**Instituto Federal de Educação, Ciência e Tecnologia da Paraíba**
Unidade Acadêmica de Informática
Curso Superior de Tecnologia em Sistemas para Internet
Disciplina: Programação para a Web II
Professor: Frederico Guedes Pereira

***Projeto Colaborativo II: Caderneta em Spring***

1) Requisitos Funcionais

Deseja-se um software web em Spring que tenha um CRUD para cadastrar estudantes e um relatório para listar a situação final de cada estudante cadastrado.

Cada estudante deve ter as seguintes propriedades:

- Nome (String)
- Data nascimento (java.util.Date)
- Faltas (Integer) [0-100]
- Nota 1 (BigDecimal) [0-100]
- Nota 2 (BigDecimal) [0-100]
- Nota 3 (BigDecimal) [0-100]
- Final (BigDecimal) [0-100]
- Situacao (“AP”, “RP”, “FN”, “MT”, “RF”), que significam aprovado, reprovado por nota final, está na final, matriculado, reprovado por faltas, respectivamente). Não use String para este campo. Implemente uma Enumeration para definir os valores acima. (veja como em https://www.devmedia.com.br/enums-no-java/38764).

Os únicos campos obrigatórios no cadastro de um aluno são o nome e a data de nascimento. As notas (1 a 3) podem ser digitadas a qualquer momento e não precisam ser digitadas todas ao mesmo tempo, isto é, você pode digitar a nota 1 depois de prova 1 e digitar a nota 2 depois da prova 2, com um mês entre elas. O campo final só é usando se o aluno fez a prova final. Para ir a final, a média das 3 notas deve ser maior ou igual a 4 e o aluno não pode ter mais de 25 faltas. As faltas podem ser informadas a qualquer tempo.

O campo situação é um campo calculado em função dos outros segundo esta tabela:

| Situação do aluno | Valor do campo Situação |
|---|---|
| Nenhuma nota digitada | MT |
| Faltando alguma nota ser digitada (1 ou +) | MT |
| Todas as notas digitadas e sem faltas digitadas | MT |
| Todas as notas digitadas e as faltas digitadas | AP (se média das notas for >= 70 e Faltas < 25)<br>FN (se média < 70 e média >= 40 e Faltas <25)<br>RF (se Faltas >= 25, independentemente da média)<br>RP (se média < 40 e Faltas < 25) |
| Todas as notas digitadas, faltas e final | AP (se (media*60 + final*40)/100 >= 50)<br>RP (caso contrário) |

O sistema deve ter uma tela de login para um usuário administrador com senha criptografada em banco.

A tela principal deve ter um menu superior do Bootstrap com as opções **Cadastro** e **Relatório**. A opção **Cadastro** deve ter os subitens de menu **Estudante** para cadastrar/modificar um estudante (nome e data nascimento) e um subitem **Notas** que liste todos os estudantes cadastrados e permita a digitação das 3 notas , das faltas e da final. O menu **Relatório** deve gerar um relatório contendo os nomes, datas de nascimentos, média, final (se tiver) e situação. Este relatório pode ser consultado em diversos momentos e poderá mostrar diferentes valores para a situação de um aluno. Por exemplo: suponha que você cadastrou 10 alunos. Depois, cadastrou a primeira nota dos 10 alunos. Se consultar o relatório, a média não será computada (pois só há 1 nota digitada), a final também

não apresentará valor, mas o campo situação constará "Matriculado" para todos os 10. Se depois de digitar as 3 notas e as faltas o relatório for solicitado, ele poderá mostrar **Aprovado**, **Reprovado**, **Reprovado por Falta** ou **Em Final** para as situações dos alunos. Se depois de feita a final e informadas as notas, só deverá constar **Aprovado**, **Reprovado** ou **Reprovado por Falta** na situação. As notas, faltas e valores da final também aparecerão no relatório quando tudo tiver sido digitado.

2) Requisitos não funcionais
   a) Utilizar o Spring MVC ou Spring Boot na implementação.
   b) Utilizar Bootstrap.
   c) Qualquer banco de dados.
   d) Utilizar o Hibernate como provider de persistência.
   e) Utilizar campos de mensagens para mostrar erros em todas os formulários.
   f) Utilizar o padrão Post_Redirect_Get.
   g) Utilizar sessões para permitir o login e logout do administrador (o nome dele deve ficar no menu superior, à direita, próximo do botão "Sair" (veja sistema spring-banco).
   h) Utilizar fragmentos para os templates Tymeleaf.

3) Tabela de avaliação de itens

| Item | Pontos |
|---|---|
| Implementou inclusão/alteração de estudantes (nome e data de nascimento) | 10 |
| Implementou o login/logout | 10 |
| Implementou a digitação de notas | 10 |
| Implementou a digitação de faltas | 5 |
| Implementou a digitação da nota final | 5 |
| Implementou o relatório | 25 |
| Utilizou Bootstrap | 10 |
| Formulários validam valores de campos e mostram mensagens de erros | 10 |
| Usou Post-Redirect-Get | 10 |
| Utiliou fragmentos nos templates Thymeleaf | 5 |
| TOTAL | 100 |

4) Recomendações:
   a) Utilizar um nome próprio para o sistema. Evite: "projetospring", "projetofred", "projetopweb2", "projetojavaee". Prefira: "Saturno[1]", "GoogleNotas", "DIP (Did I pass?)", "CadeMinhaNota", "CartolaAC" etc.
   b) Por um arquivo TXT com membros da equipe da raiz do projeto Eclipse.
   c) Usar Maven e Git.
   d) Um vídeo deve ser gravado pela equipe demonstrando todas as funcionalidades (ou bugs) do sistema rodando. Funcionalidade não apresentada no vídeo será considerada não implementada. O vídeo deve mostrar também a arquitetura do código fonte. Não precisa mostrar classe a classe, só o esquema geral ("aqui implementamos controladores para não por a lógica dentro dos backing beans", "não vamos mostrar os DAOs porque o Spring gera automático" etc).

[1] Planeta mais bonito do sistema solar.

e) Deem um nome à equipe. Mais uma vez, seja criativo. O formulário de submissão solicitará 3 coisas: nome do aluno, nome da equipe e link para o vídeo no Youtube ou outro site.
f) Não comecem a fazer na última semana, pois não dará tempo. Ponto! O melhor momento para começar foi ontem, o segundo melhor é agora.
g) Não será possível dilatar o prazo pois será a última semana de aula do semestre letivo.