dd/mm/aaaa

Observação: Observação Exemplo (cadastrada no campo observação do concurso).

Att.

RN011: as tarefas, no entanto devem ser mandadas no mesmo dia em que elas tem a data de exercício cadastradas. Os e-mails terão o seguinte formato:

- **Assunto:**

Alerta de tarefa a executar

- **Corpo:**

Você tem uma tarefa programada para hoje.

Tarefa: tirar xerox do CPF e carteira de identidade

Hora Marcada: 12:00

Prioridade: Máxima

Att.

RN012: como a funcionalidade de e-mail será desenvolvida em uma classe, esta mesma classe deve ser incluída em um outro projeto, com outro executável. O propósito é tirar a responsabilidade do usuário de ter que entrar no sistema, para que os e-mails sejam enviados.

Desta forma o executável deve ser programado para rodar em batch (background, sem que uma interface seja apresentada para o usuário). Este executável deve ser programado no sistema operacional do usuário

final, para que seja executado, tanto periodicamente quanto ao ligar o computador.

Conclusão

O trabalho foi completamente executado da forma como ele foi proposto e o executável final já se encontra em posse do cliente. Para trabalhos futuros tenho duas considerações a fazer.

Primeiramente irei monitorar o uso do software e implementar possíveis novas funcionalidades caso o cliente enxergue necessidade de forma que o trabalho de monografia não serviu com o único propósito de execução e apresentação. O software é para mim, uma forma de expressão de todos os conhecimentos que acumulei durante todo o curso e tenho a intenção de mantê-lo ativo e funcionando.

Em segundo lugar, caso o software adquira mais clientes, que a princípio irei tentar adquirir oferecendo-os a aplicação de forma gratuita, estudarei a possibilidade de alugar um servidor, por exemplo, na Amazon para suportar a aplicação online e torna-la portátil. Isso também aumentaria muito a capacidade de aderência da aplicação.

Caso o banco de dados esteja em um servidor nas nuvens, a classe de e-mail atingiria sua máxima eficiência, pois não demandaria sequer que o usuário ligasse seu computador. Ele receberia e-mail e teria acesso a eles diretamente de seus smartphones por exemplo.

Por fim gostaria de agradecer todos os envolvidos no desenvolvimento deste trabalho, desde meu orientador ao coordenador dos trabalhos de fim de curso interdisciplinar e os clientes que se dispuseram a interagir comigo durante todo o processo de desenvolvimento do software.