Design patterns

SEM Group 13A

Strategy pattern

Our software application should be compatible with different client applications. It was decided to use a strategy design pattern to accept requests through User microservice API in form of different semi-structured data formats. Therefore, we are using 2 different strategies to receive requests. This API supports now both JSON and XML formatted requests. We had the plain text strategy before as well, but it was decided to remove this strategy during the final discussion because plain text is practically not used for object transfer nowadays.

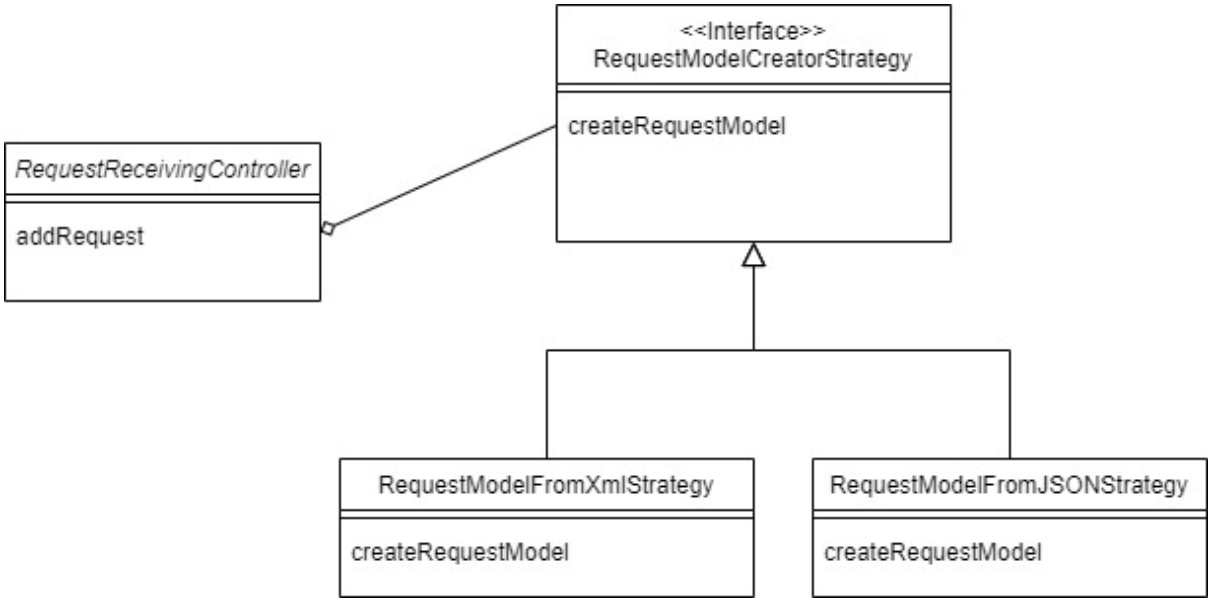
We are using a Strategy design pattern with one common interface and 2 different strategies to receive requests. A client can choose what strategy will be used by specifying the headers of HTTP requests. The controller will create a specific strategy class based on it.

All created classes are in

user-microservice -> src -> requestmodelget

and API that is using them is defined in

user-microservice -> src -> controllers -> RequestReceivingController.class -> addRequest()



Adapter pattern

It was a problem, that 2 different microservices were using different data transfer objects to communicate one with another. Therefore, we decided to use an adapter design pattern to make one microservice use the output of another microservice. We have used an adapter pattern for accepting requests in FacultyAdminController, which should communicate with WaitingListController. We are using AcceptRequest interface and Adapter class that inherits from it and takes WaitingListInterface as the parameter.

All created classes are in

user-microservice -> src -> acceptrequestadapter

and API that is using them is defined in

user-microservice -> src -> controllers -> FacultyAdminController.class -> acceptRequest()