APP WEB MVC com acesso a BD – Instruções

ITA – Desenvolvimento Ágil com Java Avançado (Coursera)

Aluno: Renato Sampaio Albuquerque

O objetivo dessa atividade é criar uma aplicação web mais completa que faz acesso ao banco de dados. A atividade é para ser desenvolvida utilizando Servlets, JSP e JDBC, seguindo o modelo MVC. Nenhum outro framework deve ser utilizado.

A aplicação a ser desenvolvida é um sistema simples de fórum, onde os usuários podem adicionar tópicos e comentários aos tópicos. Eles recebem pontos pelas ações que fazem no sistema e uma tela de ranking mostra a colocação dos usuários.

Abaixo segue a descrição das telas do sistema:

- **Tela Login** - Essa é a tela inicial. Possui os campos login e senha para o usuário se autenticar. Se o usuário se autenticar com sucesso, ele deve ser redirecionado para a ***Tela Tópicos***. Se não se autenticar com sucesso, o usuário deve ser direcionado para essa mesma tela, que deve exibir uma mensagem de erro. Deve possuir o link para a ***Tela Cadastro***.

- **Tela Cadastro** - Essa tela possui um formulário em que o usuário deve entrar com suas informações (nome, login, email e senha) e se cadastrar. Ao inserir as informações do usuário, ele é direcionado a ***Tela Login***.

- **Tela Tópicos** - Essa tela possui uma tabela com os tópicos criados pelo usuário. Cada tópico na tabela irá exibir o título e o nome do usuário que o criou. Ao clicar em um tópico, o usuário vai para a ***Tela Exibe Tópico***. Existe um link que vai para a ***Tela Ranking*** e um link que vai para a ***Tela Insere Tópico***.

- **Tela Exibe Tópico** - Essa tela exibe o tópico com o título do tópico, o nome de quem o colocou e seu texto. Abaixo do tópico são exibidos os comentários adicionados. Cada comentário possui somente o texto e o nome de quem o inseriu. Abaixo de tudo, existe um campo grande de texto e um botão para adicionar um novo comentário. Ao adicionar o comentário, o usuário deve ser redirecionado para a mesma ***Tela Exibe Tópico***. Existe um link para voltar para a ***Tela Tópicos***.

- **Tela Insere Tópico** - Essa tela apresenta um formulário para para adicionar o título de um tópico e um campo grande de texto para colocar seu conteúdo. Ao submeter, o tópico é inserido e o usuário é redirecionado para a ***Tela Tópicos***.

- **Tela Ranking** - Essa tela apresenta a lista dos usuários ordenados pela quantidade de pontos que possuem. Deve haver uma tabela com a colocação, nome, login e quantidade de pontos. Quando um usuário inserir um tópico, ele deve ganhar 10 pontos e quando adicionar um comentário deve ganhar 3 pontos. Existe um link para voltar para a ***Tela Tópicos***.

Abaixo estão descritos os requisitos técnicos da aplicação:

- O design das telas pode ser criado a critério dos usuários e não tem problema usar um design bem simples e focar mais na funcionalidade.

- Deve-se utilizar uma estrutura MVC, separando as responsabilidades das classes nas camadas. Sugere-se inclusive, criar uma camada intermediária entre os Servlets e as classes que acessam o Banco de Dados.

- Deve-se utilizar Servlets como a camada de controller, JSP como a camada de view e JDBC para o acesso ao banco de dados. Não é permitido a utilização de outros frameworks (uma exceção seria usar componentes de interface apenas para a parte visual da tela).

- Devem ser entregues os testes das classes que acessam o banco de dados com o DBUnit e pelo menos 3 testes com o Selenium que envolva a navegação entre pelo menos 2 telas.

- A estrutura de banco de dados utilizada deve ser a especificada na seção "Estrutura do banco de dados"

***Observação:** A classe desenvolvida no exercício da Semana 3 pode e deve ser aproveitada!*

Quando a aplicação ficar pronta, você deve gravar um vídeo da tela do seu computador navegando pela interface da aplicação. Esse vídeo deve mostrar: o cadastro de um usuário, a execução do login, a listagem de tópicos, a criação de um tópico, a exibição de um tópico, a criação de um comentário e a exibição do ranking. Deve-se mostrar que a pontuação do usuário aumenta quando ele cria tópicos e adiciona comentários. Deve ser feito o upload desse video para um repositório de acesso público (como o YouTube - se quiser pode deixar ele acessível a partir do link mas não listado publicamente) onde os revisores possam assisti-lo a partir do link.

PS: Não use textos ou palavras ofensivas nos exemplos que criar!

Deve ser entregue:

- O projeto da aplicação (no Eclipse ou Netbeans) com todas as classes, as páginas e os testes de unidade em um arquivo do tipo .zip

- O código da classe com os testes do Selenium em formato .java (coloque todos os testes em uma classe única para submissão)

- Link do video que demonstra a utilização da aplicação web desenvolvida

Review criteria

Será considerado na avaliação:

- Se as funcionalidades pedidas foram implementadas.
- Se os requisitos funcionais foram atendidos.
- Se a implementação utilizou as tecnologias solicitadas.
- Se as camadas foram divididas de forma adequada.
- Se o código está organizado.
- Se os testes foram criados como solicitado.

Estrutura do banco de dados

```
CREATE TABLE usuario
(
  login text NOT NULL,
  email text,
  nome text,
  senha text,
  pontos integer,
  CONSTRAINT usuario_pkey PRIMARY KEY (login)
)
```

Deve ser criada uma classe Usuario com as informações presentes na tabela e uma interface com os seguintes métodos:

```
CREATE SEQUENCE topico_id_topico_seq
  INCREMENT 1
  MINVALUE 1
  MAXVALUE 9223372036854775807
  START 1
  CACHE 1;

CREATE TABLE topico
(
  id_topico integer NOT NULL DEFAULT nextval('topico_id_topico_seq'::regclass),
  titulo text,
  conteudo text,
  login text,
  CONSTRAINT topico_pkey PRIMARY KEY (id_topico),
  CONSTRAINT topico_login_fkey FOREIGN KEY (login)
      REFERENCES usuario (login) MATCH SIMPLE
      ON UPDATE NO ACTION ON DELETE NO ACTION
);
```

```
CREATE SEQUENCE comentario_id_comentario_seq
  INCREMENT 1
  MINVALUE 1
  MAXVALUE 9223372036854775807
  START 1
  CACHE 1;

CREATE TABLE comentario
(
  id_comentario integer NOT NULL DEFAULT nextval('comentario_id_comentario_seq'::regclass),
  comentario text,
  login text,
  id_topico integer,
  CONSTRAINT comentario_pkey PRIMARY KEY (id_comentario),
  CONSTRAINT comentario_id_topico_fkey FOREIGN KEY (id_topico)
      REFERENCES topico (id_topico) MATCH SIMPLE
      ON UPDATE NO ACTION ON DELETE NO ACTION,
  CONSTRAINT comentario_login_fkey FOREIGN KEY (login)
      REFERENCES usuario (login) MATCH SIMPLE
      ON UPDATE NO ACTION ON DELETE NO ACTION
)
```