CONSTRUÇÃO DE SISTEMAS DE SOFTWARE

Relatório Entrega 2 Engenharia Informática – 2021/2022

Grupo 41

Nuno Fontes | nº 46413

Ricardo Alves | nº 51742

Decisões Importantes no Desenho da Aplicação

Camada de Negócio

Para desenvolver a segunda fase do projeto, começamos por adaptar a nossa lógica de negócio à nova versão da aplicação. Esta nova versão do projeto SaudeGes possui dois projetos cliente para executar os casos de uso desenvolvidos nas metas anteriores (GUI e Web), por isso, com o intuito de transportar informação entre estes foram definidas interfaces remotas definitas com a anotação @Remote. Estas, que se inserem no package facade, possuem a assinatura dos métodos dos diferentes casos de uso e, como tal servem para os projetos cliente os poderem executar sem que a sua implementação seja exposta.

Os services para esta meta foram adaptados para serem session beans. Todos eles não guardam estado, logo são session bean stateless e possuem a anotação @Stateless. Estes, para executar os métodos dos diferentes handlers, necessitam que as dependências destes últimos sejam injetadas através da anotação @EJB nos atributos de cada service. Se um atributo de uma classe e essa mesma estiverem em diferentes camadas ou subcamadas (neste último caso as classes service do package facade e os seus atributos handlers do package business), então terão de ser anotados com as anotações @EJB e @Stateless ou @Stateful, respectivamente.

Seguindo esta linha de pensamento, para cada classe handler foi-lhe anotada um session bean. Estes Handlers foram alterados para serem usados de maneira correta para este projeto.

Como a comunicação entre a base de dados já não é dependente de EntityManagers criados em cada classe handler, estas transações foram substituídas nas classes que representam os catálogos (session beans stateless) com uma única instância de um objeto EntityManager devido à anotação @PersistenceContext. Cada método que persiste um objeto na base de dados é anotado com @Transactional(Transactional.TxType.REQUIRES_NEW) se o objeto era novo na base de dados ou apenas @Transactional.

De modo a transferir a informação necessária para mostrar ao utilizador e proteger a integridade dos dados enviados remotamente, foi utilizado o padrão Data Transfer Object.

Cliente Web

No desenvolvimento do cliente web utilizamos as interfaces anotadas com @Remote para efetuar os casos de uso Comprar Participação Mensal em Atividade Regular e Agendar Atividade Ocasional. De maneira a dar utilidade a essas interfaces foi utilizado o padrão Front Controller em que são utilizadas diferentes classes Action.

A correspondência entre os endereços web e as classes Action é mantida no ficheiro app.properties que contem os URL's seguintes:

appRoot/action/atividade/novaAtividade = java:module/NewActivityAction

appRoot/action/atividade/criarAtividade = java:module/CreateActivityAction

appRoot/action/atividade/definirHorarioSelecionarAtividade = java:module/NewDefineScheduleAction

appRoot/action/atividade/definirHorario = java:module/DefineScheduleSelectActivityAction

appRoot/action/vendas/comprarParticipacaoMensalSelecionarAtividade = java:module/BuyMonthlySubscriptionAction

appRoot/action/vendas/comprarParticipacaoMensal = java:module/BuyMonthlyChooseActivityAction

appRoot/action/vendas/confirmarCompraParticipacaoMensal = java:module/ConfirmBuyMonthlyAction

appRoot/action/vendas/agendarAtividadeOcasional = java:module/NewBookOcasionalActivity

appRoot/action/vendas/agendarAtividadeOcasionalSelecionarAtividade = java:module/BookOccasionalActivity

appRoot/action/vendas/agendarAtividadeOcasionalSelecionarInstrutor = java:module/BookOccasionalActivitySelectInstructorAction

appRoot/action/vendas/confirmarAgendarAtividadeOcasional = java:module/ConfirmBookOccasionalActivityAction

A visualização é mantida com ficheiros JSP que utilizam as diferentes Actions para executar as várias funcionalidades da aplicação.

Cliente GUI

No projeto de cliente GUI foram criados três ecrãs: menu principal, Criar Atividade e Definir Novo Horário de Atividade Regular. O menu principal serve apenas para o utilizador escolher qual o caso de uso que quer executar. Cada ecrã responsável por um caso de uso e utiliza os session beans remotos para executar as suas funcionalidades através das classes controller, createActivityController e defineScheduleController. Para estabelecer conexão entre o controler e a view (.fxml) foram implementadas as classes Model para cada objecto.