Universidade Federal
do Rio de Janeiro
Escola Politécnica

Sistema de Gerenciamento da Organização D.R.E.A.M.

Luiz Augusto da Silva Alves

Projeto de Graduação apresentado ao Curso de Engenharia de Computação e Informação da Escola Politécnica/Departamento de Engenharia Eletrônica, Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Engenheiro.

Orientador: Aloysio de Castro Pinto Pedroza

Rio de Janeiro

Março de 2013

SISTEMA DE GERENCIAMENTO DA ORGANIZAÇÃO D.R.E.A.M.

Luiz Augusto da Silva Alves

PROJETO DE GRADUAÇÃO SUBMETIDO AO CORPO DOCENTE DO CURSO DE ENGENHARIA DE COMPUTAÇÃO E INFORMAÇÃO DA ESCOLA POLITÉCNICA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE ENGENHEIRO DE COMPUTAÇÃO E INFORMAÇÃO.

Autor: Luiz Augusto da Silva Alves (DEL/POLI/UFRJ)

Orientador: Prof. Aloysio de Castro Pinto Pedroza, Dr. (PEE/COPPE/UFRJ)

Examinador: Prof. Henrique Luiz Cukierman, D. Sc. (PESC/COPPE/UFRJ)

Examinador: Prof. Marcos do Couto Bezerra Cavalcanti, D.Sc. (PEP/COPPE/UFRJ)

RIO DE JANEIRO, RJ – BRASIL

MARÇO DE 2013

Alves, Luiz Augusto da Silva

Estudo de Caso: Sistema de Gerenciamento da Organização D.R.E.A.M. / Luiz Augusto da Silva Alves. – Rio de Janeiro: UFRJ/ Escola Politécnica, 2013.

57 p.: il.; 29,7 cm.

Orientador: Aloysio de Castro Pinto Pedroza

Projeto de Graduação – UFRJ/ Escola Politécnica/ Curso de Engenharia de Computação e Informação, 2013.

Referências Bibliográficas: p. 42-43.

1. Sistemas web 2. Banco de dados 3. Informática e sociedade 4. Gestão do Conhecimento I. PEDROZA, A. C. P. II. Universidade Federal do Rio de Janeiro, Escola Politécnica, Curso de Engenharia de Computação e Informação. III. Título.

“Who dares wins.”

“Quem ousa ganha.”

(Lema de várias unidades militares do mundo)

Dedicatória

À minha mãe Helena, uma pessoa muito carinhosa que me criou e sempre quis o meu bem.

Ao meu pai Luiz, muito trabalhador e que desde cedo despertou meu interesse por tecnologia.

À minha namorada Caroline, pelo carinho e companheirismo desde que nos conhecemos em 2005 e desde que nos tornamos namorados em 2011, sendo incentivadora e paciente, mesmo à distância.

Ao meu irmão Pedro, que daqui a alguns anos começará a vida profissional também em engenharia.

À minha avó Rosa, que é a pessoa mais bondosa que eu conheci na minha vida.

Agradecimentos

A Deus, por ter caminhado de mãos dadas comigo desde sempre.

Ao Professor Aloysio Pedroza do Programa de Engenharia Elétrica, meu orientador neste projeto e que conduziu meus pensamentos para escrever este documento.

Aos Professores Henrique Cukierman do Programa de Engenharia de Sistemas e Computação, que me deu uma oportunidade de trabalhar como bolsista de iniciação científica em um projeto da COPPE; e Professor Marcos Cavalcanti, do Programa de Engenharia de Produção, por ter aceitado participar da banca examinadora deste projeto e por ter me despertado o interesse por esta área de engenharia que alia conhecimento e inovação ao mundo dos negócios.

A todos os outros professores que me marcaram durante mais de 5 anos de UFRJ, seja pela personalidade, profissionalismo, inspiração ou simplesmente pelas disciplinas que estes ensinaram. Lembro aqui alguns como José Ferreira de Rezende, Luís Henrique Kosmalski, Fernando Gil, Alexandre Evsukoff, Cláudio Esperança.

Ao Professor Paul van Vliet do *College of Information Science & Technology* da *University of Nebraska at Omaha (UNO)*, nos Estados Unidos da América, onde estudei durante 1 ano da minha graduação. Dr. Paul me convidou para trabalhar neste projeto da ONG D.R.E.A.M. e me ensinou e incentivou a continuar trabalhando neste que hoje é o meu projeto de graduação.

Ao Steve Warren, fundador e dono da ong D.R.E.A.M. que buscou o pessoal da *UNO* para desenvolver este projeto e acreditou no meu potencial.

Aos professores da UNO Robin Gandhi, que meu deu a oportunidade de cursar uma disciplinas nos EUA junto com a turma de mestrado, sendo esta a disciplina mais difícil e desafiadora que já fiz na faculdade; Douglas Derrick e Kerry Ward que me

ensinaram que inovação e tecnologia da informção são uma das bases atuais dos negócios de quaisquer indústrias; e John Anstey, por ter me ensinado em um semestre de aula tudo aquilo que eu sei sobre educação universitária, negócios, gerenciamento, produtos, markerting, finanças e contabilidade, aprendidos na escola de negócios da UNO.

Agradeço aos amigos da faculdade pela ajúda mútua nos estudos, provas e trabalhos, que durante mais de 5 anos caminharam comigo nesta difícil jornada e pelos momentos divertidos nesse tempo todo: Vítor Sousa, Igor Campbell, Lívia Ribeiro, Mariane Martins, Thaiana Lima, Silvia Benza, Pedro Lemos, Edimar Júnior, Pedro Figueiredo, Thiago Souza, Vanessa Marques e Bernardo Costa.

Agradeço aos meus amigos da Ilha do Governador que há anos participam intensamente da minha vida: Matheus Altomare, Diego Passos, Hugo Bueno, Fábio Mendoza, Larissa de Oliveira, Gabriel Martins, Luana Botelho, Rochele Tambosi, Alessandra Yoko, entre outros do nosso grupo.

Agradeço aos meus amigos do Instituto Guanabara, Bianca Lemos, Rafael Leoni, Laís Manhães e Ricardo Fagundes pela amizade que se mantém mesmo depois da escola.

Agradeço aos meus amigos brasileiros que compartilharam comigo a experiência do intercâmbio acadêmico nos Estados Unidos durante o ano mais intenso e divertido da minha vida até aqui e que criaram uma amizade sem fronteiras: Gabriel Vasconselos, Paulo Franco, Lariana Stefanello, Matheus Amazonas, Bruno Carvalho, Charles Rodrigues, Aurélio Gandra e Marcelo Prade.

Finalmente agradeço a todos que participaram da caminhada até aqui para que eu me tornasse um engenheiro profissional.

Resumo do Projeto de Graduação apresentado à Escola Politécnica da UFRJ, como parte dos requisitos necessários para a obtenção do grau de Engenheiro de Computação e Informação.

Sistema de Gerenciamento da Organização D.R.E.A.M

Luiz Augusto da Silva Alves

Março/2013

Orientador: Aloysio de Castro Pinto Pedroza, D.Sc.

Curso: Engenharia de Computação e Informação

Resumo. Este projeto é um pequeno sistema de gerenciamento e armazenamento de dados desenvolvido para uma pequena organização não-governamental que cuida de crianças e jovens. O sistema permite adicionar, editar e visualizar as informações sobre alunos, escolas, pais, mentores, médicos, medicamentos, programas de tutoria, entre outras entidades relacionadas com o assunto. Além disto, a metodologia de desenvolvimento do software, o modelo conceitual, a implementação, a segurança do sistema e os casos de uso são detalhados ao longo do documento.

Palavras-chave: *Sistema Web, banco de dados, informática e sociedade, gestão do conhecimento*

Abstract of Undergraduate Project presented to POLI (UFRJ) as a partial fulfillment of the requirements for the degree of Computer and Information Engineer.

D.R.E.A.M. Organization Management System

Luiz Augusto da Silva Alves

March/2013

Advisor: Aloysio de Castro Pinto Pedroza, D.Sc.

Major: Major Computer and Information Engineering

Abstract. This project is a small data management and storage system developed for a small non-governmental organization that takes care of children and young man. The system allow add, edit and view information about students, schools, parents, mentors, doctors, medicine, mentoring programs, and other entities related to this subject. Moreover, the software development methodology, the conceptual model, the implementation, the system security and the use case are explained in details through this document.

Keywords: web system, database, informatics and society, knowledge management

Sumário

1. Introdução

A organização não-governamental americana chamada D.R.E.A.M. (*Developing Relationships Through Education and Mentoring* – Desenvolvendo Relacionamentos Através de Educação e Tutoria) fundada na cidade de Omaha, estado do Nebraska, Estados Unidos em 2006, tem como missão *"expandir oportunidades para as famílias e os jovens, fornecendo papéis positivos, experiências de vidas inestimáveis e apoio contínuo e assistência"* [1].

Esta ONG trabalha com programas de tutoria, onde um mentor voluntário fornece assistência financeira, social e educacional para que um dos jovens contemplados se envolva com a comunidade onde vive em atividades após o horário escolar. A ideia é que estes tutores ensinem o valor da educação e como desenvolver o caráter destes jovens para que estes tenham um futuro de sucesso.

Cada mentor é uma espécie de patrocinador que está ligado à um ou mais programas da ONG e estes programas contam com a participação de mais de um aluno beneficiado. As escolas dos alunos também podem participar dos programas fornecendo ambientes de estudos, recursos materiais e humanos em horas alternativas para reforço do ensino, por exemplo. Os pais ou guardiões dos alunos são os responsáveis por cuidar de detalhes como a saúde, médicos, medicamentos e manter a motivação em casa para que estes mantenham o interesse pelo estudo.

Atualmente apenas na cidade de Omaha 150 jovens de três escolas de ensino fundamental e médio são agraciados pelos programas da D.R.E.A.M., além de outros jovens em outros dois estados americanos.

Observação: como o trabalho foi desenvolvido para um cliente americano, alguns termos podem aparecer em inglês, porém traduzidos sempre que possível.

1.1 Objetivo do projeto

Em agosto de 2012, a direção da ONG buscava o desenvolvimento de um sistema web capaz de armazenar, organizar e relacionar os dados da organização em um ambiente online (antes, fichas de papel eram utilizadas para registrar as informações). O sistema precisaria rodar em um servidor web Microsoft IIS 7, fornecer um ambiente seguro de navegação (afinal, informações sobre crianças estariam disponíveis na Internet) e ser acessível de qualquer plataforma fixa de navegação (Google Chrome, Mozilla Firefox, Internet Explorer e Safari).

2. Pré-projeto

2.1 Metodologia de Engenharia de Software

Como este projeto é um software fruto de um problema de engenharia de computação, uma das questões relevantes a serem consideradas é qual metodologia de desenvolvimento de software a ser adotada. Depois de algumas reuniões iniciais entre o cliente, o professor orientador e o estagiário, chegou-se a conclusão de comum acordo que uma boa abordagem seria utilizar a metodologia chamada SCRUM, escolha que será justificada adiante.

Esta se enquadra da categoria de métodos ágeis de desenvolvimento de software, cuja filosofia se baseia em alguns princípios definidos em um documento elaboradora em 2001 por 17 engenheiros de software e disponibilizado gratuitamente na web com o título "*Manifesto para Desenvolvimento Ágil de Software*" [2]:

> *Estamos descobrindo maneiras melhores de desenvolver software, fazendo-o nós mesmos e ajudando outros a fazerem o mesmo. Através deste trabalho, passamos a valorizar:*

> ***Indivíduos*** *e interações mais que processos e ferramentas*
> ***Software*** *em funcionamento mais que documentação abrangente*
> ***Colaboração com o cliente*** *mais que negociação de contratos*
> ***Responder a mudanças*** *mais que seguir um plano*
>
> *Ou seja, mesmo havendo valor nos itens à direita, valorizamos mais os itens à esquerda.*

Por trás do SCRUM, está um mundo globalizado onde é difícil especificar completamente de primeira o escopo de um projeto a ser executado, pois as mudanças tanto de tecnologia quanto das regras de negócio de uma organização são muito rápidas e, nem sempre é possível prever o comportamento dos usuários, da tecnologia, do mercado e da própria empresa futuramente. O SCRUM foi criado para que projetos de software pudessem ser entregues em versões parciais, com apenas parte das funcionalidades em operação, sendo que estas são de maior importância para o negócio do cliente, e portanto são melhores definifidas e entendidas antes da implementação efetiva.

Estas funcionalidades, requisitos, correções ou melhorias a serem implementadas são chamadas de **histórias do usuário**, e descrevem sucintamente o que fazer e como esta deve funcionar. Outras funcionalidades que sejam menos importantes entram na base da fila de prioridades e são desenvolvidas após implementação daquelas no topo da fila. O conjunto de histórias do usuário forma o **product backlog** (Figura 1), que consiste basicamente em uma lista destas em ordem de prioridade de acordo com o interesse do dono do produto.

Os três atores que interagem em um projeto com SCRUM são: o dono do produto (***product owner***) que é o cliente do projeto ou pelo menos representa o cliente do projeto e suas expectativas; o ***scrum master*** que é o responsável por facilitar ao máximo o desenvolvimento do sistema ao remover os impedimentos que

bloqueiam ou atrapalham a equipe desenvolvedora que objetiva entregar o que o dono do produto espera para aquele *sprint*; e finalmente a **equipe de desenvolvimento**, que efetivamente implementa o que o dono do produto deseja e especifica no *product backlog*. A equipe de desenvolvimento deve ser auto-organizável para que possa definir e executar as tarefas sozinhas, já que ao contrário do que possa se pensar, o *scrum master* não é uma espécie de gerente de projeto ou líder da equipe.

Algumas das características importantes (e vantajosas) do SCRUM são: rapidez, incrementos periódicos e interatividade. A primeira característica está atrelada ao fato de que, ao contrário de um processo tradicional de desenvolvimento em cascata (sequencial e sem interrupções para mudança de escopo, testes, verificações e validações), o método ágil "quebra" o projeto em processos menores executados em ciclos chamados ***sprints*** (Figura 1). Estes podem variar de 1 semana a 1 mês em geral e definem o espaço de tempo entre a liberação (*release)* de uma versão com as especificações desta sprint e a próxima a ser executada. A segunda característica citada que influenciou na escolha do SCRUM é o fato de que ao final de cada *sprint* o cliente terá um sistema com uma ou mais funcionalidades a mais em relação à versão entregue anteriormente. Finalmente, o cliente participa ativamente do desenvolvimento do produto em todas as suas fases, e não apenas por exemplo no levantamento de requisitos e na validação final. Isto significa que o cliente simplesmente não envia uma lista de requisitos e alguns meses depois espera o resultado; a participação em cada ciclo (*sprint*) é feita nas reuniões de revisão que acontecem ao final de cada sprint, sendo papel deste aprovar ou não as histórias implementadas.

Sobre a "*entrega acelerada de serviços ao cliente*" e "*engajamento do usuário com o sistema*", SOMMERVILE [3] escreve em seu livro sobre Engenharia de Software que "*os incrementos iniciais do sistema podem fornecer uma funcionalidadede de alta prioridade, de forma que os clientes logo poderão obter valor do sistema durante seu desenvolvimento. [...] Os usuários do sistema precisam estar envolvidos no processo*

de desenvolvimento incremental porque eles devem dar feedback à equipe de desenvolvimento sobre os incrementos entregues. Esse envolvimento não significa somente que o sistema terá mais chances de atender aos requisitos, mas também que os usuários finais que estão comprometidos com o sistema querem vê-lo funcionando."

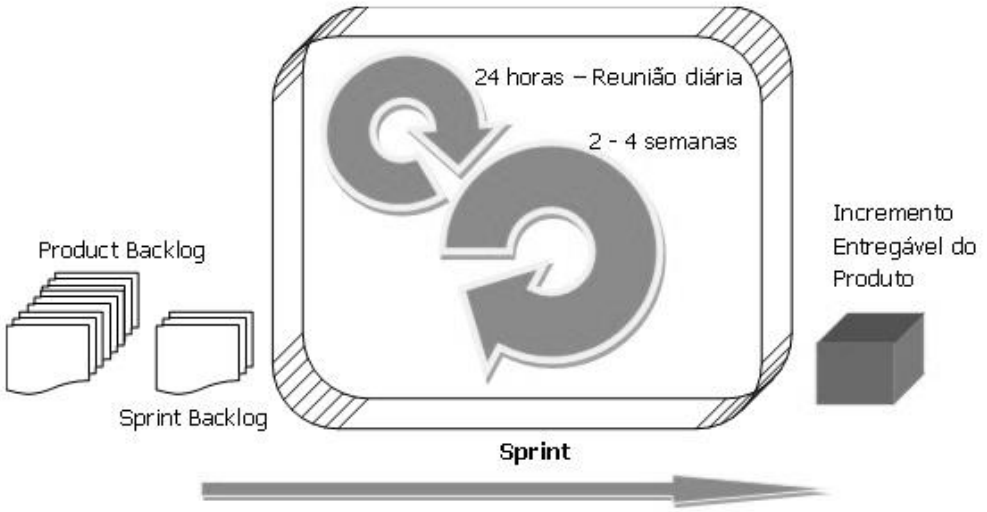


Figura 1. Esquema de funcionamento do SCRUM. Adaptado de [4]. O *sprint backlog* é um subconjunto do *product backlog* que contém as histórias de usuário a serem implementadas em determinado *sprint*.

Os requisitos do projeto foram definidos inicialmente em reunião com o diretor da ONG Steve Warren e sua assistente para que as regras da entidade fossem compreendidas e prioridades iniciais definidas. Ainda sobre a engenharia de requisitos, Sommervile cita uma frase importante: "*requisitos vagos e com baixa prioridade serão implementados quando e se os usuários solicitarem*". Com base nesta análise e após algumas reuniões extras, foi estabelecido que o SCRUM seria a melhor metologia de desenvolvimento e foi aceita pelas partes devido à rapidez de entrega e melhores práticas para o caso de mudanças inesperadas de escopo. Ficou

também definido que os *sprints* teriam duração de 2 semanas, sendo agendadas reuniões de validação a cada fim de ciclo.

2.2 Levantamento de Requisitos

Os requisitos elicitados tiveram como base: as regras de negócio da ONG explicadas pelo diretor, a documentação administrativa e as fichas utilizadas para o registro em papel dos dados de todos os componentes da ONG (mentores, estudantes, pais, médicos, escolas, etc.).

O Sistema de Gerenciamento D.R.E.A.M. tinha como requisitos principais operações de inserção, vizualição e edição (atualização) das entidades relativas à ONG, mapeadas e relacionadas em um modelo entidade-relacionamento descrito a seguir. A operação remoção, uma das quatro utilizadas em sistemas CRUD (*Create, Remove, Update, Delete*) foi explicitamente solicitada pelo cliente para que não fosse incluída no sistema. Talvez pelo fato de ser a mais sensível das operações, deletar um registro da base de dados por acidente poderia ser crítico para a D.R.E.A.M..

Outro requisito foi que a URL do sistema não fosse de conhecimento público, uma vez que informações sobre menores de idade estariam disponibilizadas na web. Além disto, as requisições ao servidor deveriam ser feitas utilizando-se o protocolo HTTPS, que é a versão segura do já conhecido protocolo HTTP sobre uma camada de segurança SSL para que os usuários pudessem obter conexões seguras. Sobre isto, TANEMBAUM [5] afirma que *“depois que a conexão segura é estabelecida, a principal tarefa da SSL é manipular a compactação e a criptografia”.*

A quantidade de usuários foi definida em 10, diretor, secretária e assistentes administrativos com pleno acesso aos dados no sistema. Este requisito dispensou a separação em níveis de privilégios comuns em outros sistemas web, ou seja, todos poderiam utilizar o Sistema de Gerenciamento D.R.E.A.M. de forma igual.

Um requisito exigido pelo cliente foi a geração de quatro relatórios que gerassem as seguintes listas: mentores de acordo com a etnia, mentores de acordo com o gênero, mentorados de acordo com a etnia e mentorados de acordo com o gênero.

2.3 Modelo conceitual do sistema

O modelo descrito a seguir surgiu como um esboço muito básico definido entre o cliente e o desenvolvedor para que de início pudessem ter uma ideia de como deveria ser o sistema ao final do projeto. Ao longo da implementação, liberações de versões e inclusão e/ou exclusão de novas funcionalidades, o projeto foi entregue conforme o diagrama da Figura 2.

Como o sistema foi projetado para ser a solução de um problema de armazenamento e relacionamento de dados através de uma interface web “amigável” entre o usuário e o banco de dados que faz este trabalho, foi denifinido que uma estrutura de menus e submenus em árvore de fácil compreensão seria um modelo candidato a ser adotado. Por enquanto Os relacionamentos e dependências entre as entidades/classes do modelo serão descritas na próxima seção.

O controle de acesso é feito na raiz da árvore de diretórios (*Index*) através da tela de login: caso o usuário tenha acesso ao sistema, este é autorizado à explorar a área restritra; caso contrário, o acesso é negado. Para o primeiro caso, o sistema redireciona o usuário para a página inicial (*Home*) de onde é possível acessar quatro menus (*Mentor*, *Aluno*, *Miscelânea* e *Relatórios*) e uma página (*Novo Usuário*).

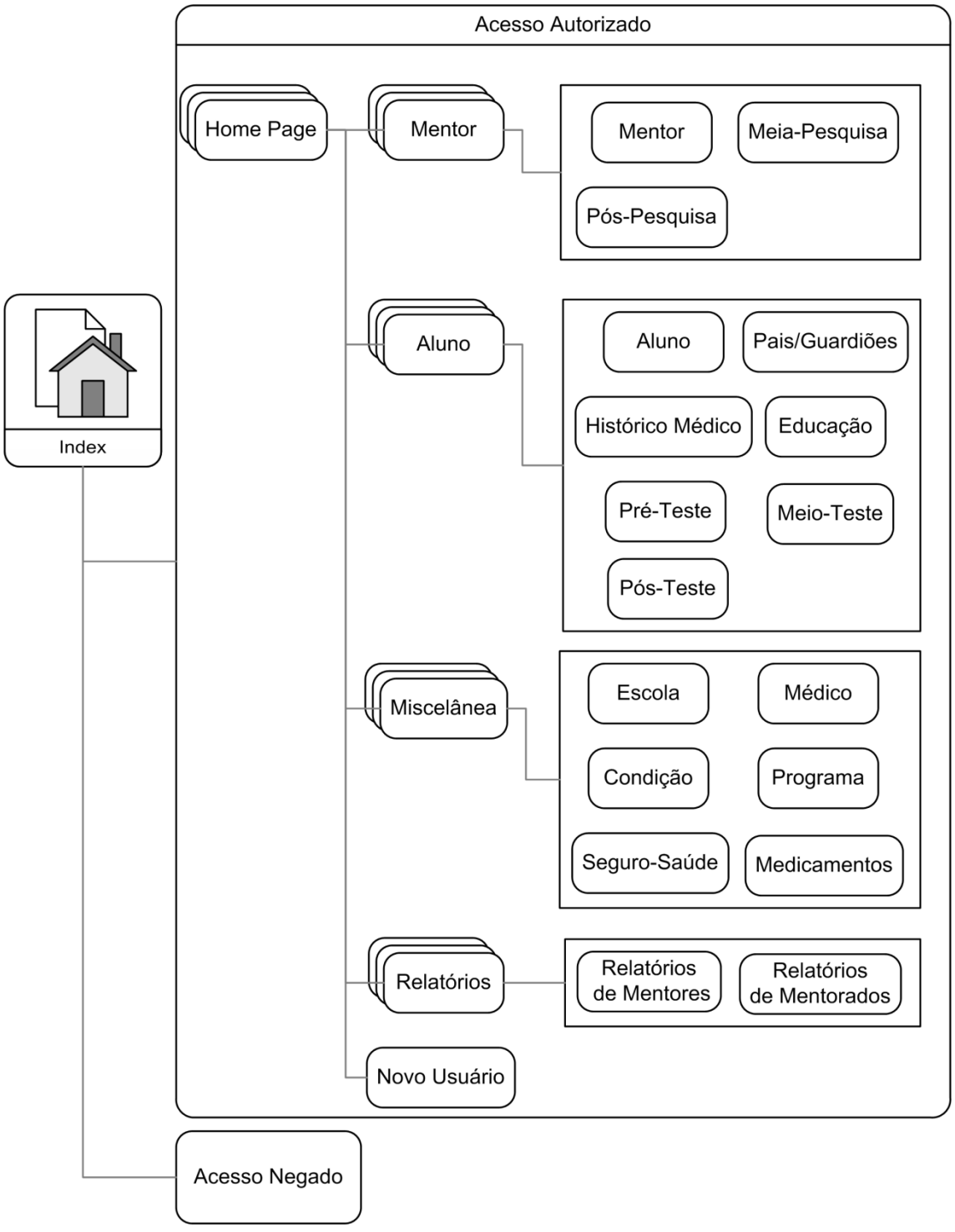


Figura 2. Modelo conceitual do sistema proposto.

No menu *Mentor*, o usuário pode cadastrar, visualizar ou adicionar novos mentores que ingressam na ONG e que serão responsáveis pelo acompanhamento dos alunos durante o período em que este relacionamento estiver estabelecido em

Mentor. Ainda neste menu, podem ser incluídas e visualizadas as pesquisas de avaliação que ocorrem durante e após o programa de mentoramento: *Meia-Pesquisa* e a *Pós-Pesquisa*. Estas pesquisas são feitas para que a administração da ONG acompanhe e avalie a participação destes mentores através de perguntas sobre as expectativas, resultados e o relacionamento com seus mentorados.

O menu *Aluno* é o maior pois abrange sete submenus, sendo estes *Aluno*, *Pais/Guardiões*, *Histórico Médico*, *Educação*, *Pré-Teste*, *Meio-Teste* e *Pós-Teste*. Sobre os últimos três, a primeira pesquisa é um questionário para testar se um aluno tem o perfil que habilite-o a participar de um dos programas da ONG, enquanto as outras duas são de auto-avaliação para que os administradores possam acompanhar a evolução de um aluno durante um programa. O submenu *Educação* contém informações como a escola, o ano e o coeficiente de rendimento acumulado do aluno. Já o submeno *Histórico Médico* informa sobre condições medicas especiais, remédios que o aluno toma e as respectivas dosagens.

O menu *Miscelânea* permite aquelas operações anteriores (inserir, visualizar e editar) em entidades tais como *Escola*, *Médico*, *Condição*, *Programa*, *Seguro-Saúde* e *Medicamentos*. Os nomes são bem intuitivos e dão uma boa ideia do que representam. Cabe aqui talvez esclarecimentos que *Condição* refere-se de uma maneira geral à condições física ou psicológicas dos alunos cadastrados e que *Programa* consiste nos programas de tutoria oferecidos pela ONG nos quais participam alunos e mentores.

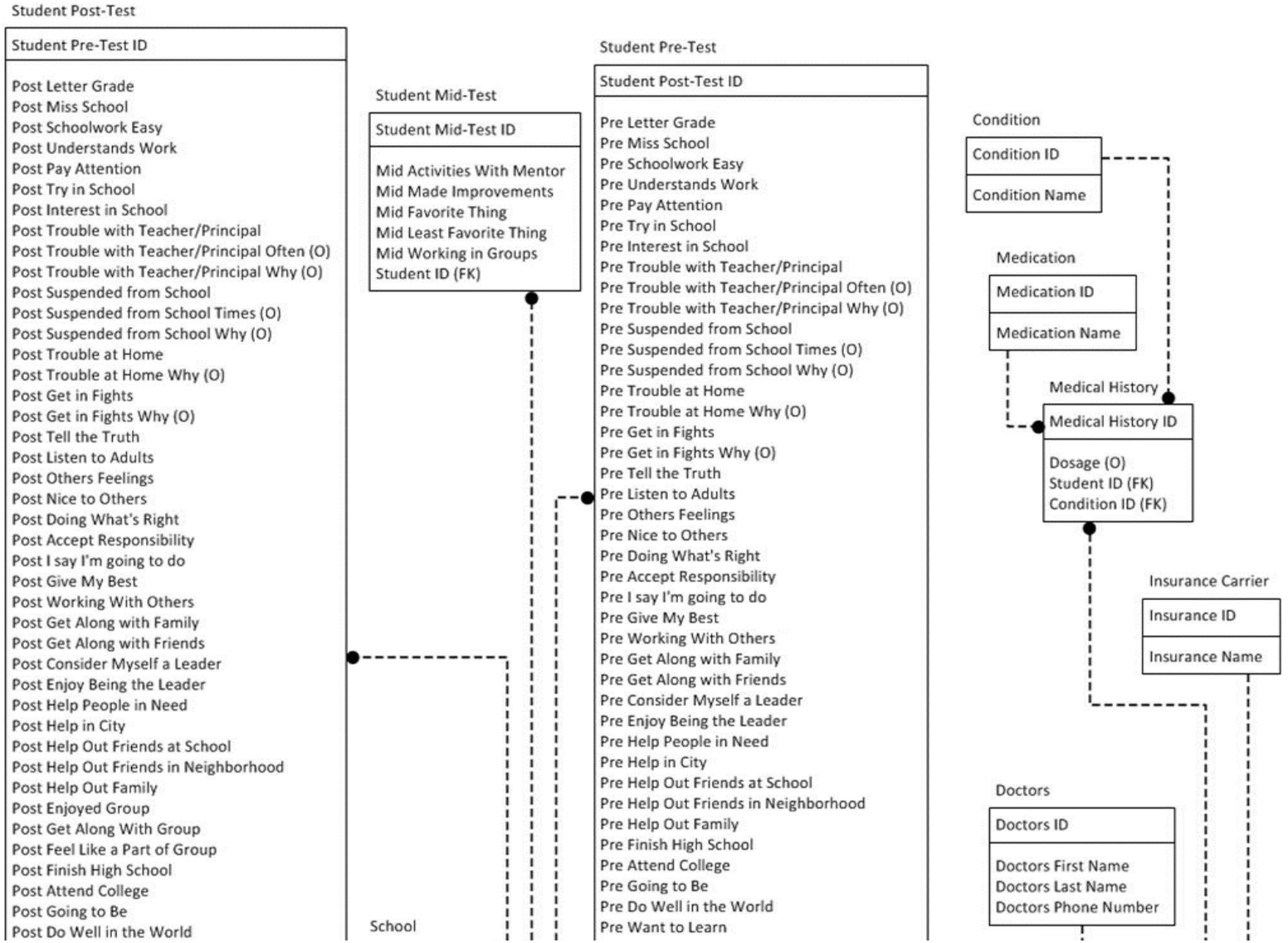
Finalmente, o submenu *Relatórios* gera os quatro relatórios citados ao final da seção anterior. Na interface gráfica, estes relatórios deveriam se apresentar de maneira tal que a formatação fosse organizada e permitisse a impressão em folha de papel. A página *Novo Usuário* será usada pelos administradores da ONG para criar novos usuários conforme suas necessidades.

2.4 Modelo entidade-relacionamento

O modelo de dados entidade-relacionamento é utilizado em um nível alto de abstração para estabelecer os relacionamentos entre entidades e listar seus atributos. Sobre isto, NAVATHE [6] escreve que *"o objeto básico que o modelo ER representa é uma entidade, 'algo' do mundo real, com uma existência independente"* e que *"cada entidade tem atributos – propriedades particulares que a descrevem"*.

A Figura 3 representa o modelo ER do Sistema de Gerenciamento da D.R.E.A.M., tendo como entidade central o *Aluno*. Observe que a esta tabela está associado o maior número de relacionamentos, demonstrando que a modelagem conceitual das regras de negócios da organização realmente condizem com a realidade, já que a ONG tem mesmo como foco o aluno. Um aluno pode se relacionar muitas com as pesquisas, sendo que estas são entidades fracas, significando que sua existência depende da existência de um aluno. O mesmo acontece com as pesquisas da entidade que representa os mentores.

Um médico e um seguro-saúde podem se relacionar com muitos alunos, embora o contrário seja de um aluno para um médico e um seguro-saúde. Observa-se também que um aluno pode ter mais de um guardião, em geral um pai e uma mãe e, um guardião pode manter mais de um aluno, no caso de os alunos serem irmãos. Estas duas entidades estão relacionadas através de um atributo chamado *Relacionamento* em *Student-To-Parent/Guardian* (pai e filho, avó e neto, tio e sobrinho, por exemplo).


Student Post-Test
Student Pre-Test ID
Post Letter Grade
Post Miss School
Post Schoolwork Easy
Post Understands Work
Post Pay Attention
Post Try in School
Post Interest in School
Post Trouble with Teacher/Principal
Post Trouble with Teacher/Principal Often (O)
Post Trouble with Teacher/Principal Why (O)
Post Suspended from School
Post Suspended from School Times (O)
Post Suspended from School Why (O)
Post Trouble at Home
Post Trouble at Home Why (O)
Post Get in Fights
Post Get in Fights Why (O)
Post Tell the Truth
Post Listen to Adults
Post Others Feelings
Post Nice to Others
Post Doing What's Right
Post Accept Responsibility
Post I say I'm going to do
Post Give My Best
Post Working With Others
Post Get Along with Family
Post Get Along with Friends
Post Consider Myself a Leader
Post Enjoy Being the Leader
Post Help People in Need
Post Help in City
Post Help Out Friends at School
Post Help Out Friends in Neighborhood
Post Help Out Family
Post Enjoyed Group
Post Get Along With Group
Post Feel Like a Part of Group
Post Finish High School
Post Attend College
Post Going to Be
Post Do Well in the World
Student Mid-Test
Student Mid-Test ID
Mid Activities With Mentor
Mid Made Improvements
Mid Favorite Thing
Mid Least Favorite Thing
Mid Working in Groups
Student ID (FK)
Student Pre-Test
Student Post-Test ID
Pre Letter Grade
Pre Miss School
Pre Schoolwork Easy
Pre Understands Work
Pre Pay Attention
Pre Try in School
Pre Interest in School
Pre Trouble with Teacher/Principal
Pre Trouble with Teacher/Principal Often (O)
Pre Trouble with Teacher/Principal Why (O)
Pre Suspended from School
Pre Suspended from School Times (O)
Pre Suspended from School Why (O)
Pre Trouble at Home
Pre Trouble at Home Why (O)
Pre Get in Fights
Pre Get in Fights Why (O)
Pre Tell the Truth
Pre Listen to Adults
Pre Others Feelings
Pre Nice to Others
Pre Doing What's Right
Pre Accept Responsibility
Pre I say I'm going to do
Pre Give My Best
Pre Working With Others
Pre Get Along with Family
Pre Get Along with Friends
Pre Consider Myself a Leader
Pre Enjoy Being the Leader
Pre Help People in Need
Pre Help in City
Pre Help Out Friends at School
Pre Help Out Friends in Neighborhood
Pre Help Out Family
Pre Finish High School
Pre Attend College
Pre Going to Be
Pre Do Well in the World
Pre Want to Learn
Condition
Condition ID
Condition Name
Medication
Medication ID
Medication Name
Medical History
Medical History ID
Dosage (O)
Student ID (FK)
Condition ID (FK)
Insurance Carrier
Insurance ID
Insurance Name
Doctors
Doctors ID
Doctors First Name
Doctors Last Name
Doctors Phone Number
School

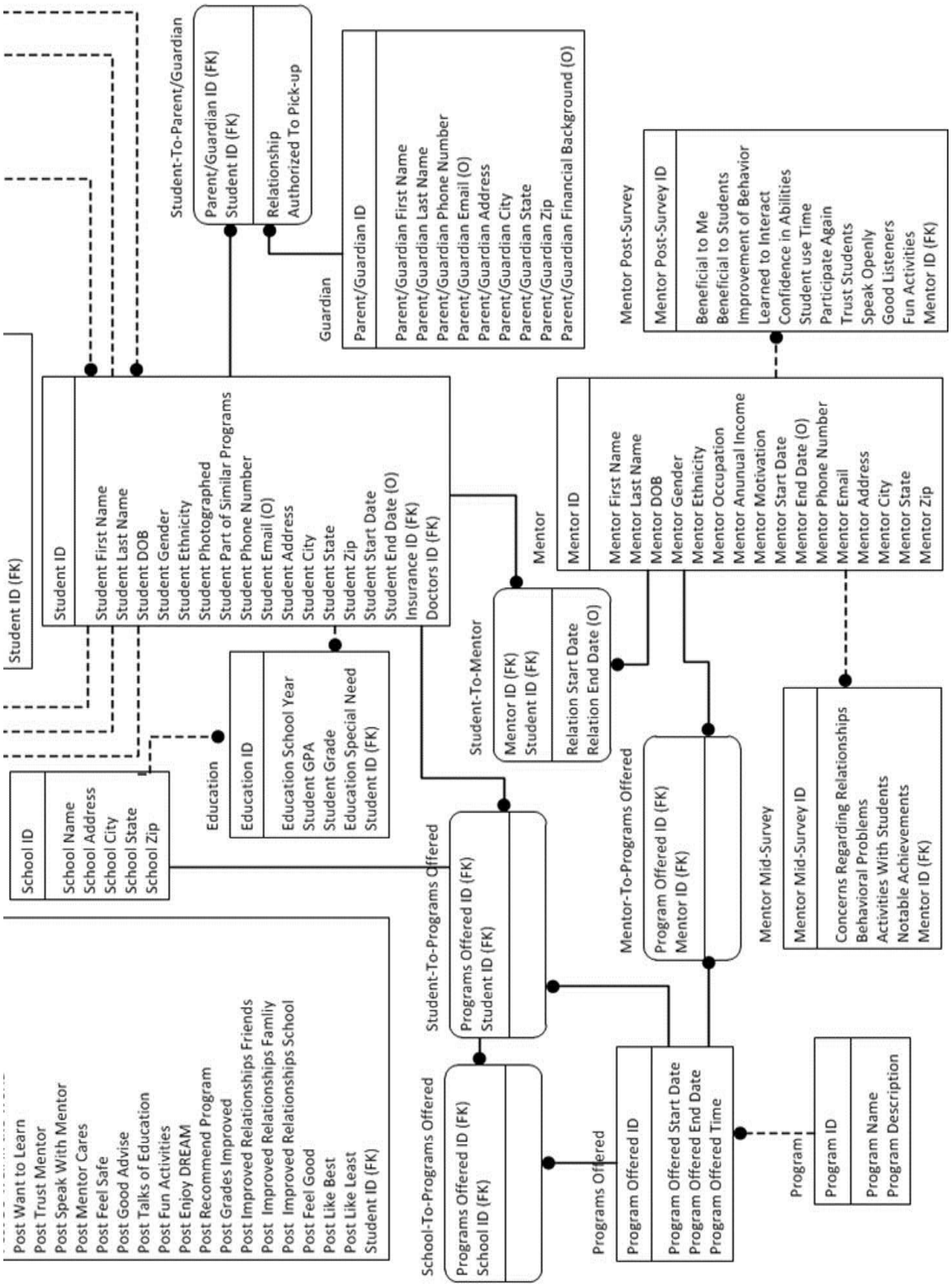


Figura 3. Modelo Entidade-Relacionamento do sistema.

Algo similar acontece com os mentores, que podem mentorar mais de um aluno e estes podem ser mentorados por mais de um mentor, relacionados através do atributo *Data de Início do Relacionamento* e *Data de Término do Relacionamento*. Ainda sobre alunos, cada aluno pode ter relação com mais de uma escola, ao mesmo tempo que uma escola educa vários alunos, sendo a entidade que intermedia esta ligação é a *Educação*.

A entidate *Program* representa um dos programas que a ONG tem e estes se relacionam de um para muitos com *Programs Offered* de modo que este seja uma abstração de uma das edições de um dos programas, definidas por uma data de início e outra de fim. Uma analogia pode ser feita entre uma disciplina e uma turma: a discplina Cálculo 1, por exemplo, pode ter edições em Cálculo 1 2012/1, Cálculo 1 2012/2, Cálculo 1 2013/1, etc. Estes programas são oferecidos também para escolas, alunos e mentores.

Abaixo segue a definição de cada atributo das principais entidades:

Tabela 1. Entidade Aluno (*Student*) e seus atributos.

Student	
Representa um aluno da ONG e contém seus dados pessoais.	
Atributo	**Descrição**
StudentID	Identificador único de um aluno
StudentFirstName	Primeiro nome do aluno
StudentLastName	Último nome do aluno
StudentDOB	Data de nascimento do aluno
StudentGender	Gênero do aluno
StudentPhotographed	Pergunta se está autorizado a ser fotografado
StudentPartOfSimilarProgram	Pergunta se já fez parte de algum programa parecido
StudentPhoneNumber	Telefone do aluno
StudentEmail	Email do aluno
StudentAddress	Endereço do aluno
StudentEthnicity	Etnia do aluno
StudentCity	Cidade do aluno
StudentState	Estado do aluno
StudentZip	Código de endereçamento postal do

	aluno
StudentStartDate	Data de início do relacionamento do aluno com a ONG
StudentEndDate	Data de término do relacionamento do aluno com a ONG
InsuranceID	Identificador único de uma companhia de seguro-saúde
DoctorID	Identificador único de um médico

Tabela 2. Entidade Pai/Guardião (*Parent/Guardian*) e seus atributos.

Parent/Guardian	
Representa um pai ou responsável que mantém a guarda de um aluno.	
Atributo	**Descrição**
Parent/GuardianID	Identificador único de um pai
Parent/GuardianFirstName	Primeiro nome do pai
Parent/GuardianLastName	Último nome do pai
Parent/GuardianPhoneNumber	Telefone do pai
Parent/GuardianEmail	Email do pai
Parent/GuardianAddress	Endereço do pai
Parent/GuardianCity	Cidade do pai
Parent/GuardianState	Estado do pai
Parent/GuardianZip	Código de endereçamento postal do pai
Parent/GuardianFinancialBackground	Descrição breve da situação financeira do responsável

Tabela 3. Entidade Mentor (*Mentor*) e seus atributos.

Mentor	
Representa um mentor que auxilia um aluno.	
Atributo	**Descrição**
MentorID	Identificador único de um mentor
MentorFirstName	Primeiro nome do mentor
MentorLastName	Último nome do mentor
MentorDOB	Data de nascimento do mentor
MentorGender	Gênero do mentor
MentorPhoneNumber	Telefone do mentor
MentorEmail	Email do mentor
MentorAddress	Endereço do mentor
MentorEthnicity	Etnia do mentor
MentorCity	Cidade do mentor
StudentState	Estado do mentor
MentorZip	Código de endereçamento postal do mentor

MentorStartDate	Data de início do relacionamento do mentor com a ONG
MentorEndDate	Data de término do relacionamento do mentor com a ONG
MentorOccupation	Profissão do mentor
MentorAnnualncome	Renda anual do mentor
MentorMotivation	Motivação do mentor para participar de programas da ONG

Tabela 4. Entidade Escola (*School*) e seus atributos.

School	
Representa uma escola.	
Atributo	**Descrição**
SchoolID	Identificador único de uma escola
SchoolName	Nome da escola
SchoolAddress	Endereço da escola
SchoolCity	Cidade da escola
SchoolState	Estado da escola
SchoolZip	Código de endereçamento postal do mentor

Tabela 5. Entidade Educação (*Education*) e seus atributos.

Education	
Entidade associativa que relaciona uma escola com um aluno.	
Atributo	**Descrição**
EducationID	Identificador único de uma “educação”
EducationSchoolYear	Ano da “educação” na vida escolar de um aluno
StudentGPA	Média ponderada que mede o desempenho acumulado de um aluno ao final de um ano. Semelhante ao Coeficiente de Rendimento
StudentGrade	Média de um aluno ao final de um ano.
EducationSpecialNeed	Descreve alguma necessidade especial de um aluno, caso haja
StudentID	Identificador único de um aluno

Tabela 6. Entidade Programa (*Program*) e seus atributos.

Program	
Representa um dos programas da ONG	
Atributo	**Descrição**
ProgramID	Identificador único de um programa
ProgramName	Nome do programa
ProgramDescription	Descrição do programa da ONG

Tabela 7. Entidade Programa Ofercido (*Program Offered*) e seus atributos.

Program Offered	
Representa uma edição de um programa oferecido em um determinado período	
Atributo	**Descrição**
ProgramOfferedID	Identificador único de uma edição de um programa oferecido
ProgramOfferedStartDate	Nome do programa
ProgramOfferedEndDate	Descrição do programa da ONG
ProgramOfferedTime	Descrição do programa da ONG

3.Implementação

3.1 Arquitetura do sistema web

Para a implementação deste projeto foi adotada a chamada Arquitetura Cliente/Servidor de Três Camadas para Aplicações Web, onde há um intermediário entre o cliente que faz requisições e o servidor de banco de dados que responde atendendo o que for solicitado pelo cliente, que pode ser um servidor web ou servidor de aplicações, dependendo do caso. Segundo NAVATHE [6], esta camada intermediária traz algumas vantagens, como por exemplo a segurança na autenticação e autorização de usuários ao entrar no sistema:

"Esse servidor desempenha um papel intermediário armazenando as regras de negócio (procedimentos ou restrições) que são usadas para acessar os dados do servidor de banco de dados. Também pode incrementar a segurança do banco de dados checando as credenciais do cliente antes de enviar uma solicitação ao servidor

de banco de dados. [...] O servidor intermediário aceita as solicitações do cliente, processa-as e envia comandos de banco de dados ao servidor de banco de dados, e então atua como um conduíte por passar (parcialmente) os dados para a apresentação aos usuários em um formato GUI (Graphic User Interface).”

3.2 A tecnologia ASP.NET

A tecnologia Microsoft ASP.NET para páginas web dinâmicas e a linguagem C# [7] foram escolhidas para a implementação da segunda camada da arquitetura explicada acima, ou seja, executando comandos de acordo com as entradas obtidas na camada acima e processando os dados e executando transações no banco de dados na camada abaixo, sendo este o Microsoft SQL Server 2012. Esta tecnologia atua lado do servidor web, neste caso o escolhido foi o Microsoft IIS7 (Internet Information Services 7), uma vez que este já hospedava o website oficial do cliente..

Figura 4. Arquitetura Lógica cliente/servidor de três camadas com as respectivas tecnologias adotadas. Adaptado de NAVATHE [6].

O framework ASP.NET permite que um sistema de gerenciamento como o proposto fosse implementado de forma fácil, rápida e confiável, seguindo a arquitetura descrita acima, através do padrão de software MVC (Model-View-Controller). Este padrão utiliza um dos conceitos mais importantes em engenharia de software que é a separação de conceitos. Sobre isto, LAPLANTE [8] escreve que separação de conceitos é encontrada através da modularização do código e do desenho de software orientado a objeto; em [9] a sugestão é separar o projeto em 3 camadas de abstração: apresentação, negócios e dados.

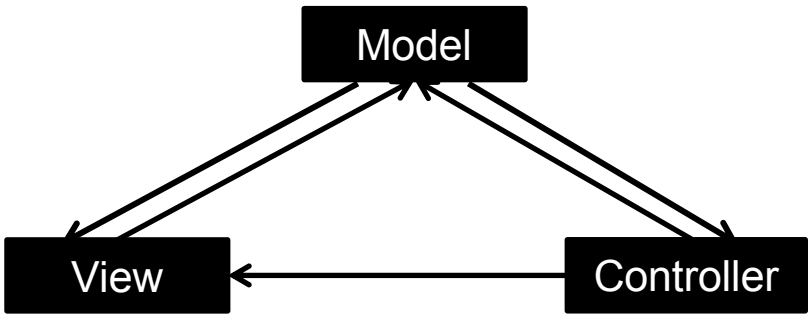


Figura 5. Diagrama representando a comunicação no padrão de software MVC.

Cada uma destas 3 camadas lógicas tem um papel diferente na implementação de uma aplicação web:

- O modelo (*model*) é basicamente a camada de domínio de dados da aplicação, implementando fisicamente o modelo conceitual de dados da ONG descrito na Figura 3. Outra função desta camada é a recuperação e armazenamento de dados no banco.
- A visão (*view*) é a parte que trata da apresentação adequada para o usuário através do conteúdo (HTML) e do layout (CSS) . A interface gráfica foi desenvolvida com base em formulários para preenchimento e envio para processamento pelo controlador.
- O controlador, como sugere o nome manipula a interação com o usuário através da resposta a eventos (geralmente ações de usuários), controlando

dados de entrada da visão, processando-os e enviando ao modelo. Neste projeto o controlador contém a lógica necessária para executar validações nas entradas dos formulários e as operações de inserção, seleção e atualização no banco de dados. De acordo com [10], controllers são usados para enviar mensagens para o modelo e fornecer a interface entre o modelos e suas visões associadas.

O esquema abaixo exemplifica o comportamento do fluxo de dados em uma execução do sistema proposto, onde cada bloco é um arquivo implementado em um diretório específico de acordo com o modelo de um projeto ASP.NET. Observe que todos tem a extensão .cshtml, indicando que foi utilizado a linguagem C# (C Sharp).

O modelo de dados implementado nas tabelas de um banco no SQL Server estão definidos no arquivo StudentModel.cshtm, onde por exemplo define-se que “StudentID” é do tipo inteiro, não pode ter valor nulo pois é a chave primária da entidade e seu valor também é incrementado automaticamente quando um registro novo é adicionado na base de dados; já “StudentFirstName” é do tipo *string* e não pode receber valores nulos, pois é o nome de um aluno e não faz sentido um aluno que tenha outras informações, porém sem nome; outro exemplo de definição é o atributo “StudentDOB” que define a data de nascimento de um aluno e tem o tipo *datetime* no formato MM/DD/AAAA, padrão americano para datas. Esta é uma das vantagens do padrão MVC e da orientação a objetos: as entidades são implementadas diretamente na linguagem C#, que cria scripts SQL, que por sua vez criam as tabelas e os relacionamentos no SQL Server.

A página web com menus, formulários, links e botões que, por exemplo, adiciona um novo aluno ao banco de dados, é implementada no arquivo StudentAdd.cshtml, sendo este a *view* da entidade e a interface gráfica direta de interação com o usuário. Ao receber entradas nos formulários, esta visão as envia

para o controlador através de métodos POST, onde são processados em StudentAddController.cshmtl. Este arquivo contém o código responsável por tratar as entradas da visão e as repassar para o modelo. Finalmente, quando houver necessidade de uma visão recuperar dados armazenados no banco através de metodos GET, esta pode acessar diretamente um modelo, que disponibilizará as informações na mesma.

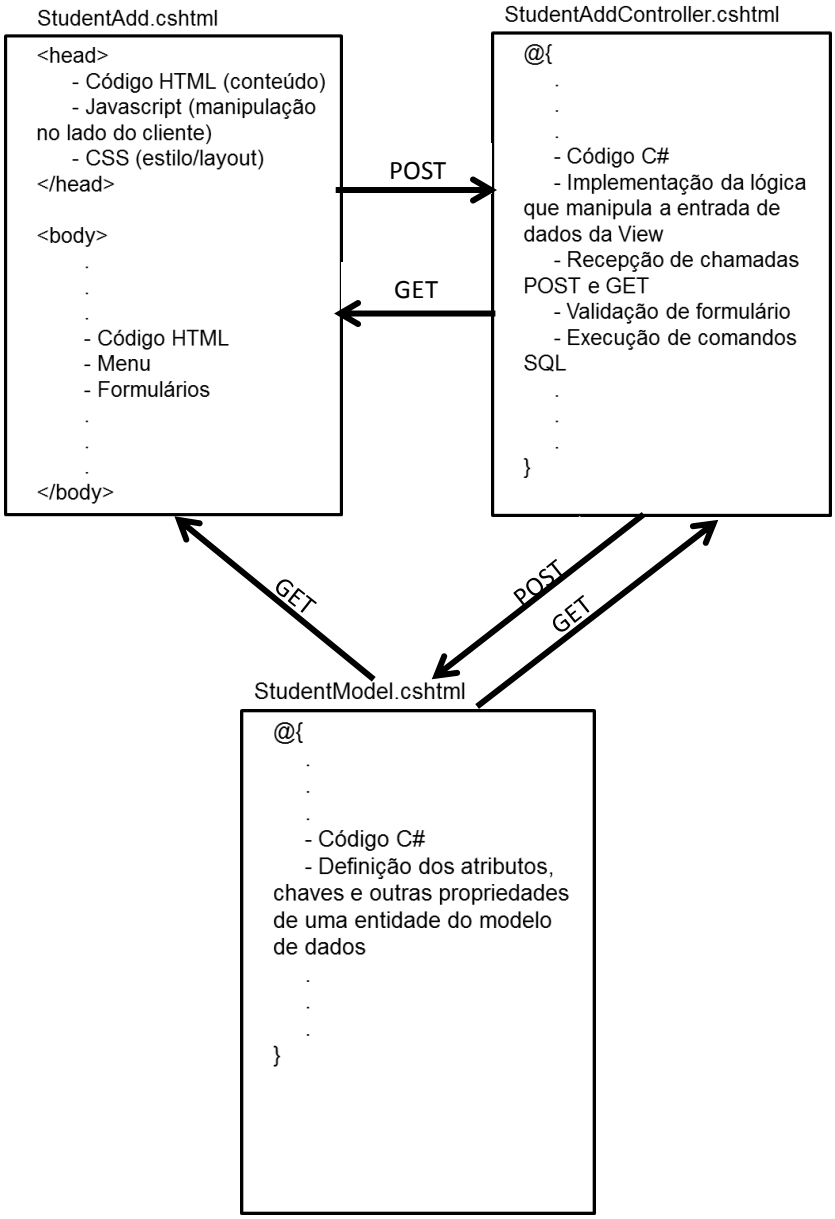


Figura 6. Esquema de troca de mensagens (dados e comandos) entre os arquivos do sistema.

3.3 Interface gráfica

A interface gráfica é definitivamente o elo entre os dados armazenados no sistema de gerenciamento e o usuário que opera o computador, fazendo desta uma camada fundamental para o correto e fácil uso de um sistema informático, além da agregação de valor para o cliente (neste caso a ONG não é um negócio objetiva o lucro, mas precisa do apoio desta ferramenta para gerenciar informações importantes para a administração da organização).

A primeira tela assim que a URL é acessada pela primeira vez é o controle de acesso feito através da dupla “usuário” e “senha”. Em seguida o painel inicial da tela é exibido com o menu principal na lateral esquerda, em três níveis onde é possível acessar o cadastro e a visualização de novas informações. O sistema é bem intuitivo de usar, sendo que mensagens de êxito ou erro são lançadas a cada evento executado confirmando ou não aquela ação.

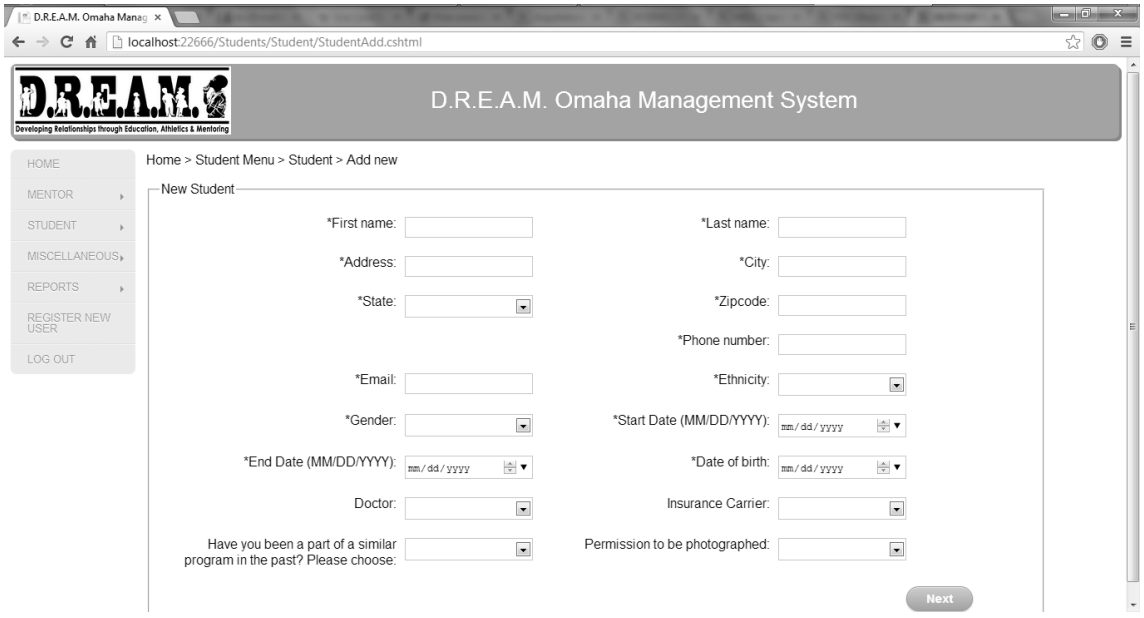


Figura 7. Exemplo da interface gráfica da tela de adicionar um novo aluno, acessada no Google Chrome.

Além de menus de formulário com as já citadas operações de adicionar, visualizar e editar, um menu com relatórios com criado, permitindo a geração de relatórios referentes a alunos e a mentores, de acordo com etnia e gênero. O cabeçalho de cada coluna permite ordenar a lista gerada de acordo com algum critério (ordem alfabética de nome, ordem cronológica de data, por exemplo). De acordo com os requisitos, estes relatórios deveriam ser apresentados de forma que possibilitassem a impressão em papel.

Figura 8. Imagem do relatório de alunos por etnia gerado pelo sistema.

3.4 Cuidados com segurança

Devido ao fato de que o projeto tem como propósito final armazenar informações sobre uma ONG que cuida de crianças, esses dados são considerados extremamente sensíveis por motivos de segurança. Ninguém gostaria de um vazamento público de informações sobre jovens e sua vida na web, e por isto o cuidado deste projeto com a segurança precisou de atenção especial.

O primeiro fato a considerar é que ao implantar o sistema no servidor do cliente, este deveria possuir uma URL diferente da URL utilizada no site oficial da entidade: isto é uma forma de desvincular o site institucional do sistema web de gerenciamento de informações. A URL também utiliza a versão segura do protocolo HTTP, conhecida como HTTPS, onde as informações de login e outras entradas dos formulários são criptografadas na conexão, antes de serem transmitidas ao servidor.

Outro ponto importante a considerar é o armazenamento de senhas dos usuários no banco de dados, que não pode ser feito de forma explícita, pois se o banco estiver vulnerável, as senhas estarão disponíveis para acesso ao sistema. A solução foi criptografar estas senhas também no lado do servidor com um algoritmo MD5. Observe na imagem abaixo a senha armazenada para o login "luiz.aug.alves@gmail.com", correspondente ao valor "ufrj2014":

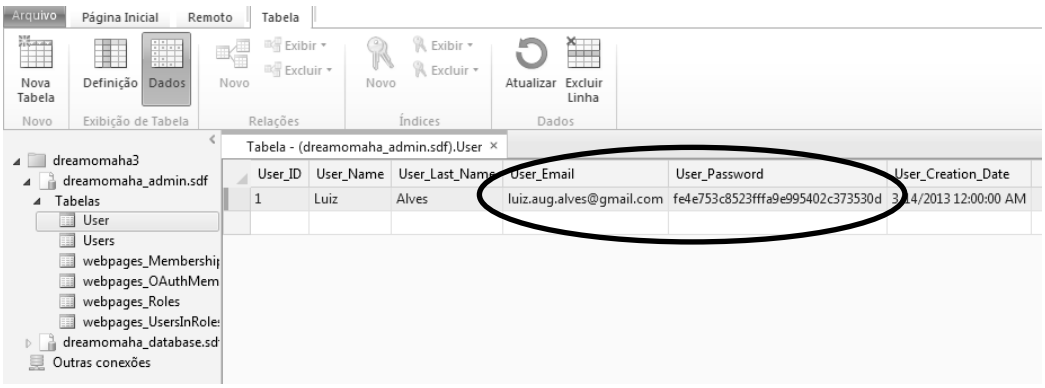


Figura 9. Esta imagem é uma captura de tela do banco de dados onde na tabela User, a senha "ufrj2014" do usuário luiz.aug.alves@gmail.com é codificada para a versão criptografada utilizando o algoritmo MD5.

Ainda sobre a questão da identificação, autenticação e autorização, uma tradicional opção de recuperação de senhas esquecidas foi implementada: uma

pergunta pessoal com uma resposta secreta, que ao ser preenchida retorna um link para criar uma senha nova, mas nunca exibe explicitamente a senha antiga na tela.

Uma das formas mais clássicas de ataque listada como uma das 10 principais vulnerabilidades da web pelo Open Web Application Security em 2010 [11] é a Injeção de SQL, técnica que consiste em aproveitar uma falha de sintaxe da linguagem em que o sistema é implementado para injetar código SQL malicioso através de formulários HTML, com o objetivo de executar comandos no banco de dados como *select* para obter dados ou, mais sério ainda, *drop table* para apagar uma tabela.

Este projeto contempla proteção contra este tipo de ataque, já que internamente sua implementação foi pensada para não permitir que comandos maliciosos sejam validados. A simples query abaixo da maneira como está (concatenando a query em si com uma entrada do usuário), insere na tabela Mentor o valor recebido da variável mentor_first_name:

```
"INSERT INTO Mentor (Mentor_First_Name) VALUES '" + mentor_first_name
+ "';"
```

Se o valor, por exemplo, for “John”, torna-se:

```
"INSERT INTO Mentor (Mentor_First_Name) VALUES 'John';"
```

No entanto, se o valor for algo como “xxxxxx’; DROP TABLE Mentor; --”, temos que a string torna-se:

```
"INSERT INTO Mentor (Mentor_First_Name) VALUES 'xxxxxx'; DROP TABLE
Mentor; --';"
```

A string está dividida em duas partes: a primeira é um comando *insert* que apenas insere o valor ‘xxxxxx’ na coluna Mentor_First_Name da tabela Mentor; e a segunda que é um comando *drop table*, que deleta a tabela mentor da base de dados.

Tudo o que estiver após “--” é considerado comentário e será ignorado pelo compilador. A solução foi utilizar um caracter-coringa (@0) do próprio ASP.NET que parametriza os valores recebidos pelo formulário, sem a necessidade da concatenação explicada acima:

```
"INSERT INTO Mentor (Mentor_First_Name) VALUES @0;"
```

O valor recebido em @0 pode até ser “xxxxxx’; DROP TABLE Mentor; --”, vindo da entrada do formulário, porém este será o conteúdo armazenado na base de dados. Certamente não é o nome de nenhum Mentor, mas também o código malicioso não é executado.

Um dos artifícios mais utilizados para forçar a entrada em sistemas com falhas de segurança é o ataque de dicionário, técnica simples que consiste em uso de força bruta para, dado o nome de um usuário, testar as senhas mais prováveis listadas em uma coleção (dicionário). O fundamento aqui é que muitos usuários optam por senhas fáceis como “123456” ou a palavra “senha” por exemplo, o que gera uma vantagem matemática na hora de advinhar estas. Contudo a falha não é apenas do usuário, pois o sistema desenvolvido precisa de uma política de senhas forte que não comprometa a segurança dos dados (ou pelo minimize os danos aos mesmos) .

A política escolhida pelo desenvolvedor em acordo com o cliente foi de que haveria liberdade na escolha de caracteres, porém em caso de 3 tentativas falhas o sistema seria bloqueado por 24 horas.

4. Casos de uso, testes e resultados

Nesta seção são apresentados cinco casos de uso que são representados por diagramas de sequências, mostrando a interação entre o usuário, os dados e as camadas de abstração da arquitetura do sistema.

4.1 Caso de uso 1

Este caso de uso demonstra uma tentativa de acessar o sistema com um par login/senha inválidos, buscando a combinação no banco de dados e retornando falha. Após uma nova tentativa, desta vez correta, o sistema libera o acesso.

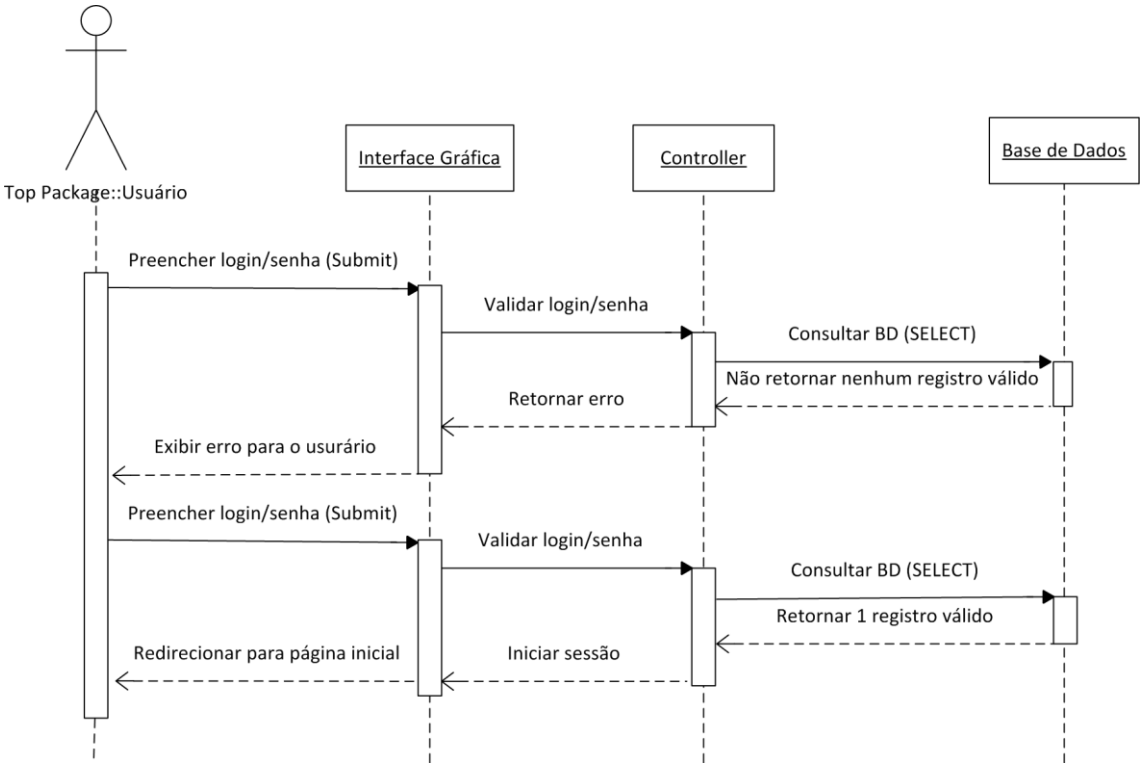


Figura 10. Caso de uso 1.

7.2 Caso de uso 2

Neste exemplo, tenta-se adicionar um mentor novo com algum erro no preenchimento do formulário (algum campo obrigatório que não foi preenchido ou, se foi, de forma incorreta). O controller processa os dados para validá-los e retorna para a interface gráfica uma mensagem de erro. O usuário, agora alertado, preencherá corretamente o formulário e os dados serão inseridos no banco de dados.

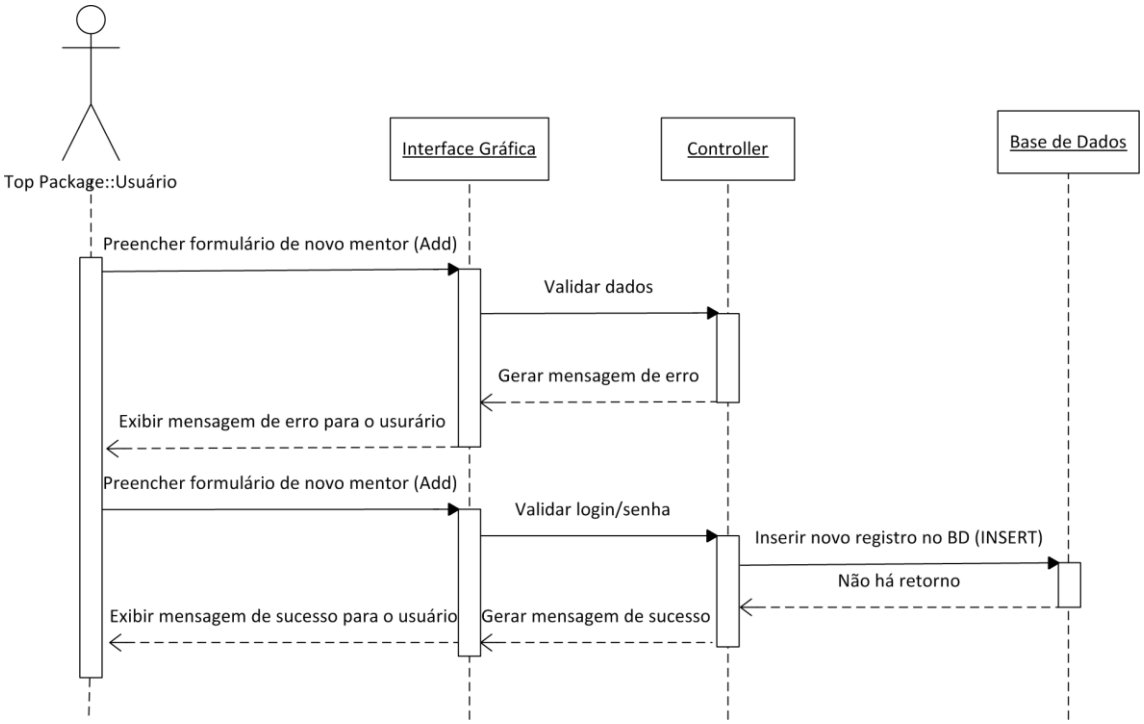


Figura 11. Caso de uso 2.

7.3 Caso de uso 3

O diagrama de sequências abaixo refere-se ao caso de uso em que um novo aluno é adicionado ao sistema e compreende 3 etapas: primeiro as informações básicas são preenchidas, depois uma opção de escolha de mentores a serem associados ao novo aluno é exibida, e finalmente a opção de escolha de um programa do mentor que foi selecionado no passo anterior para ser associado a este aluno é exibida. Por último as informações são adicionadas na base de dados.

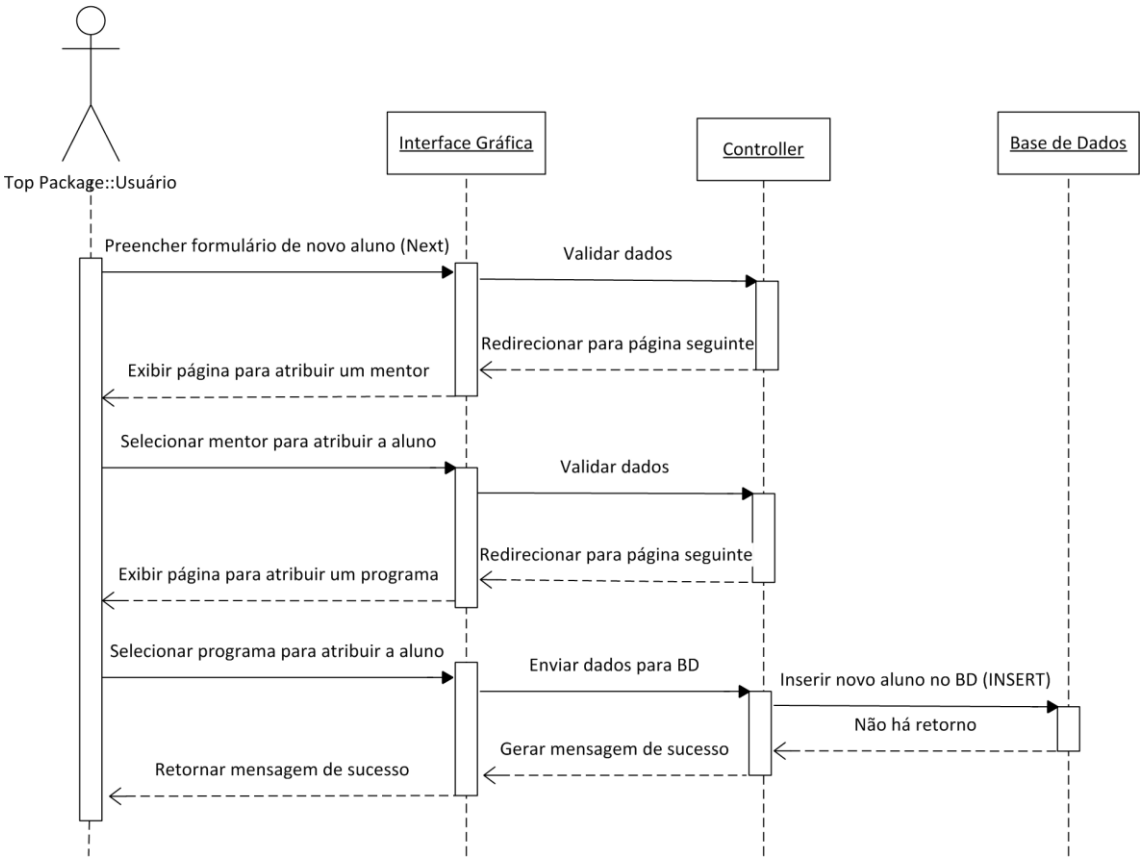


Figura 12. Caso de uso 3.

7.4 Caso de uso 4

Quando um usuário quiser visualizar e depois editar as informações de um mentor já existente, basta selecionar a opção “View/Edit” no menu que um campo de busca será exibido. O usuário pode pesquisar o mentor pelo nome ou pelo sobrenome, selecioná-lo e se desejar alterar os dados, basta editar e clicar no botão para atualizar a base de dados com os novos valores.

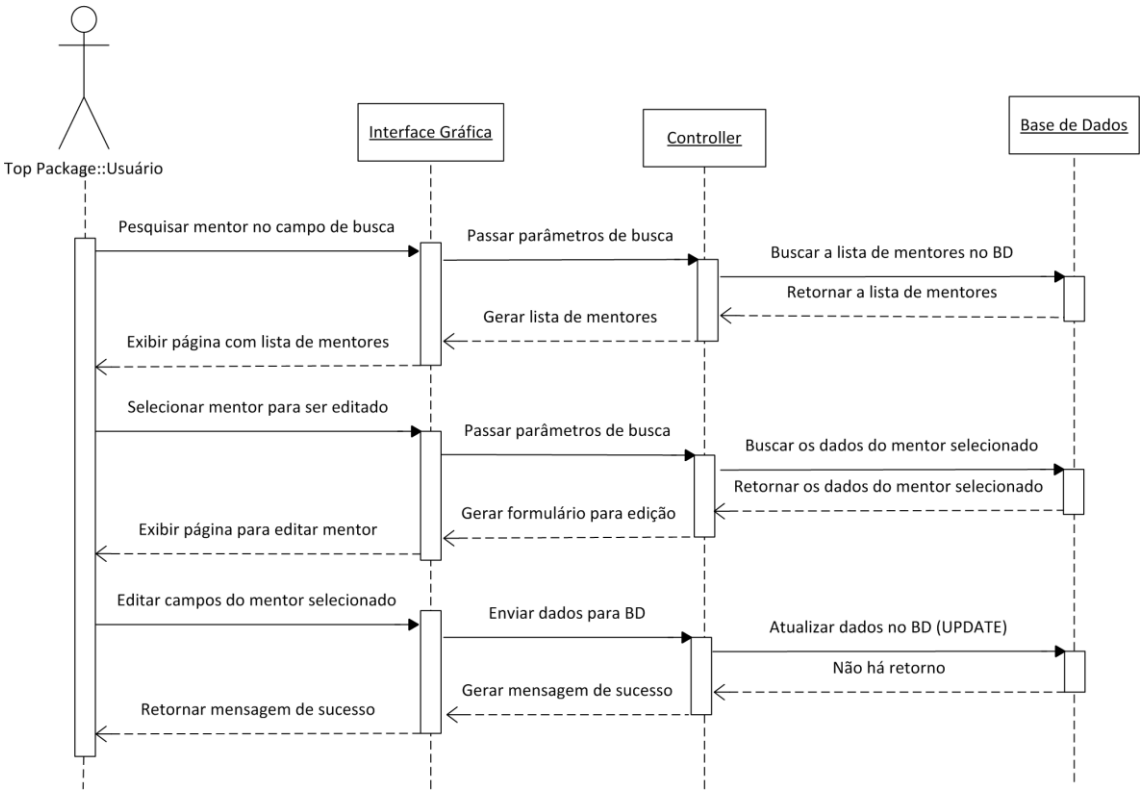


Figura 53. Caso de uso 4.

7.5 Caso de uso 5

O diagrama de sequências abaixo exemplifica os passos executados para que seja visualizado um relatório de alunos por etnia e para que o mesmo seja ordenado de acordo com um critério (ordem alfabética de nome, por exemplo).

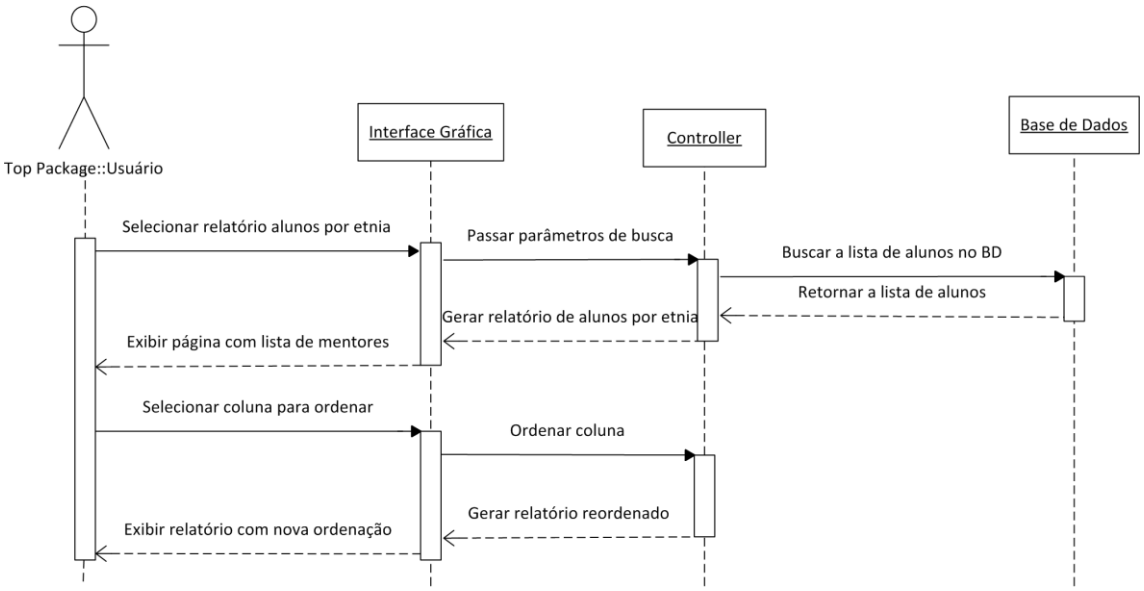


Figura 64. Caso de uso 5.

7.6 Testes e resultados

Os seguintes testes funcionais e de segurança abaixo foram executados sempre pensando nas situações mais importantes. Os respectivos resultados acompanham cada teste:

Tabela 8. Testes funcionais executados.

Teste funcionais	
Teste	**Resultado**
Adicionar mentor	Aprovado
Visualizar/Editar mentor	Aprovado
Adicionar aluno	Aprovado
Visualizar/Editar aluno	Aprovado
Adicionar pai/guardião	Aprovado
Visualizar/Editar pai/guardião	Aprovado
Adicionar escola	Aprovado
Visualizar/Editar escola	Aprovado
Adicionar programa	Aprovado
Visualizar/Editar programa	Aprovado
Adicionar pesquisas (para mentores e alunos)	Aprovado
Visualizar/Editar pesquisas (para mentores e alunos)	Aprovado

Tabela 9. Testes de segurança executados.

Teste de segurança	
Teste	**Descrição**
Login válido	Aprovado
Ataque de dicionário (bloqueio temporário de acesso após 3 tentativas de login)	Aprovado
Finalizar sessão (log out)	Aprovado
SQL Injection (injeção de código malicioso)	Aprovado
Armazenamento de senha criptografada no banco de dados	Aprovado

8. Conclusão

O projeto do Sistema de Gerenciamento da Organização D.R.E.A.M. foi concluído com êxito de acordo com as especificações e atendendo às expectativas do dono do produto: um sistema online que armazena e relaciona em um banco de dados informações sobre a ONG D.R.E.A.M., com operações de inserção, edição e visualização, e de forma segura para proteger os dados sensíveis sobre os jovens.

Vale ressaltar que a escolha do SCRUM como metodologia de projeto de software foi decisiva para que este sistema fosse implementado em alguns poucos meses, demonstrando sua efetiva eficiência especialmente para atender às novas demandas de projetos de pequeno porte, tal como o da organização D.R.E.A.M..

A grande vantagem é a agilidade e a redução da burocracia na execução das histórias de usuário: talvez a opção pelo modelo cascata acarretasse em longas negociações e os próprios processos de mudanças de requisitos são mais lentos. Por outro lado, a desvantagem percebida durante o projeto em relação ao SCRUM é que a falta de um contrato formal que define o que será entregue ao final do projeto (ou pelo menos ao final de cada sprint). Embora formalismo, burocracia e questões legais estejam diametralmente opostos ao conceito de agilidade, isto abre a possibilidade para que o dono do produto queira algo fora do escopo definido nas reuniões de revisão, ou para que a equipe desenvolvedora se recuse a fazer algo fora das especificações para que o andamento dos trabalhos naquele *sprint* não seja prejudicado. Para este trabalho que foi executado de forma voluntária, não houve este problema, embora contratos de escopo variáveis seja um item importante quando do momento da contratação de uma equipe para projeto de software.

O projeto foi implementado e testado com sucesso em um servidor local (localhost), faltando apenas a implantação (deployment) no servidor que o cliente

usará para hospedar a aplicação. Algumas modificações em arquivos de configuração precisariam ser feitas, mas nada relativo ao código do projeto.

Finalmente, como trabalhos futuros a ideia seria lançar funcionalidades que fornecessem estatísticas e gráficos com base nos dados armazenados, além de novas opções de relatórios que ficaram fora do escopo. Somados aos relatórios já existentes, estes elementos estatísticos certamente entregariam ainda mais valor às atividades da organização, fazendo com que o sistema tenha como função não mais apenas o armazenamento e relacionamento de números, datas e sequênicas de caracteres que antes se perdiam em fichas de papel. Por último, destaca-se também um dos propósitos indiciais do projeto: ser uma fonte geradora de conhecimento para auxiliar nos processos administrativos da ONG, concentrado em um único sistema, porém facilmente acessado através de um navegador web de qualquer computador ligado à Internet.

10. Referências

[1] *Site offcial D.R.E.A.M.* Disponível em: <http://joindream.org/about-us/>. Acessado em: 22/03/2013.

[2] *Manifesto para Desenvolvimento Ágil de Software*. Disponível em: <http://agilemanifesto.org/iso/ptbr/>. Acessado em: 22/03/2013.

[3] SOMMERVILLE, Ian. *Engenharia de Software*. 8ª ed., pp 260, 261. Pearson Addison-Wesley. São Paulo. 2007.

[4] *Blog My Scrumhalf*. Disponível em: <http://blog.scrumhalf.com.br/wp-content/uploads/image/figura_sprint.jpg>. Acessado em: 22/03/2013.

[5] TANENBAUM, Andrew S. *Computer Networks*. 4ª ed, p. 609. Prentice Hall. Boston. 2002.

[6] NAVATHE, Shamkant, ELMASRI, Ramez. *Sistemas de Banco de Dados*. 4ª ed., pp. 31, 39. Pearson Addison-Wesley. São Paulo. 2005.

[7] *ASP.NET Architecture*. Disponível em: <http://msdn.microsoft.com/en-us/library/yedba920(v=vs.71).aspx>. Acessado em 22/03/2013.

[8] LAPLANTE, Philip A. *What Every Engineer Should Know about Software Engineering*. p. 85. CRC Press. 2007.

[9] Microsoft Patterns & Practices Team. *Microsoft Application Architecture Guide*. 2ª ed. Microsoft Press. 2009.

[10] KRASNER, *Glenn E., POPE, Stephen T. A cookbook for using the model-view controller user interface paradigm in Smalltalk-80. Journal of Object Oriented Program., Vol. 1, Nº. 3., pp. 26-49. 1988.*

[11] *Open Web Application Security Project - Top 10 Application Security Risks 2010*. Disponível em: <https://www.owasp.org/index.php/Top_10_2010-Main>. Acessado em: 22/03/2012.

Apêndice 1 – Manual de instruções

D.R.E.A.M. Omaha Management System

User's manual

Table of contents

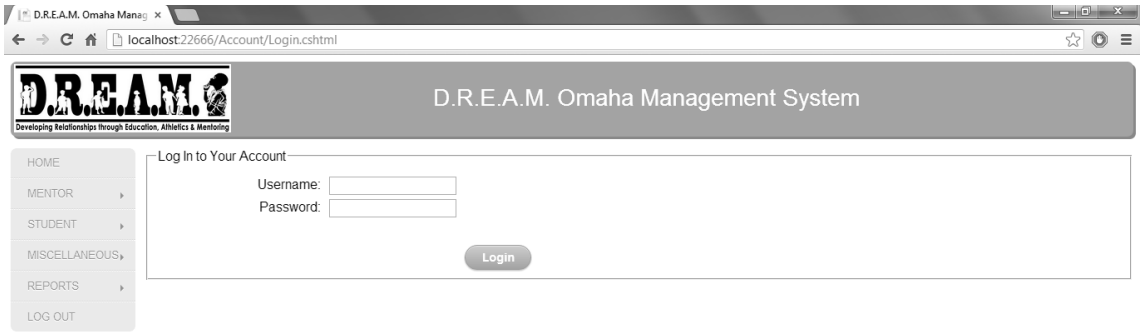
1. Introduction

This manual is the guide to use the D.R.E.A.M. Omaha Management System, an easy and intuitive web-based database developed to store, organize and relate the information about the organization such as students, parents, guardians, mentors, medical and school information.

The system runs better in Google Chrome v. 22 (or above) and Mozilla Firefox v. 17 (or above).

2. Logging into the system

This is the login page, the first page that you access before use the system. Enter your username and the password in these fields and click the “Login” button. You will be redirected to the Welcome page as soon you are authenticated. You can try to access the links in the side menu (on the left), but the access will be denied if you are not logged in. After 12 minutes inactive, the session will expire and you will have to log in again.

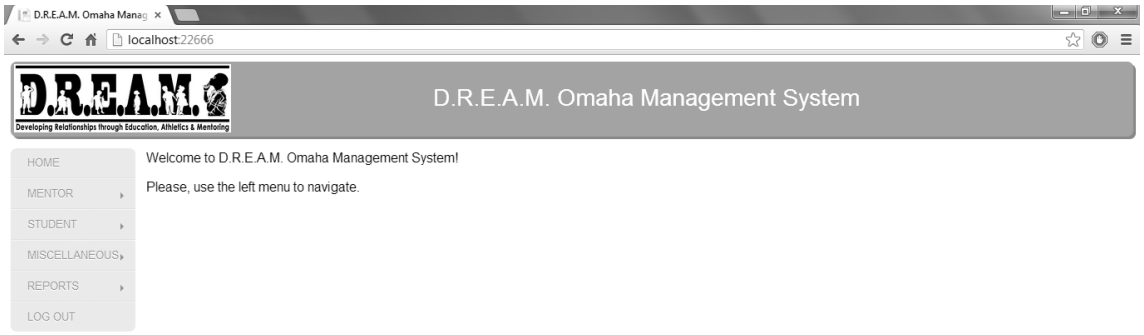


Picture 1

Do not forget to log out when you finish the activities.

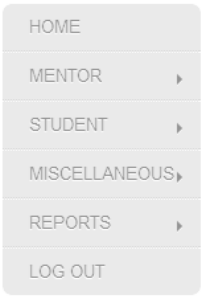
3. Navigating

The welcome screen is the homepage and you can select the sections: Mentor, Student, Miscellaneous or Reports.



Picture 2

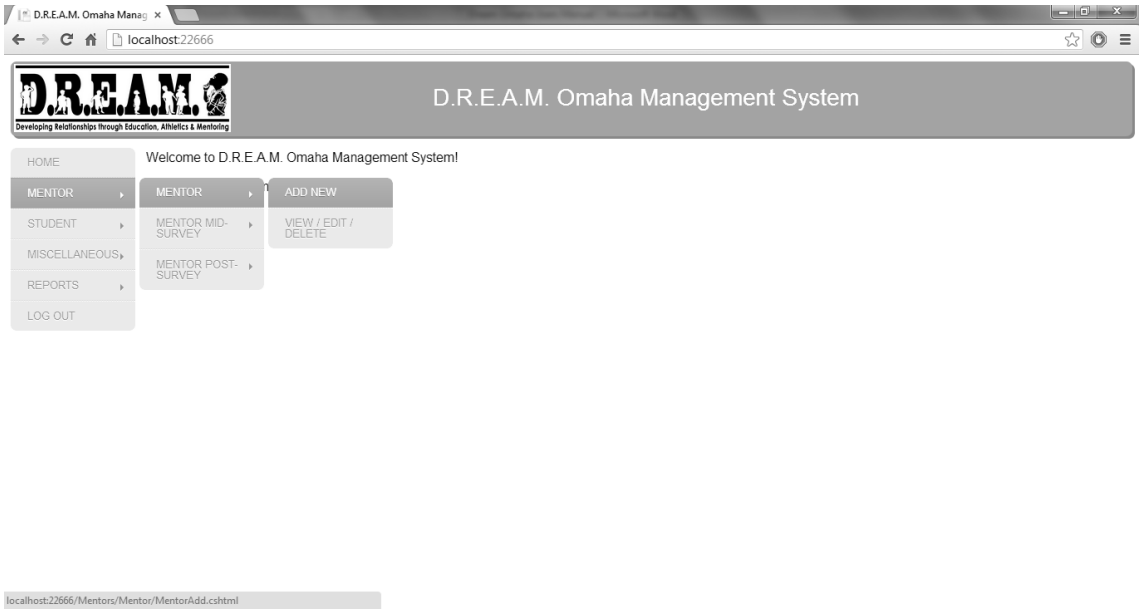
The side menu is used to separate the information in four big areas (Mentor, Student, Miscellaneous and Reports) and subareas.



Picture 3

3.1 Mentor

To add a new mentor, follow Mentor > Mentor > Add new and click to open the blank form.

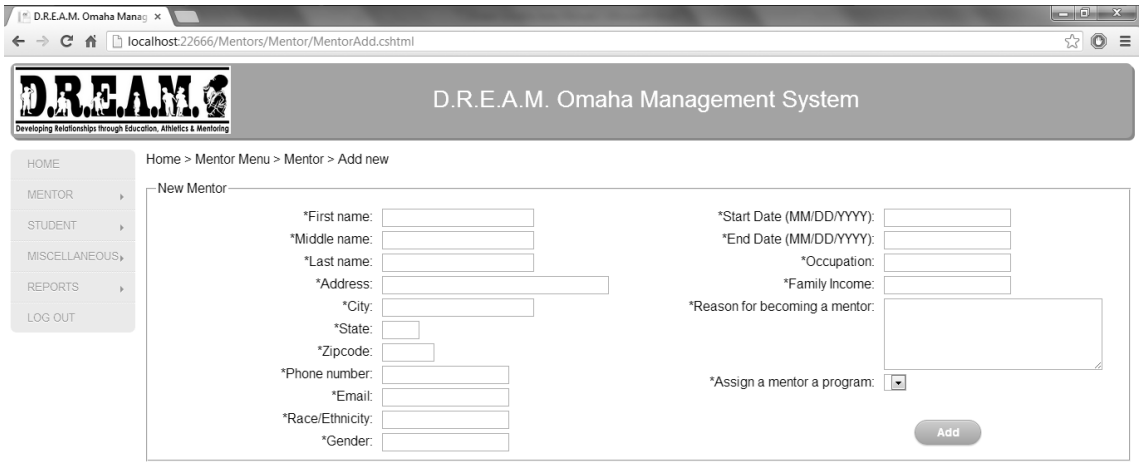


Picture 4

Fill all the required fields marked with *. If you do not fill it correctly, an error message will be issued and the data will not be inserted in the database. In “phone number” and “family income” use **only** numbers. This will make it easier to the search engine look for specific values (This note is used also for others forms).

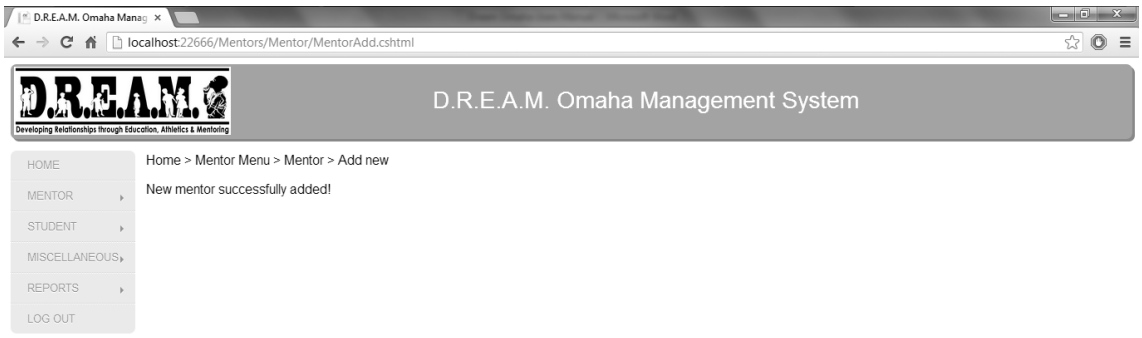
The “Assign a mentor program:” option is required but, firstly the database is empty. No program will be assigned to the mentor. Do not worry because this will happen in the first access, however, later you will have to edit this mentor and add a program. Whenever you see an empty drop-down list it is due to the fact that no option was previously added to the database.

For example, you can add 7 mentors without programs assigned before add the first program, but the list will only display something after the insertion of this first program and then the 8^{th} mentor will automatically have this first program assigned.



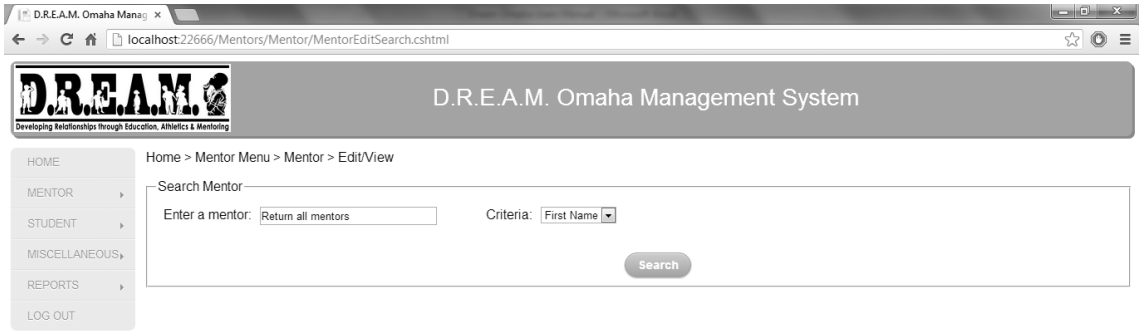
Picture 5

Finally, the transaction will be completed when you see the message "New mentor successfully added!".

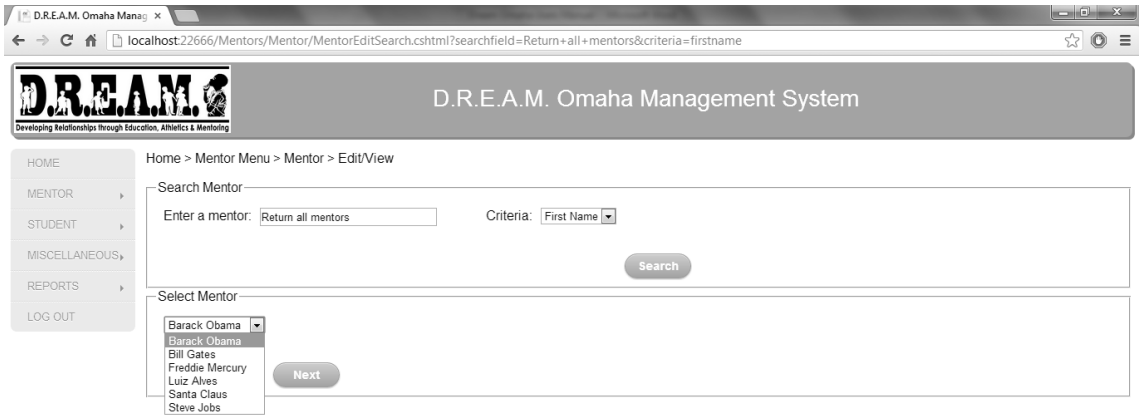


Picture 6

The process to view, edit or delete a record is simple. Go to Mentor > Mentor > View or Edit and click to open the search field. You can click the field and enter a specific mentor according to the search criteria (first name or last name). If you let it with the message “Return all mentors”, then all the mentors in the database will show up and you can select a mentor to edit, view or delete. See the example below:

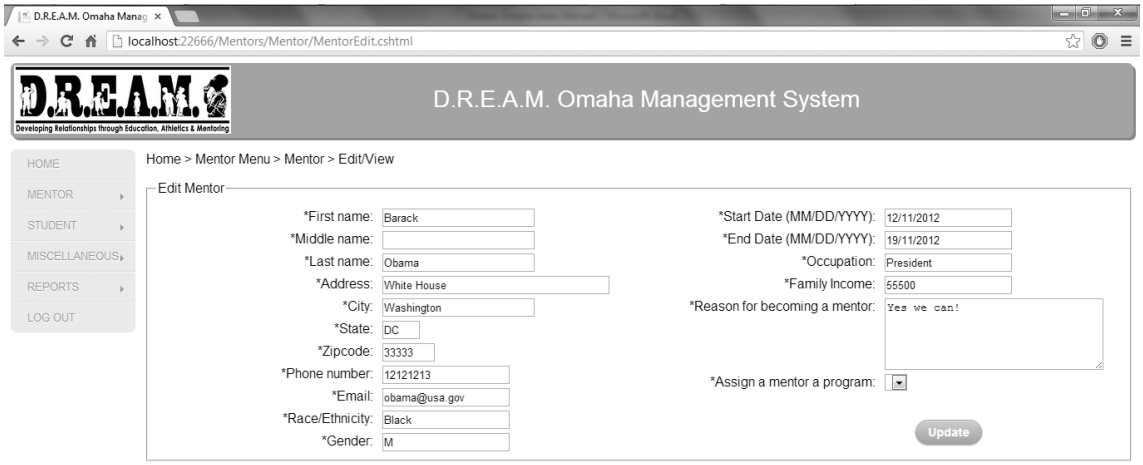


Picture 7



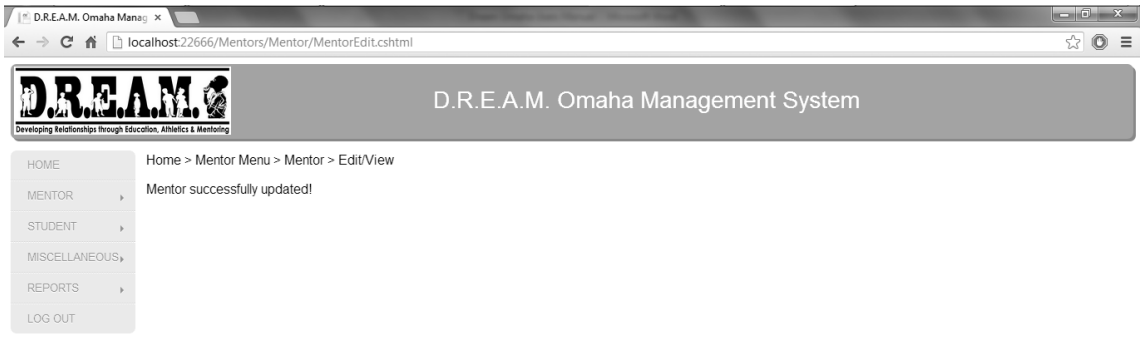
Picture 8

Click “Update” to update the information.



Picture 9

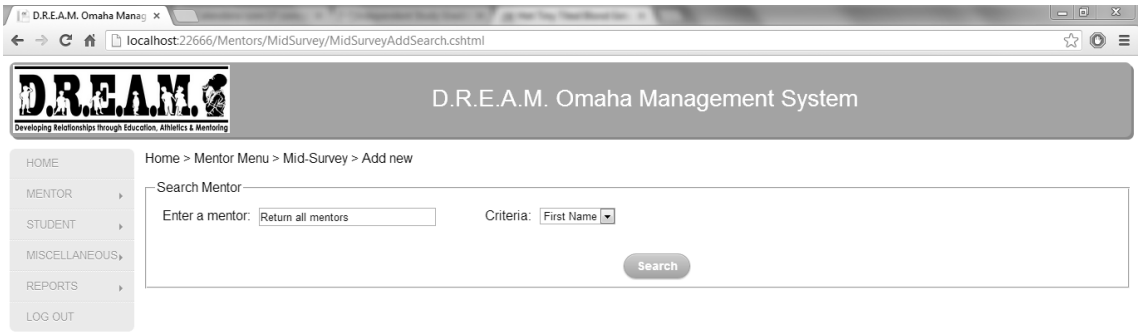
After the message “Mentor successfully updated”, you will be sure that the transaction was completed.



Picture 10

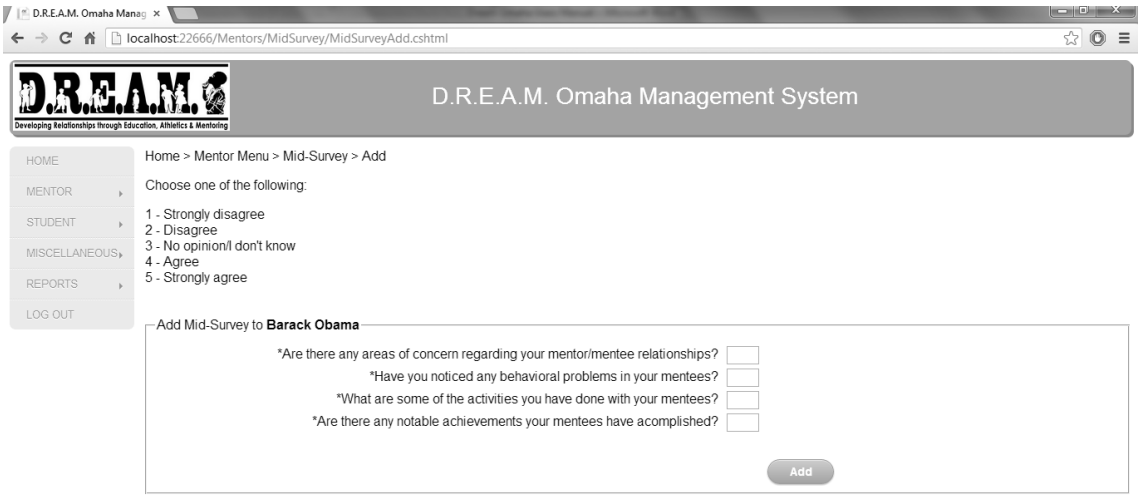
3.2 Mentor Mid-Survey and Mentor Post-Survey

If you want to add a new mid-survey to a mentor, go to Mentor > Mentor Mid-Survey > Add new and click to start the Mentor Mid-Survey process. You will see the same search page from the Edit Mentor process. Repeat the process to choose a mentor and click “Next”. This is necessary because firstly you must indicate the mentor that you will assign the survey.



Picture 11

Choose **only** numbers from 1 to 5 to fill the survey and click “Add”. A message new “Mid-Survey added” will be issued if everything is correct. If not, a message “Fill the required fields” will be issued.



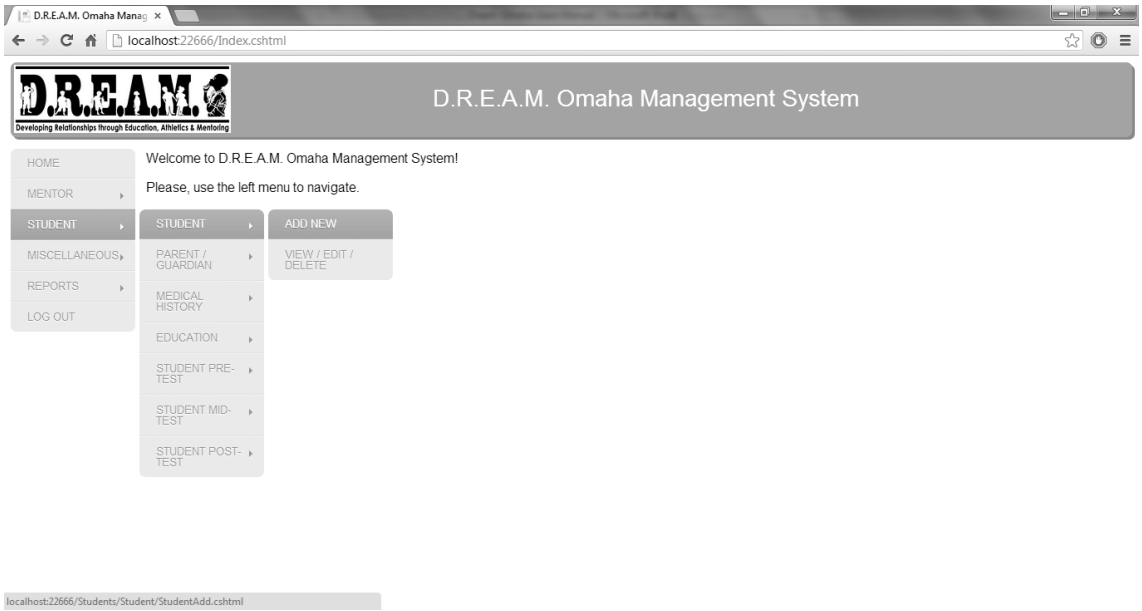
Picture 12

To view or edit a survey, follow Mentor > Mentor Mid-Survey > View / Edit and repeat the process to select a mentor and then view or edit the survey. Click “Update” to confirm the transaction.

Repeat these processes to add, view or edit a Mentor Post-Survey too.

3.3 Student

To add a new student go to Student > Student > Add new. A page similar to the ones that you saw in the Add Mentor menu will show up and you will need to fill the blank form. Click “Add” to confirm the transaction.



Picture 13

For the other areas, just repeat these instructions and follow the directions correctly.

4. Reports

Go to “Reports” in the side menu and choose one of the reports generated from the database. For example, if you choose “Mentee Ethnicity Report” you will see a table displaying a list of mentees and their first and last names, ethnicities, start and end date. If you click on one of the headers you can sort the column according to a certain criteria (alphabetical order for words, chronological order for dates and ascending or descending order for numbers).

You can print it directly from the browser or also send the link through email to somebody else of your interest.


D.R.E.A.M. Omaha Manag
localhost:22666/Reports/MenteeEthnicityReport.cshtml
D.R.E.A.M.
Developing Relationships through Education, Athletics & Mentoring
D.R.E.A.M. Omaha Management System
HOME
MENTOR
STUDENT
MISCELLANEOUS
REPORTS
REGISTER NEW USER
LOG OUT
Home > Reports > Mentee Ethnicity Report
Mentee Ethnicity
First Name
Last Name
Ethnicity
Start Date
End Date
First Name 1
Last name 1
Hispanic
11/09/2012
First Name 2
Last Name 2
Caucasian
03/10/2009
First Name 3
Last Name 3
Black
01/10/2010
02/02/2013

Picture 14

Apêndice 2 – Termo de concordância

Universidade Federal do Rio de Janeiro
Federal University of Rio de Janeiro
Polytechnic School/Department of Electronic Engineering

Authorization Document

We, **D.R.E.A.M.**, a nonprofit organization represented by **Steve J. Warren**, CEO and founder, hereby authorize through this document the legal usage of the **D.R.E.A.M** brand and logo for educational and only for educational purposes in the final undergraduate project of **Luiz Augusto da Silva Alves**, a computer engineering student in the **Federal University of Rio de Janeiro**, Rio de Janeiro, Brazil, that will use the knowledge acquired during the development of the D.R.E.A.M. Management System as a case study. The intention of this letter is to legally protect the Federal University of Rio de Janeiro and Luiz Augusto da Silva Alves from copyright issues and unauthorized usage of D.R.E.A.M. brand.

Steve J. Warren

CEO & Founder

D.R.E.A.M.

Omaha, February 15th 2013