Inatel

DM110 – Trabalho Final da Disciplina

Alunos:

Felipe Gustavo de Freitas Rocha

Higor Augusto Silvério

Monique Gabriela de Souza

Professor:

Roberto Ribeiro Rocha

Objetivo

Este documento tem como objetivo mostrar a modelagem do serviço de gerenciamento de parceiros (entidade *Partner*) da aplicação proposta como trabalho final da disciplina DM110 – Desenvolvimento Java EE, da Pós-Graduação em Desenvolvimento de Aplicações para Dispositivos Móveis e *Cloud Computing*.

Premissas

Como diretivas para a execução e modelagem dos serviços, foram tomados como dados as especificações passadas em sala de aula. Destacam-se as seguintes premissas:

- Desenvolvimento de serviços, seguindo o padrão REST (*REpresentational State Transfer*), com *session beans stateless* e entidades JPA (Java *Persistente API - Application Programming Interface*);
- Serviços devem ser destinados a suportar as operações:
    - inclusão de novo parceiro;
    - busca de parceiro através de seu identificador (*partner code*);
    - listagem de parceiros; e
    - atualização do registro de um parceiro.
- Os serviços devem acessar o banco de dados da aplicação através dos *session beans*;
- O *session bean* deve chamar um serviço de mensageria (MDB - *Message-Driven Beans*);
- Deve ser feito registro de auditoria das ações executadas dentro da API;
- O serviço de mensageria deve obrigatoriamente chamar um *session bean* para acessar o banco de dados;
- A entidade *Partner* possui os seguintes atributos:
    - *partner code*;
    - *name*;
    - *phone*;
    - *email*; e
    - *rating*.
- A entidade de auditoria, utilizada para registrar as ações dentro da API, possui como atributos:
    - *identifier* (id);
    - *partner code*;
    - *operation*; e
    - *timestamp*.

Estrutura

A organização do projeto, que utiliza o Maven como gerenciador de dependências, tendo como empacotador um módulo EAR (*Enterprise Application aRchive*), pode ser visualizado da seguinte forma:

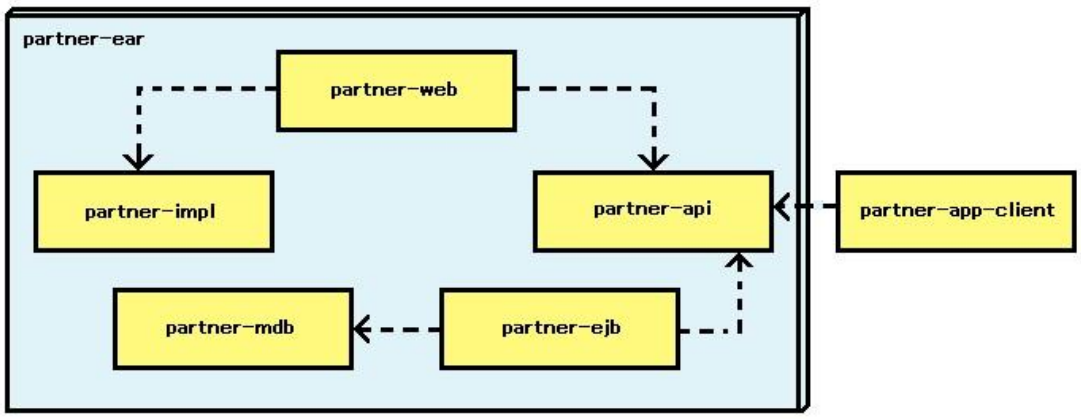

Figura 1 - Estrutura geral do projeto

A arquitetura das requisições REST, quando utilizadas pelo cliente, nas operações descritas anteriormente neste documento, pode ser exemplificada na imagem seguinte:

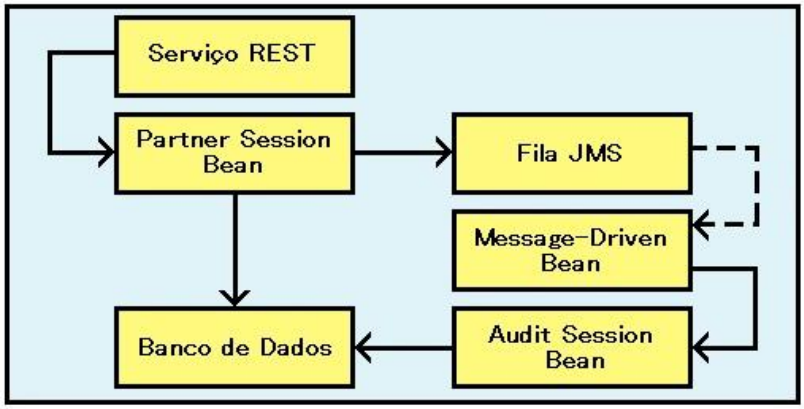

Figura 2 - Arquitetura geral do projeto

Diagrama de classes

A visualização do diagrama de classes a aplicação completa, pode ser vista na figura abaixo:

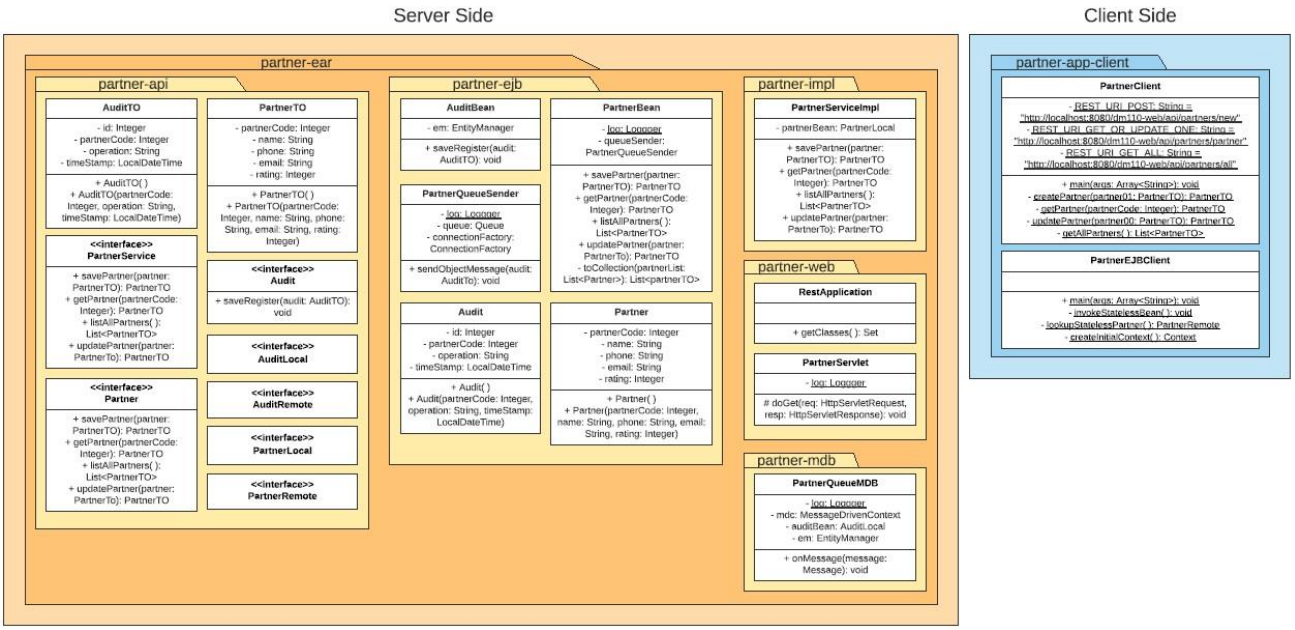

Figura 3 - Visão geral do diagrama de classes da modelagem do serviço

Este diagrama também poder ser visto, em sua versão PDF, na URL abaixo:

- https://github.com/higorasilverio/final-work-dm110/blob/master/docs/modeling-assets/uml-classes.pdf

De modo geral, assim como mostrado na Figura 1, este foi dividido em duas partes: *Client Side* (módulo *partner-app-client*) e *Server Side* (módulos empacotados dentro do *partner-ear*).

O *Client Side*, por se tratar apenas de um módulo, pode ser facilmente visualizado:

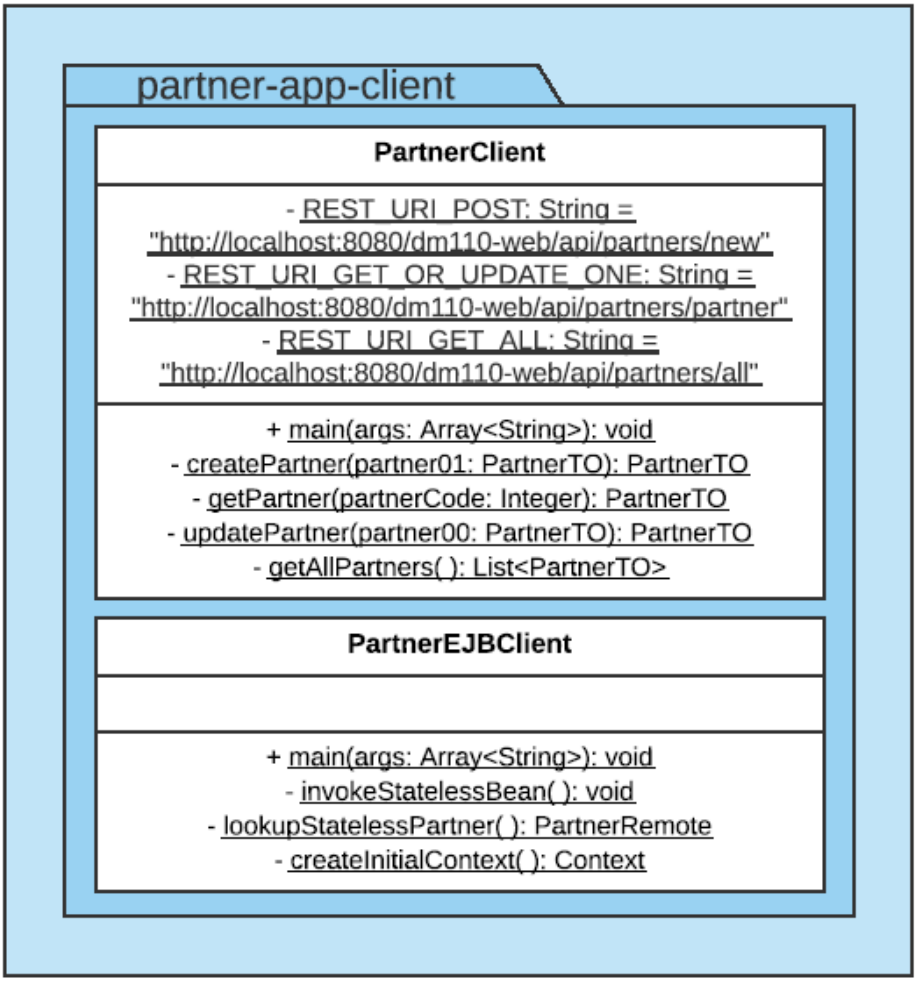

Figura 4 - *Client Side*: partner-app-client – Implementação do acesso à aplicação pelo lado do cliente

Para melhor visualização, o *Server Side* pode ser visualizado através de seus módulos, todos empacotados dentro do módulo EAR, de forma separada:

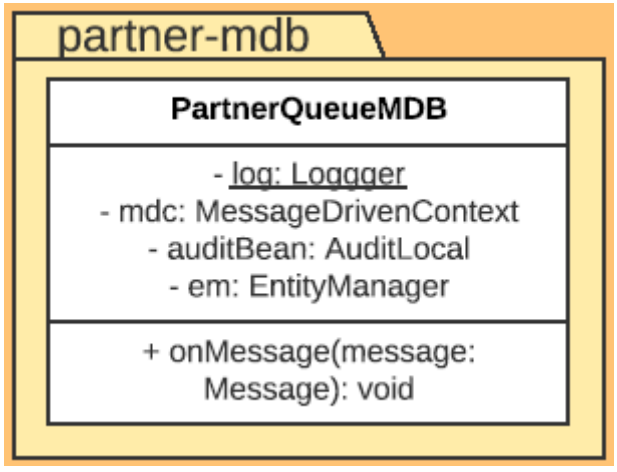

Figura 5 - *Server Side*: *partner-mdb* – Serviço de mensageria da aplicação

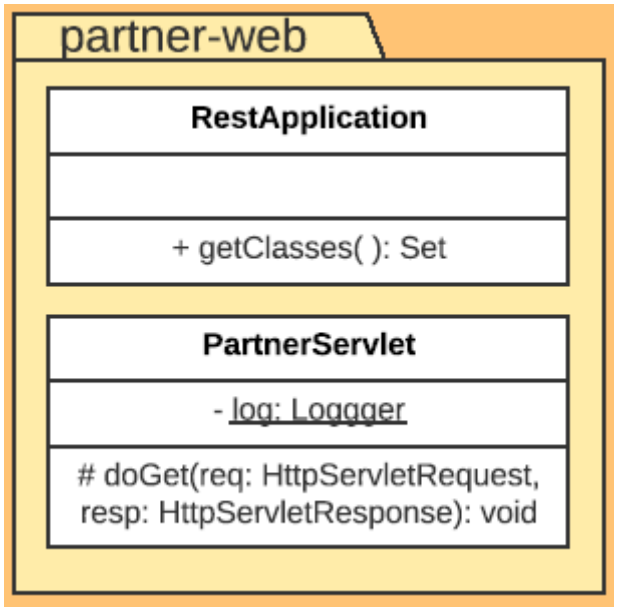

Figura 6 - *Server Side*: *partner-web* – Exposições dos serviços via aplicação *Web*

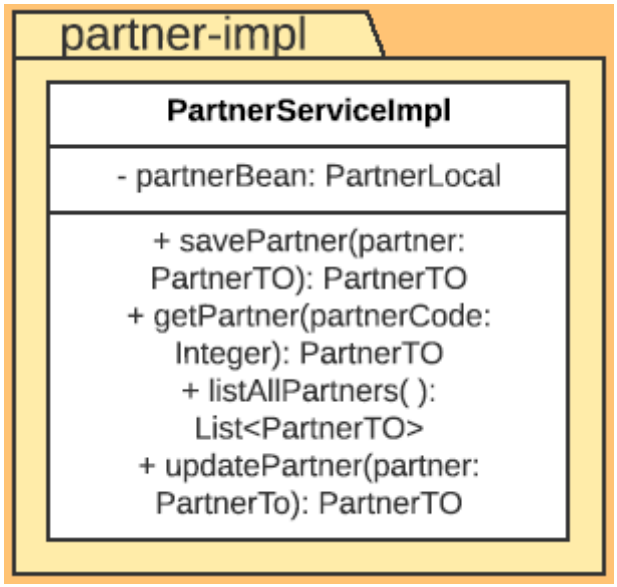

Figura 7 - *Server Side*: *partner-impl* – Implementação do serviço de gerenciamento de parceiros

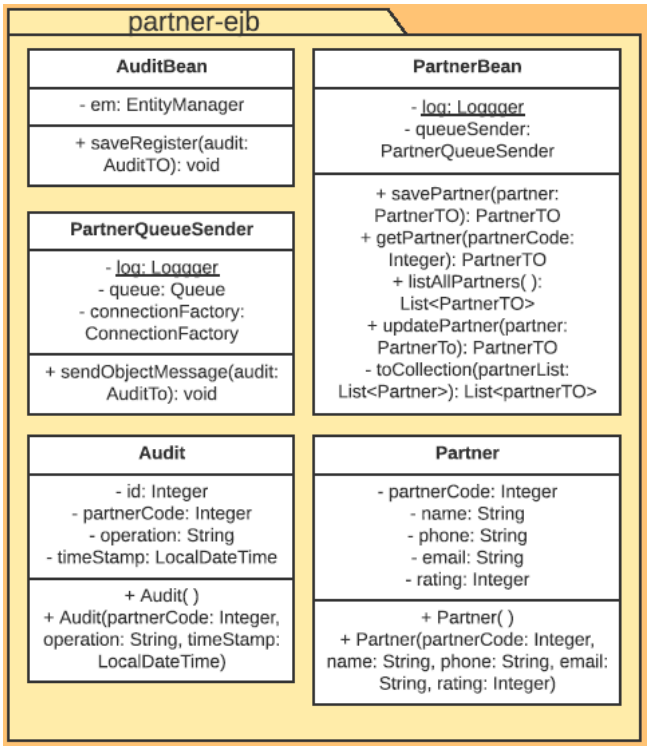

Figura 8 - *Server Side*: *partner-ejb* – Java *beans* e entidades de persistência de parceiros e auditoria do serviço

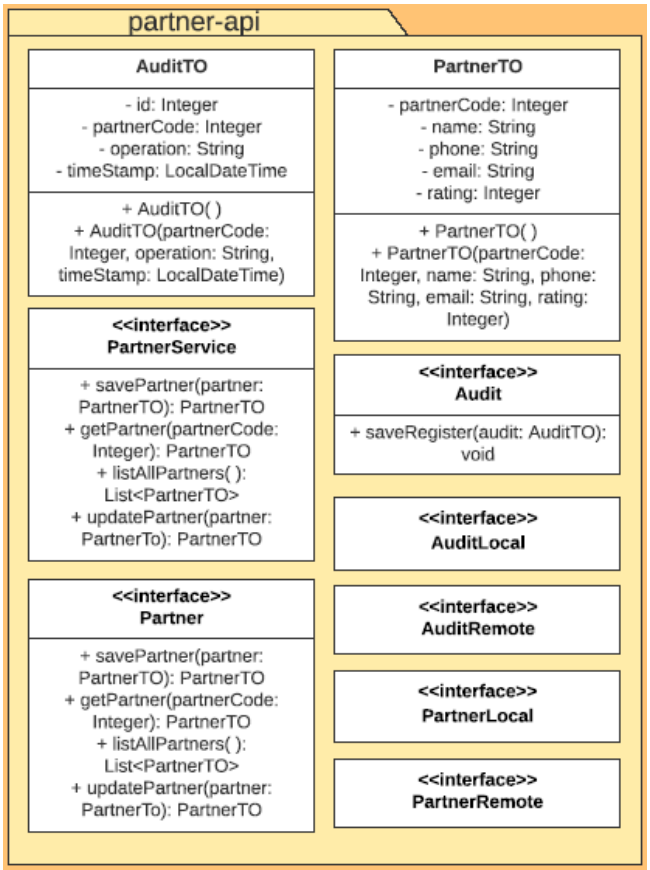

Figura 9 - *Server Side*: *partner-api* – Objetos de transferência intermediários e definição das interfaces de serviços

Conclusão

Este documento mostra como foi pensado e implementado o serviço de gerenciamento e acesso aos parceiros da aplicação proposta nas aulas de DM110 – Desenvolvimento JavaEE.

Para isto, é mostrada as premissas adotadas na concepção do serviço, a arquitetura e estrutura básica deste, assim como feito em sala de aula, e, por fim, uma descrição, por meio de um diagrama UML simplificado, de como os módulos Java estão implementados.

Este serviço tem como intuito descrever uma aplicação escalável e com baixo acoplamento, visando atender a uma necessidade empresarial real.