# Coruja

## Conteúdo

* Eixo
  + Níveis
    - Competências

Exemplo:

* Eixo = Português
* Nível = Ano 1, Semestre 1
* Competência = Compreensão de Texto => PIVk (Interpretar frases e expressões com autonomia)

## Teste

* Teste
  + Conteúdo Específico (Eixo, Nìvel)
* Questão
  + Conteúdo Específico (Competência)
  + 5 Alternativas
    - Cada uma direciona o aluno para outra questão

## Professor

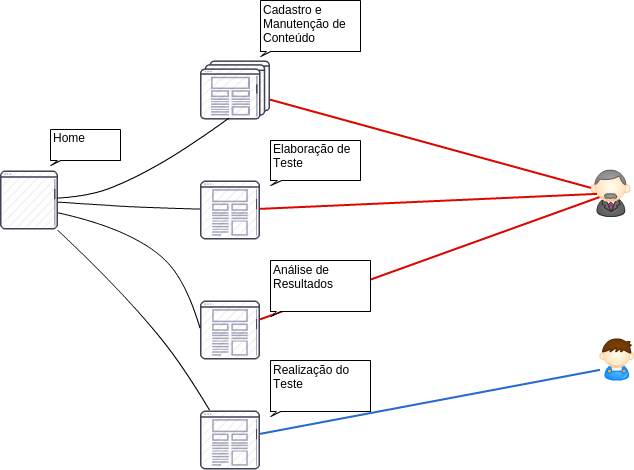
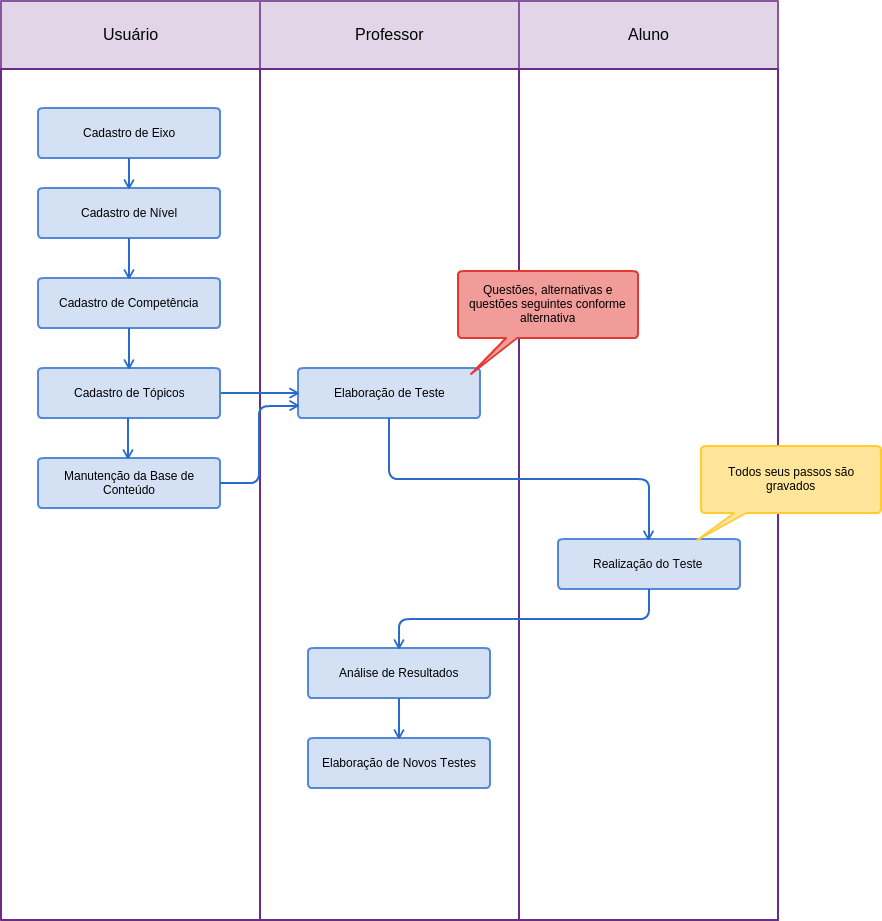
* Cadastra e faz manutenção do conteúdo
* Elabora os Testes
* Analisa Resultados

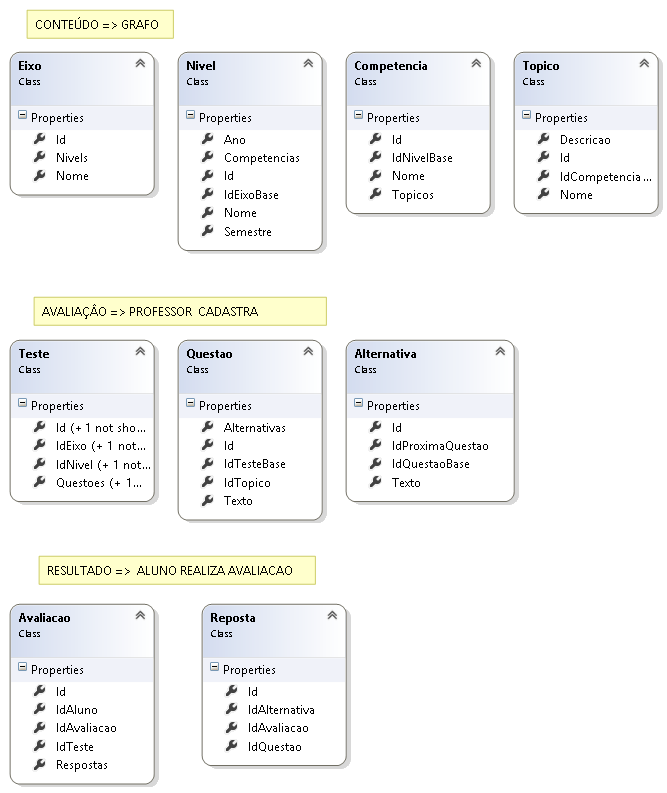
## Aluno

* Realiza o Teste
* Tem seu Teste gravado (Questão => Resposta)

# Sequência de Eventos

* Professor cadastra e faz manutenção do conteúdo
* Provessor elabora teste (questões e associação entre alternativas e questões)
* Aluno faz o teste (teste é gravado)
* Professor analisa os resultados (relatórios por Eixo, Nível, Competência, Aluno, etc)
* A partir da análise o ensino pode ser redirecionado
* Professor elabora novos testes (tem também a possibilidade de adicionar questões gravadas de outros testes)





//A seguir estão os métodos básicos para se atingir o objetivos propostos

//Usei sintaxe básica do MVC5 e entity frmawork 6.0

//instancia DbContext(acesso ao banco de dados)

db = DbContext;

//Elaboração de Teste

//Adiciona-se um Teste Novo que ganha um ID

//método recebe como parâmetros o Id de referencia do Eixo e do Nível

AdicionarTeste(int \_IdEixo, int \_IdNivel)

{

//Novo Objeto Teste

var novoTeste = new Teste();

//O teste é linkado a um Eixo e Nivel (ex: Portugues, ano 1 semestre 1)

novoTeste.IdEixo = \_IdEixo;

novoTeste.IdNivel = \_IdNivel;

//O novo teste é adicionado à base de dados

db.Testes.Add(novoTeste);

db.SaveChanges();

}

//Adiciona-se uma Questão ao novo Teste

//Questão tem ID único

//método recebe como parâmetros o Id de referencia do Teste cadastrado, O Id de referência do Tópico e o Texto do enunciado

AdicionarQuestao(int \_IdNovoTeste, int \_IdTopico, string \_Texto)

{

//Novo Objeto Questao

var novaQuestao = new Questao();

//A questao é vinculada ao teste cadastrado

novaQuestao.IdTesteBase = \_IdTeste;

//A questao é vinculada a um Topico (PIVk) de Id único que está vinculado a uma Competencia (Compreensão de Texto)

novaQuestao.IdTopico = \_IdTopico;

//É adicionado o enunciado da questao

novaQuestao.Texto = \_Texto;

//A questão é adicionada ao Novo Teste

//Quando a questão é adicionada à base de dados, ela recebe um Id único (ordem da questao)

db.Questaos.Add(questao);

db.SaveChanges();

}

//Adiciona-se uma alternativa à nova questão

//método recebe como parâmetros o Id de referencia da nova questão cadastrada, O Id de referência da próxima questão e o Texto do enunciado

AdicionarAlternativa(int \_IdNovaQuestao, int \_IdProximaQuestao, string \_Texto)

{

//Novo Objeto Alternativa

var NovaAlternativa = new Alternativa();

//A nova alternativa é linkada à qestão

NovaAlternativa.IdQuestaoBase = \_IdNovaQuestao;

//A nova alternativa é linkada à próxima questão conforme alternativa escolhida

NovaAlternativa.IdProximaQuestao = \_IdProximaQuestao;

//É adicionado o enunciado da alternativa

NovaAlternativa.Texto = \_Texto;

//A alternativa é adicionada à questão

db.Alternativas.Add(NovaAlternativa);

db.SaveChanges();

}

//Ao finalizar o Processo acima temos um Teste cadastrado com várias questões e alternativas para cada questão.

//O Aluno pode entrar no sistema e realizar o teste

//No próximo passo o Aluno inicia a avaliação realizando o Teste Cadastrado

//Ao logar no sistema o aluno tem à disposição o Teste de acordo com seu nível, eixo

IniciarAvaliacao (int \_IdAluno)

{

//Novo Objeto Avaliacao

//Esses dados serão armazenados no banco de dados e registram o caminho percorrido pelo aluno

var novaAvaliacao = new Avaliacao();

//Avaliacão é linkada ao Id do Aluno

novaAvaliacao.IdAluno = \_IdAluno;

//Avaliacão é linkada ao Id do Teste em questão

novaAvaliacao.IdTeste = \_IdTeste;

//A avaliacao é registrada na base de dados

db.Avaliacoes.Add(novaAvaliacao);

db.SaveChanges();

}

// A tela da avaliação apresenta uma questão por vez de modo que após responder e salvar a resposta, a tela é direcinada para a próxima questão de acordo com a alternativa escolhida

SalvarResposta(int IdNovaAvaliacao, int IdQuestao, int \_IdAlternativa, \_IdProximaQuestao)

{

//Novo Objeto Resposta

var novaResposta = new reposta();

//resposta é linkada à avaliação

novaResposta.IdAvaliacao = \_IdNovaAvaliacao;

//resposta é linkada à Questão

novaResposta.IdQuestao = \_IdQuestao;

//resposta recebe Id da alternativa escolhida

novaResposta.IdAlternativa = \_IdAlternativa;

//Resposta contendo questão e alternativa são registradas na base

db.Respostas.Add(novaResposta);

db.SaveChanges();

//Seleciona e retorna a proxima questao

ProximaQuestao(int idProximaQuestao)

//Nova Questão é mostrada na tela

}

ProximaQuestao(int \_idProximaQuestao)

{

var novaQuestao = db.Questoes.FindById(idProximaQuestao);

return novaQuestao

}

// O Teste realizado pelo aluno pode ser consultado com o caminho todo mapeado

ConsultarTeste(int \_IdAluno, int \_)IdTeste)

{

var avaliacaoAluno = db.Avaliacoes.

.Where(x => (x.IdAluno == \_IdAluno && x.IdTeste == \_IdTeste)

.ToList();

// respotasAvaliacao contem todas as respostas do aluno no referido teste

var respotasAvaliacao = db.Respostas

.Where(x => (x.IdNovaAvaliacao == avaliacaoAluno.Id)

.ToList();

}