Trabalho Final (2015/1)

Disciplina de Técnicas de Programação

A) Data de entrega e apresentação: 30/06/2015

B) Objetivo:

O objetivo deste trabalho é consolidar o conhecimento sobre conceitos e construção de sistemas empresariais orientados a objetos em arquiteturas multicamadas através da exploração dos tópicos discutidos na disciplina de Técnicas de Programação.

C) Enunciado do problema:

Estamos interessados em um sistema de informação para o controle de leilões.

Os leilões suportados pelo sistema são de duas naturezas:

- **Leilão de demanda**: Compradores fazem lance de demanda para comprar um lote que o vendedor pretende disponibilizar pelo maior preço. Vence o participante que fez o maior lance de demanda primeiro entre todos os lances, desde que seu lance seja maior do que o preço mínimo estabelecido pelo vendedor.
- **Leilão de oferta:** Vendedores ofertam um lote que o comprador pretende adquirir pelo menor preço. O preço do lote leiloado é determinado pelos vendedores a cada lance. Vence o participante que fez o menor lance de oferta primeiro entre todos os lances, desde que seu valor seja inferior ao preço máximo estabelecido pelo comprador.

O sistema deve suportar ainda duas formas de realização de lances:

- **Aberto:** A qualquer momento é possível verificar o valor de todos os lances já gerados e qual o lance vencedor no momento.
- **Fechado:** Não é possível saber os lances já realizados. Apenas ao final do leilão os valores são tornados públicos.

Ao ser criado um leilão no sistema, deve ser indicado qual a natureza (oferta ou demanda) e qual a forma de lances (aberto ou fechado). Também é necessário informar data/hora de início (a partir da qual são aceitos lances) e término (a partir da qual não são mais aceitos lances). Todo leilão deve possuir a identificação do usuário vendedor (no caso de leilões de demanda) ou comprador (no caso de leilões de oferta). Por fim, todo leilão possui um preço (mínimo ou máximo) associado ao lote.

Lote é um conjunto formado de um ou mais bens que serão leiloados. Os bens a serem leiloados devem possuir uma descrição breve, uma descrição completa e uma categoria (por exemplo, mobiliário, informática, etc).

Sobre os lances, é necessário conhecer data e hora que foi realizado, valor e identificador de qual usuário realizou o lance.

Sobre os usuários do sistema, é necessário conhecer pelo menos nome, cpf ou cnpj e endereço de e-mail.

Ao final do período que um determinado leilão está aberto, o sistema deve determinar o vencedor do leilão.

Em suma, o sistema deve permitir os seguintes casos de uso:

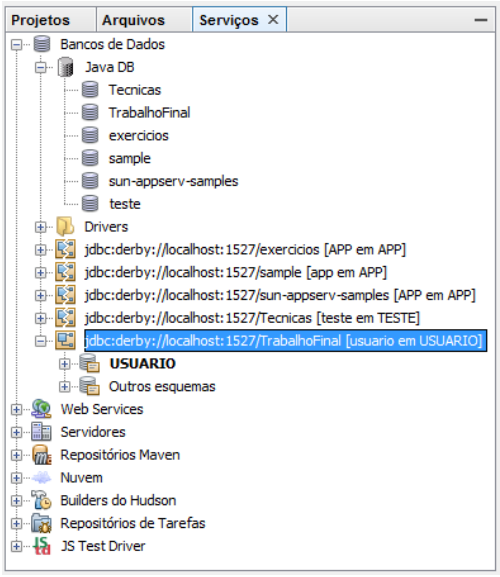
- Cadastrar usuários;
- Cadastrar leilão e bens a serem leiloados;
- Visualizar detalhes de um leilão já terminado ou em andamento;
- Cadastrar lance em um leilão em andamento;
- Cancelar lance em um leilão em andamento.

D) Base de dados:

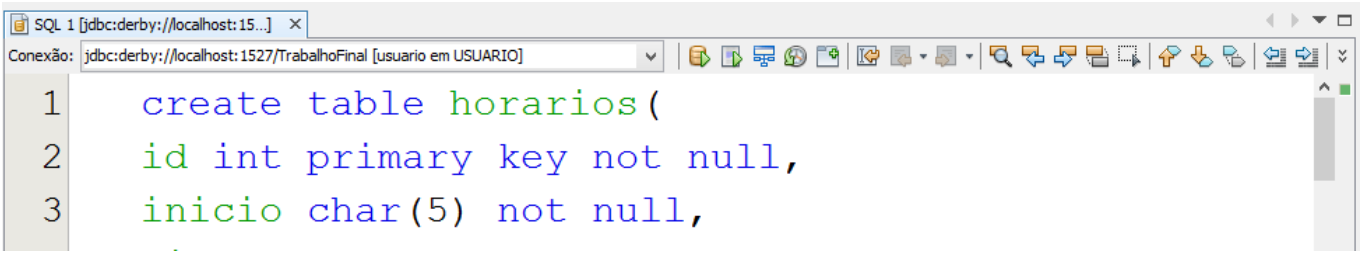
O trabalho deve fazer uso de uma base de dados externa, ou seja, não embutida, de modo a disponibilizar o requisito cliente-servidor do sistema. Importante: o driver JDBC a ser utilizado se encontra no pacote "derbyclient.jar".

Siga os seguintes passos para configurar um servidor externo de banco de dados JavaDB (Derby).

- Inicie o NetBeans.
- Clique na aba "Serviços". Localize os nodos "Bancos de Dados - JavaDB" e com o botão direito do mouse selecione a opção "Iniciar Servidor" para inicializar o servidor de banco de dados. Depois de inicializado, clique com o botão direito e selecione a opção "Criar Banco de Dados..." para criar uma nova base de dados com o nome "TrabalhoFinal". Configure um usuário ("usuario") com senha ("senha"). Como resultado, será criado um link de conexão com o banco de dados. Dê um duplo clique sobre ele para estabelecer uma conexão e verificar se está tudo correto.



- Com a conexão estabelecida, clique com o botão direito sobre a mesma e selecione o menu "Executar Comando...". Na janela que se abriu, copie e cole o script de criação das tabelas. Para executar o script, clique no primeiro botão ao lado do menu drop-down (é o botão com uma seta verde sobre o ícone de base de dados).



E) Requisitos:

Os seguintes itens são obrigatórios na implementação do sistema:

- Arquitetura multicamada (pelo menos 3).
- Uso dos padrões de projeto explorados em sala de aula, sendo obrigatoriamente:
  - Uso do padrão "Facade" para isolar a camada de domínio da camada de apresentação;

    - Uso do padrão arquitetural “Domain Model” na camada de domínio;
    - Uso do padrão “DAO” na camada de persistência.
- Interface gráfica de usuário desktop (interface textual de console não será aceita);
- Persistência em banco de dados relacional;
- A camada de persistência deve ser implementada **sem a utilização de frameworks mapeadores objeto/relacional** (como JPA, Hibernate, etc) ou qualquer gerador automático de código ;
- Tratamento correto do encapsulamento de exceções entre as camadas;
- O banco de dados deverá ter sido previamente populado (um arquivo contendo os scripts para geração do BD devem ser entregues juntamente com o código fonte) com, no mínimo, os valores necessários para uma boa cobertura de casos de teste.

F) Desenvolvimento, apresentação e avaliação do trabalho:

- O trabalho pode ser realizado individualmente ou em grupos de, no máximo, 3 alunos.
- Os trabalhos serão apresentados no laboratório. Durante a apresentação, TODOS os alunos devem estar presentes e aptos a responder às perguntas. Respostas insatisfatórias por um aluno ou a sua ausência acarretarão descontos na nota final.
- A apresentação do trabalho é de inteira responsabilidade dos alunos (configuração da máquina, do ambiente de software, banco de dados, etc.) e o código-fonte utilizado deverá ser o mesmo entregue ao professor. É tarefa do grupo garantir que o sistema esteja apto a ser executado no dia da apresentação.
- Sistemas que não consigam ser executados ou apresentados no dia da apresentação receberão nota **zero**.
- Mensagens de erro apresentadas durante a execução do programa, mesmo que a aplicação não pare de executar, serão consideradas como erros de execução, e acarretarão descontos na nota do trabalho.
- Em caso de erro de sintaxe (compilação), o peso final do trabalho será valorado em **zero**.
- Em caso de erro de semântica (conteúdo), o peso final do trabalho sofrerá uma redução.
- Os trabalhos serão avaliados de acordo com critérios a serem estabelecidos pelo professor da disciplina, considerando o que é pedido no enunciado e o que foi realizado com sucesso pelo sistema. Também serão avaliadas a modelagem do sistema (correta criação das classes necessárias, com seus atributos e métodos, encapsulamento, e correto estabelecimento de relações entre as classes) e sua implementação de acordo com os conceitos de orientação a objetos e arquitetura multicamada.
- A comprovação do uso de teste unitário, padrões de projeto e de programação por contratos será levada em conta na avaliação do trabalho.
- ***Trabalhos copiados resultarão em nota zero para todos os alunos envolvidos.***

G) Entrega do trabalho:

- Todos os arquivos necessários a execução do sistema, bem como os arquivos-fonte, scripts de banco de dados e os arquivos de documentação, deverão ser empacotados em um único arquivo (.zip) e submetidos através do sistema Moodle até a data de entrega.
- Devem fazer parte da documentação pelo menos:
    - Diagrama de classes do sistema. O diagrama de classes deverá ser entregue junto com documentação em texto salientando os pontos onde foram utilizados padrões de projeto na implementação da solução. É importante que as classes estejam agrupadas em pacotes de acordo com a arquitetura de camadas do sistema.
    - Diagrama relacional da base de dados.
    - Os diagramas devem estar disponíveis em imagens com resolução suficiente e de fácil visualização. Não serão aceitos diagramas que estejam em formato original da ferramenta de desenho (como Visio, Astah, e outros).
    - Casos de teste utilizados no teste unitário (se for o caso).
- Não serão aceitos trabalhos enviados por correio eletrônico nem fora do prazo estabelecido.