Faculdade de Ciências e Tecnologia

Universidade de Coimbra - DEI

IS Project 2 - Java Persistence API(JPA) and Enterprise Java Beans(EJB)

Mestrado em Engenharia Informática

autores: João Batanete 2009113460, Ricardo Rei 2014233736

Introdução

Nesta secção iremos fazer uma breve descrição das tecnologias utilizadas ao longo do projeto.

Java Persistence API(API):

O JMS é uma API de Java que oferece uma camada de abstração durante a criação e interação com bases de dados.

No JPA, as tabelas da base de dados são representadas por entidades(classes com a anotação @Entity) e os atributos da classe representam as colunas da tabela e as relações(*many-to-many*, *many-to-one* ou *one-to-one*) da tabelas com as outras, gerando automaticamente as chaves primárias e forasteiras, e convertendo objetos usuais de java em variáveis SQL(exemplo:Strings em VARCHAR's) de forma automática.

Este modelo de programação simplifica substancialmente a implementação da camada de dados em sistemas.

Enterprise JavaBeans(EJB):

EJB's são uma tecnologia Java que possibilita o encapsulamento da camada de negócio de uma aplicação, através de chamadas remotas ou locais a métodos, simplificando a separação entre a camada de apresentação e de negócio de uma aplicação ou sistema. Os métodos das EJB's funcionam de forma transacional, também ajudando a manter a integridade dos dados.

As EJB's podem ser de sessão(para atender requests de clientes) ou orientadas a mensagens(de forma semelhante ao JMS, utilizado no projeto anterior). Para este projeto foram utilizadas apenas EJB's de sessão.

Por outro lado, as EJB's de sessão podem ser do tipo *stateless* ou *stateful.*

Stateless EJB's não guardam estados nem variáveis entre cada chamada remota de um ou mais clientes, e uma instância da bean pode até atender clientes diferentes, pelo que não deverão ser utilizadas em situações em que tal seja necessário, para evitar corrupção de dados, falhas de segurança e outros problemas possíveis. São no entanto mais ligeiras do que as *stateful,* pelo que deverão ser a opção a utilizar caso não seja necessário manter nenhum estado comunicativo entre cliente e servidor.

Stateful EJB's mantém o referido estado de comunicação entre servidor e cliente entre cada chamada aos métodos, e possibilitam a utilização de "variáveis de sessão". Um *use case* clássico frequentemente referido para as *stateful beans* é a implementação de um carrinho de compras numa loja online, guardando a lista de compras numa variável de sessão. No entanto, este tipo de beans é mais pesado do que as *stateless,* e necessitam de uma instância por cada cliente conectado, pelo que o seu uso extensivo pode levar a problemas de escalabilidade na aplicação.

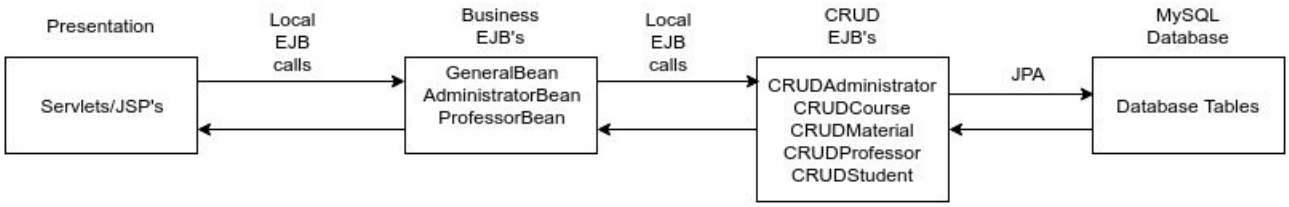
Uma vez que não considerámos necessário guardar variáveis de sessão para nenhuma das funcionalidades das aplicação, foram utilizadas apenas *stateless beans* durante este projeto.

<u>Java Servlets</u>:

Servlets são uma tecnologia Java utilizada geralmente para implementar a interface da camada web(*client-side)* com o servidor de uma aplicação web. Embora se possa implementar uma aplicação web com a camada de negócio implementada por cima das servlets aquando da receção dos requests, uma abordagem do tipo torna-se rapidamente difícil de manter, devido à falta de separação entre as camadas, que leva também a código mais confuso e difícil de manter. Esta abordagem é no entanto utilizada em aplicações pequenas, devido a não possuir o *overhead* extra de implementar a camada de negócio numa componente separada do projeto.

Arquitetura

O sistema encontra-se estruturado da seguinte forma:



A camada de apresentação comunica com a de negócio através de chamadas locais às “Business EJB’s”, que por sua vez estão ligadas a uma segunda camada de EJB’s que realiza interação com a base de dados recorrendo a entidades JPA.

Nas secções seguintes iremos descrever o funcionamento de cada componente com um nível maior de detalhe.

Componentes do sistema

Nesta secção será descrito o funcionamento e as decisões tomadas ao longo da implementação das várias componentes do projeto.

Mini-programa para gerir admins(ISproj2-manageadmins):

Este programa destina-se apenas a ser utilizado para criar ou remover admins do sistema. De notar que para ser utilizado o Wildfly tem de estar ligado com o EAR fornecido exportado na pasta, uma vez que necessita de aceder a uma bean remota para funcionar.

O programa deve ser executado na linha de comandos e recebe a operação pretendida como argumento.

Para criar um Administrador deve ser utilizado o seguinte comando(Linux):

java -jar ISproj2-manageadmins.jar -c

Em seguida, deve-se preencher os campos conforme desejado.

Para apagar um administrador, deverá ser utilizado o comando:

java -jar ISproj2-manageadmins.jar -d

Neste caso apenas será necessário fornecer o email da instituição/login do administrador.

Camada de apresentação(ISproj2-WEB):

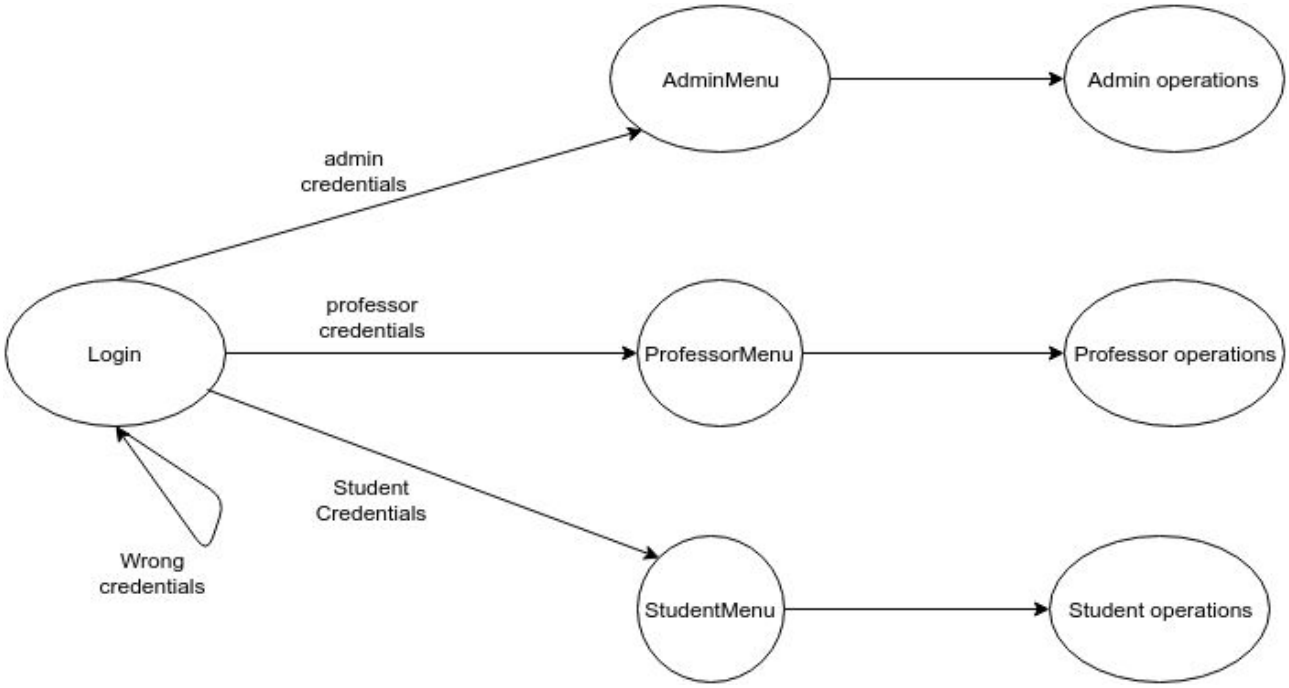
Na camada de apresentação são utilizadas Servlets(lado do servidor) e ficheiros JSP(lado do cliente).

Para a manutenção de “sessões” após o login, e para transportar dados persistentes entre as servlets(como o utilizador e a *password* do utilizador), foram utilizadas *cookies*.

Quando o utilizador procura aceder a um conteúdo ao qual não tem acesso(exemplo: um estudante tenta ligar-se ao menu do admin), é automaticamente redirecionado para a página de login. Esta verificação é feita obtendo as credenciais presentes nas *cookies* e chamando o método *login* da GeneralBean, obtendo assim o tipo de conta(admin, aluno, professor ou inexistente) correspondente às credenciais.

As páginas em si foram criadas a partir de templates do site *bootstrap*, por questões de facilidade de implementação e da inexperiência do grupo a desenvolver interfaces gráficas. Uma vez que esta era uma componente secundária no projeto, considerámos esta abordagem suficiente.

Em baixo encontra-se um esquema básico de como as páginas web estão organizadas.



Camada de negócio(ISproj2-EJB):

A camada de negócio é constituída por EJB's acedidos localmente pelas Servlets da camada de apresentação.

Em termos de organização das beans, optámos por criar uma utilizada para as funcionalidades do professor(ProfessorBean), outra para as do admin(AdministratorBean) e outra para as funcionalidades gerais(GeneralBean).

As beans desta camada não interagem diretamente com a base de dados, realizando chamadas às beans CRUD para este fim.

Como já foi referido, todas as beans implementadas são do tipo *stateless,* devido a não termos considerado necessário manter um estado comunicativo entre servidor e cliente em nenhuma das operações implementadas.

Em seguida iremos detalhar as funcionalidades de cada bean presente nesta camada.

De notar que uma descrição mais detalhada de cada método pode ser encontrada no código fonte, em comentário.

GeneralBean:

Esta bean implementa as operações que todos os utilizadores podem realizar na aplicação. Estas são:

- efetuar login(verificação de credenciais)
- obter o nome e o conteúdo de materiais guardados no sistema

Uma vez que os estudantes apenas podem listar e fazer o download de materiais, apenas utilizam esta bean.

AdministratorBean:

Esta bean implementa as operações que um admin pode realizar na aplicação. Estas são:

- criar um novo professor ou aluno, ou apagar um
- mudar um campo de informação de um professor ou aluno
- criar um novo curso, com ou sem uma lista de alunos e um professor, ou apagar um
- adicionar ou remover um aluno de um curso
- mudar o professor associado a um curso
- apagar um material do sistema

ProfessorBean:

Esta bean implementa as operações que um admin pode realizar na aplicação. Estas são:

- adicionar um material a um curso
- obter os estudantes associados a um curso lecionado pelo professor
- apagar materiais associados a cursos lecionados pelo professor
- realizar uma pesquisa por alunos de acordo com os seus dados(nome,email,...)

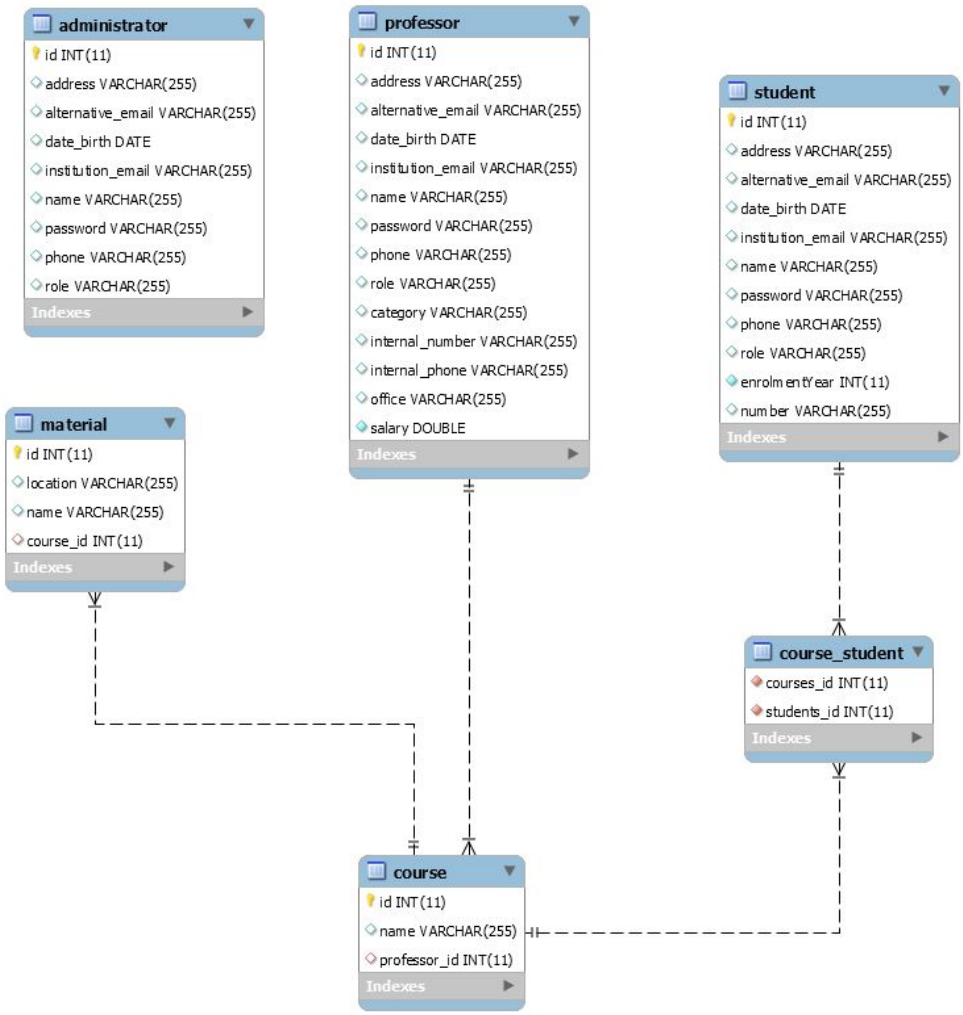
Nota1: A classe CreateAdminBean presente no projeto é apenas utilizada pelo programa utilizado para criar e apagar admins, não tendo qualquer utilidade nas outras funcionalidades do sistema.

Nota2: Na classe ProfessorBean, os campos MATERIALFOLDERWINDOWS e MATERIALFOLDERLINUX devem ser alterados consoante o sistema operativo utilizado e a diretoria pretendida para guardar os ficheiros. Caso o OS não seja nem Windows nem Linux, o campo utilizado é o MATERIALFOLDERWINDOWS.

Camada de dados(ISproj2-CRUD e ISproj2-JPA):

Esta camada é constituída por uma segunda camada de EJB's que se destinam apenas a realizar operações CRUD(create, read, update e delete) na base de dados, e pelas próprias entidades JPA utilizadas para representar e interagir com esta.

Em seguida encontra-se um diagrama ER das entidades/tabelas presente na base de dados.



As EJB's CRUD possuem métodos que acedem às entidades JPA e realizam as operações de escrita ou leitura pretendidas. Uma informação mais detalhada das funcionalidades de cada método está presente em comentário no código da interface das EJB's.

Opções importantes tomadas aquando da implementação da camada de dados:

- Os três tipos de utilizador(admin, estudante e professor) são entidades que derivam a *mappedsuperclass* Person, uma vez que a maioria dos seus campos são iguais, o que facilitou a implementação. De notar que não existe uma tabela Person na base de dados, uma vez que as *mappedsuperclasses* atuam apenas como template para as entidades que derivam delas.
- A relação *one-to-many* de cursos para professores é bidirecional, com o curso a ser o dono da relação. Assim, a base de dados não necessita de nenhuma tabela auxiliar para manter a relação. A razão de ser bidirecional, é o facto de precisarmos de uma referência para os cursos a partir do professor aquando do apagamento deste, para podermos retirar a sua referência da sua lista de cursos.

- A relação *many-to-many* dos cursos para os estudantes é também bidirecional. pelo mesmo motivo(apagamento). Os cursos são os donos da relação, pelo que possuímos a tabela auxiliar Course_Student em lugar de Student_Course.

Encriptação de passwords:

Para a encriptação de passwords na base de dados, foi utilizado o algoritmo SHA-1. Embora este algoritmo já se encontre algo ultrapassado quando comparado a alternativas como MD5 e SHA-256, sendo este um projeto de natureza académica o grupo considerou-o suficiente.

Testes realizados

Nesta secção iremos descrever os principais testes realizados ao funcionamento da aplicação antes da sua submissão.

Operação/requerimento	**Testes realizados e output esperado**
1-script para criar/apagar admins	● criar admin que não existe->admin deve ser criado ● apagar admin que existe->admin deve ser apagado ● tentar criar admin que já existe->erro ● tentar apagar admin que não existe->erro ● tentar logar como admin inexistente ou previamente apagado->erro
2-como admin, criar novos alunos e professores	● criar um estudante ou professor que não existe->utilizador deve ser criado ● apagar estudante ou professor que existe-> deve ser apagado ● tentar criar estudante ou professor que já existe->erro ● tentar apagar estudante ou professor que não existe->erro ● tentar logar como estudante ou professor inexistente ou previamente apagado->erro
3-como admin, editar campos de informação de um utilizador	● editar campo existente e comum aos 3 tipos de Person, quando a pessoa existe->campo editado com sucesso

	● editar campo que só existe num tipo de Person, numa person desse tipo->campo editado com sucesso ● editar campo que só existe num tipo de Person, numa person diferente desse tipo->erro ● editar qualquer campo numa Person que não existe->erro ● editar campo que não existe->erro
4-como utilizador não autenticado, apenas ter acesso à página de login	● procurar entrar em qualquer página da app sem estar logado->erro ● procurar entrar em qualquer página da app para que não temos permissão(exemplo:entrar como prof no menu do admin)->erro ● entrar numa página para a qual temos permissão->sucesso
5-realizar o login e com o email da instituição e a password	● user e pass corretas->aceder a página que corresponde ao tipo de utilizador logado ● user e pass errados->refresh na página de login
6-realizar logout a partir de qualquer localização	● fazer logout->cookies apagadas e página de login mostrada
7-como admin, criar um curso com uma lista de alunos e professor, e mudar qualquer informação presente nele	● criar curso(curso não existe, professor e alunos existem)->sucesso ● criar curso(curso existe)->erro ● criar curso(professor não existe)->erro ● criar curso(pelo menos um aluno não existe)->erro ● adicionar student a curso(student existe, não está no curso)->sucesso ● adicionar student a curso(student não existe)->erro ● adicionar student a curso(student existe, já está no curso)->sucesso ● renomear curso(curso existe, não existe curso com o novo nome)->sucesso ● renomear curso(curso não existe)->erro ● renomear curso(curso existe, curso com o novo nome existe)->erro
8-como professor, fazer upload a materiais de um curso lecionado por mim	● upload de um material(curso existe,material não existe no curso)->sucesso ● upload de um material(curso existe,material existe no curso)->erro ● upload de um material(curso não existe)->erro ● upload de um material(professor não leciona curso)->erro
9-como professor, remover materiais de um curso	● apagar um material(curso existe,material

lecionado por mim	não existe no curso)->sucesso ● apagar um material(curso existe,material existe no curso)->erro ● apagar um material(curso não existe)->erro ● apagar um material(professor não leciona curso)->erro
10-como qualquer utilizador logado, listar e fazer o download de materiais de qualquer curso	(materiais foram listados e descarregados sem problemas)
11-como professor, listar os alunos de um determinado curso lecionado por mim ordenados por ordem crescente ou decrescente	● professor leciona curso->lista corretamente ordenada ● professor não leciona curso->erro
12-pesquisar por alunos segundo uma lista de critérios	● sempre sucesso(caso não exista nenhum aluno é mostrada uma lista vazia)
13-como administrador, apagar qualquer tipo de dados do sistema	● apagar qualquer entidade inexistente->erro ● apagar professor->todos os cursos lecionados por ele metem o campo professor_id a NULL. ● apagar administrador->sucesso ● apagar aluno->removido da lista de estudantes de todos os cursos que estudava ● apagar curso->removido de todas as listas de cursos dos seus estudantes e professor, todos os seus materiais são apagados ● apagar material->removido da lista de materiais do seu curso

Referências

- slides da disciplina
- http://eai-course.blogspot.pt (blog do professor)
- https://en.wikipedia.org/wiki/Enterprise_JavaBeans
- https://en.wikipedia.org/wiki/Java_Persistence_API
- http://getbootstrap.com/
- https://www.draw.io/