# 1 Introdução

A organização de eventos e atividades desportivas está a afirmar-se como uma atividade importante e com potencial lucrativo considerável. No entanto, a organização destes eventos e atividades tem diversos intervenientes com diferentes interesses e apresenta desafios complexos, não só na definição e gestão dos mesmos, como na comunicação com as diversas partes interessadas. Embora existam algumas ofertas no mercado de aplicações para apoiar a organização de diferentes tipos de eventos e atividades desportivas, estas dão apoio apenas a aspetos específicos como, por exemplo, a venda de bilhetes. Assim, aproveitando esta necessidade de mercado, CSS lançou-se na construção de uma aplicação que apoie a organização de eventos e atividades desportivas nas suas diversas vertentes.

Nos três últimos anos letivos, os alunos de CSS desenvolveram uma primeira iteração de três módulos[1] de um software de gestão de eventos e atividades desportivas, a saber:

- *DesporGes*, um módulo centrado apenas no apoio à gestão de provas desportivas de tipo taça e campeonato.
- *TornGes*, um módulo centrado na gestão de torneios que assentam na classificação de jogadores com o sistema de Elo[2].
- *EspacoGes,* um módulo centrado na gestão de utilização de espaços para atividades desportivas.

Este ano pretende-se desenvolver um módulo que apoie a gestão de aulas.

# 2 *AulasGes*

Na primeira iteração, o módulo *AulasGes* vai centrar-se no apoio à criação de aulas de diferentes modalidades e à ativação destas aulas (realizadas pelos gestores das instalações desportivas) e à inscrição dos utentes nas aulas ativas.

As instalações desportivas geridas pelo sistema permitem a realização de atividades de uma ou mais modalidades. O sistema irá permitir que os gestores realizem a criação de aulas regulares para estas modalidades, as quais ocorrem em horário e dias da semana fixos. Posteriormente, o sistema permite ativar uma aula durante um período de tempo (por exemplo, um ano letivo) e simultaneamente definir a instalação em que as sessões dessa aula vão decorrer. O sistema permite ainda que os utentes se inscrevam nas aulas ativas, sendo suportadas duas modalidades de inscrições:

- inscrições regulares, que podem ser realizadas em qualquer altura, e dão acesso a todas as futuras sessões da aula (a menos que a inscrição seja cancelada por falta de pagamento da mensalidade)
- inscrições avulso, que só podem ser realizadas nas 24 horas anteriores a uma sessão da aula e dão acesso apenas a essa sessão

[1] Pode ler sobre esta forma de modularização centrada no domínio em https://martinfowler.com/bliki/PresentationDomainDataLayering.html

[2] Sistema de Rating ELO
https://pt.wikipedia.org/wiki/Rating_ELO

Especificamente, vão ser considerados 4 casos de uso cuja descrição breve é fornecida abaixo.

***1) Criar Aula*** — O gestor indica que quer criar uma aula de uma modalidade desportiva. O sistema indica as modalidades existentes e pede ao gestor a seguinte informação: a modalidade, o nome da aula, os dias-da-semana em que a aula decorre, a hora de início e a sua duração (em minutos). O sistema verifica que

- a modalidade indicada é válida,
- o nome tem 6 caracteres, pelo menos 3 alfanuméricos, não tem espaços, e é único,
- a duração é positiva e não é inferior ao mínimo definido para as aulas da modalidade indicada

e cria uma nova aula, no estado inativo, com os dados fornecidos.

***2) Ativar Aula*** — O gestor indica que quer ativar uma aula. O sistema indica as instalações e pede ao gestor para indicar a designação da aula, a instalação a atribuir, duas datas que definem o período em que a aula vai estar ativa e o número máximo de alunos em cada aula. O sistema verifica que

- a aula indicada é válida e não está ativa
- o par de datas define um período no futuro
- a instalação indicada é válida
- a instalação permite a realização de atividades da modalidade da aula
- a instalação está livre no horário/dias-da-semana da aula, durante todo o período em que a aula estiver ativa
- o número máximo de participantes é inferior à capacidade da instalação indicada

e ativa a aula, registando os dados fornecidos. O sistema deve criar todas as sessões da aula no período em que a aula está ativa e registar a ocupação da instalação com essas sessões.

***3) Inscrever em Aula*** — O utente indica que se quer inscrever numa aula. O sistema indica as modalidades existentes e pede ao utente para indicar uma modalidade e se pretende fazer uma inscrição regular ou avulso. Se a modalidade e o tipo de inscrição fornecidos forem válidos, o sistema mostra as aulas ativas para essa modalidade na qual o utente se pode inscrever, ordenadas por hora de início (no caso das inscrições avulso, só são incluídas aulas ativas que tenham uma sessão nas próximas 24h e uma vaga nessa sessão). Mais especificamente, o sistema mostra, para cada aula, o nome, o horário/dias-da-semana, o nome da instalação e o número de vagas existente. O utente escolhe a aula e indica o seu número de utente. O sistema verifica que o número do utente e a designação da aula são válidos e que a inscrição ainda é efetivamente possível nessa aula. O sistema regista a inscrição do utente, o que inclui o seu custo. No caso de ser uma inscrição avulso, o utente fica com uma vaga da próxima sessão da aula e o custo é *preço por hora * duração da aula/60*. No caso de ser uma inscrição regular, o utente fica com uma vaga em todas as sessões futuras da aula e o custo é a mensalidade a pagar, calculada com *preço por hora * duração da aula/60 * numero de sessões por semana * 4*.

***4) Visualizar Ocupação de Instalação*** — O gestor de espaços indica que quer visualizar a ocupação de uma instalação desportiva fornecendo o seu nome e uma data. O sistema valida que o nome da instalação é válido e mostra a ocupação dessa instalação. As entradas da lista devem conter o nome da aula e o período em que vai decorrer e estar ordenadas por hora.

Nota: Nesta primeira iteração não se vão desenvolver os casos de uso para criar modalidades, instalações e utentes. Tal ficará para uma iteração futura. Parta do princípio que existe um leque fixo de tipos de instalação (*Sala de Bicicletas, Estúdio, Piscina, Campo de Ténis)* e que todas as instalações, modalidades e utentes já foram de alguma forma introduzidos.

As instalações têm um nome (que as identifica univocamente), uma lotação máxima e são de um determinado tipo. Cada instalação permite a realização de atividades de uma ou mais modalidades. As modalidades são identificadas univocamente pelo seu nome. Existe uma duração mínima para as aulas de cada modalidade (em minutos) e um custo por hora. Os utentes têm um número de inscrição no sistema, um nome e um nif.

# 3 Que devemos fazer?

Neste trabalho prático e no seguinte pretende-se que desenvolvam os 4 casos de uso da aplicação *AulasGes*, ainda sem se preocuparem como será feita a apresentação. O código da aplicação deve ser organizado em camadas, incluindo uma camada com a lógica de domínio e uma camada de serviços.

Neste trabalho pretende-se que se foquem na concepção e desenvolvimento da camada com a lógica de domínio e da camada de serviços, sem se preocuparem ainda com a persistência e acesso aos dados e com a apresentação.[3] A camada com a lógica de domínio deve seguir o padrão ***Domain Model*** e a camada de serviços terá simplesmente o propósito de expor as funcionalidades da aplicação (API) escondendo a organização da camada de negócio.

Em concreto, devem:

- elaborar os *system sequence diagram* (SSD) e o modelo de domínio de forma a identificar as operações do sistema, os conceitos relevantes e as associações entre estes;
- produzir um modelo de classes que mostre a organização da camada de serviços e a organização da camada com a lógica de domínio (a qual deve seguir o padrão *Domain Model*);
- proceder a uma implementação deste modelo com a implementação total de todas as classes menos dos catálogos; nos catálogos só precisam de definir os métodos necessários no restante código, podendo deixar a sua implementação por fazer[4]

Devem usar o plugin ***SonarLint*** para o Eclipse para vos ajudar a controlar a qualidade do vosso código e o sistema de controlo de versões ***Git***, disponível no servidor git.alunos.di.fc.ul.pt, para vos ajudar no desenvolvimento cooperativo.

Um elemento do grupo de trabalho deverá seguir as instruções para fazer *fork* do repositório em https://git.alunos.di.fc.ul.pt/css000/css_meta3. **Importante:** devem definir o vosso *fork* como sendo privado (outros alunos não o poderão ver) e adicionar à lista de membros do projeto o utilizador css000 com o nível de *Reporter*.

O repositório fornecido contém um projeto Eclipse com o código de uma versão do sistema *SaleSys* com características semelhantes às que se pretende nesta entrega, pelo que deve começar por o analisar. No ficheiro README encontra uma descrição dos aspetos mais importantes da solução de desenho implementada.

[3] Na entrega 4 irão acrescentar a camada de acesso aos dados. A camada de apresentação será endereçada na entrega 5.

[4] Mas aconselhamos que incluam o mínimo que vos permita testar o resto do código, para atacarem o trabalho para a próxima entrega com o código já depurado.

## 4 Como e quando entregamos?

Identifiquem o *commit* como sendo a entrega 3 (`git tag entrega3`) e coloquem essa identificação no servidor gitlab (`git push origin entrega3`). Consideramos apenas este *tag*, se o *commit* tiver uma data anterior a **8 de Abril de 2020**, a *deadline* para entrega.

O repositório deve conter:

1. um único documento PDF com os SSDs e o modelo de classes da vossa solução
2. o código fonte da vossa solução