**2ª entrega**

**Entrega intermédia**

Integração de Sistemas de Informação

**Aluno/os:**

21140 - Pedro Vieira Simões

21145 – Gonçalo Moreira da Cunha

21152 – João Carlos da Costa Apresentação

**Professor/es: Óscar Ribeiro**

**Licenciatura em Engenharia de Sistemas Informáticos**

Barcelos, dezembro de 2022

IPCA GYM

# Resumo

Este trabalho prático, relativo à unidade curricular de **Integração de sistemas de informação,** propende a melhorar a performance de trabalho em equipa num desafio que irá explorar as necessidades de um smart campus, no IPCA e demonstrar técnicas e conceitos abordados inter e extracurricular.

A ideia principal do projeto será um sistema para um ginásio e uma aplicação para os utilizadores. Nesta unidade curricular em específico iremos abordar a construção e interação com a API que irá sustentar o nosso projeto.

Conteúdo

[Resumo 4](#_Toc122563046)

[1. Introdução 7](#_Toc122563047)

[1.1. Contextualização 7](#_Toc122563048)

[1.2. Motivação e Objetivos 7](#_Toc122563049)

[1.3. Estrutura do Documento 7](#_Toc122563050)

[2. Produto 8](#_Toc122563051)

[2.1. Visão do Produto 8](#_Toc122563052)

[3. Organização do Grupo 9](#_Toc122563053)

[4. Diagramas 10](#_Toc122563054)

[4.1. Diagrama Entidade-Relação 10](#_Toc122563055)

[5. Código 12](#_Toc122563056)

[5.1. Programação por Camadas 12](#_Toc122563057)

[13.1.1 Backend\_IPCA\_Gym 12](#_Toc122563058)

[13](#_Toc122563059)

[13](#_Toc122563060)

[13](#_Toc122563061)

[13](#_Toc122563062)

[13](#_Toc122563063)

[13.1.2 LayerBLL 14](#_Toc122563064)

[15](#_Toc122563065)

[13.1.3 LayerBOL 16](#_Toc122563066)

[13.1.3 LayerDAL 17](#_Toc122563067)

[6. Conclusão 19](#_Toc122563068)

[7. Bibliografia 19](#_Toc122563069)

[Figura 1 - Diagrama de Entidade-Relação 10](#_Toc122563070)

[Figura 2 - Camadas 12](file:///C:\Users\Gonçalo%20Cunha\Desktop\Universidade\ipca_gym\ipca_gym\Entrega_Intermedia_21140_21145_21152.docx#_Toc122563071)

[Figura 3 - Lista dos controladores 12](file:///C:\Users\Gonçalo%20Cunha\Desktop\Universidade\ipca_gym\ipca_gym\Entrega_Intermedia_21140_21145_21152.docx#_Toc122563072)

[Figura 4 - Main 12](file:///C:\Users\Gonçalo%20Cunha\Desktop\Universidade\ipca_gym\ipca_gym\Entrega_Intermedia_21140_21145_21152.docx#_Toc122563073)

[Figura 5 - Get dos clientes todos 13](file:///C:\Users\Gonçalo%20Cunha\Desktop\Universidade\ipca_gym\ipca_gym\Entrega_Intermedia_21140_21145_21152.docx#_Toc122563074)

[Figura 6 - Get de um cliente pelo ID 13](file:///C:\Users\Gonçalo%20Cunha\Desktop\Universidade\ipca_gym\ipca_gym\Entrega_Intermedia_21140_21145_21152.docx#_Toc122563075)

[Figura 7 - Adicionar Cliente 13](file:///C:\Users\Gonçalo%20Cunha\Desktop\Universidade\ipca_gym\ipca_gym\Entrega_Intermedia_21140_21145_21152.docx#_Toc122563076)

[Figura 8 - Remover Cliente 13](file:///C:\Users\Gonçalo%20Cunha\Desktop\Universidade\ipca_gym\ipca_gym\Entrega_Intermedia_21140_21145_21152.docx#_Toc122563077)

[Figura 9 - Alteração dados de um cliente 13](file:///C:\Users\Gonçalo%20Cunha\Desktop\Universidade\ipca_gym\ipca_gym\Entrega_Intermedia_21140_21145_21152.docx#_Toc122563078)

[Figura 10 - Camada BLL 14](file:///C:\Users\Gonçalo%20Cunha\Desktop\Universidade\ipca_gym\ipca_gym\Entrega_Intermedia_21140_21145_21152.docx#_Toc122563079)

[Figura 11 - Exemplo Utils 14](file:///C:\Users\Gonçalo%20Cunha\Desktop\Universidade\ipca_gym\ipca_gym\Entrega_Intermedia_21140_21145_21152.docx#_Toc122563080)

[Figura 12 - Logics Amostrar cliente por ID 15](file:///C:\Users\Gonçalo%20Cunha\Desktop\Universidade\ipca_gym\ipca_gym\Entrega_Intermedia_21140_21145_21152.docx#_Toc122563081)

[Figura 13 - Logics lista de Clientes 15](file:///C:\Users\Gonçalo%20Cunha\Desktop\Universidade\ipca_gym\ipca_gym\Entrega_Intermedia_21140_21145_21152.docx#_Toc122563082)

[Figura 14 - Logics Adicionar Cliente 15](file:///C:\Users\Gonçalo%20Cunha\Desktop\Universidade\ipca_gym\ipca_gym\Entrega_Intermedia_21140_21145_21152.docx#_Toc122563083)

[Figura 15 - Logics Remover Cliente 15](file:///C:\Users\Gonçalo%20Cunha\Desktop\Universidade\ipca_gym\ipca_gym\Entrega_Intermedia_21140_21145_21152.docx#_Toc122563084)

[Figura 16 - Logics Alterar dados do Cliente 15](file:///C:\Users\Gonçalo%20Cunha\Desktop\Universidade\ipca_gym\ipca_gym\Entrega_Intermedia_21140_21145_21152.docx#_Toc122563085)

[Figura 17 - Propriedades Ginásio DB 16](file:///C:\Users\Gonçalo%20Cunha\Desktop\Universidade\ipca_gym\ipca_gym\Entrega_Intermedia_21140_21145_21152.docx#_Toc122563086)

[Figura 18 - Propriedades Ginásio 16](file:///C:\Users\Gonçalo%20Cunha\Desktop\Universidade\ipca_gym\ipca_gym\Entrega_Intermedia_21140_21145_21152.docx#_Toc122563087)

# Introdução

## Contextualização

Provindo da ideia do projeto inicial, esta unidade curricular tem como propósito a construção da arquitetura do sistema, implementando a API do projeto que irá executar serviços web.

## Motivação e Objetivos

A ideia de um sistema para o ginásio foi originada pela ideia de futuramente o IPCA vir a ter mais instalações à medida que este vai crescendo e desta forma existir uma forma de gerir o mesmo e ainda ajudar os clientes.

Temos por objetivos pessoais:

* Cimentar conhecimentos obtidos ao longo do percurso académico;

Objetivos do projeto:

* Construir a arquitetura do sistema
* Montagem de uma API que suporte serviços web, incluído:
  + Swagger;
  + Base de dados;
  + Autenticação;

## Estrutura do Documento

O documento está estruturado de forma que seja de simples leitura. Existe recurso a referências de material fornecido pelo professor Óscar Ribeiro e/ou referências a excertos de Web grafia.

Este trabalho encontra-se também dividido em grupos, de forma a facilitar a procura e associação face ao material fornecido pelo docente.

# Produto

## 2.1. Visão do Produto

Dentro dos subtópicos possíveis encaixados no Smart Campus vai ser abordado a Saúde. Foi decidido toda uma construção em torno do desenvolvimento android que visa à nossa universidade acompanhar a vida saudável e atlética dos estudantes.

O IPCA GYM nasce após notar-se a necessidade desse mesmo acompanhamento e a falta de um setor que permita a atividade aos jovens, no sentido de incentivar aos estudantes a realizar um estilo de vida saudável.

Será então possível aos estudantes terem um acompanhamento mobile da sua atividade física, tal como os diferentes exercícios que pode fazer ao longo do seu treino.

Os gestores do ginásio conseguirão fazer uma monitorização de todas as pessoas inscritas no ginásio, já que, em conjunto com outra unidade curricular, irá ser implementado um sistema externo para gestão de acesso através de um chip/cartão eletrónico.

Este projeto visa alcançar este objetivo através da implementação de uma aplicação Mobile e de hardware de gestão de acessos para que se torne mais cómoda a utilização da mesma.

# Organização do Grupo

O grupo 7 da cadeira de Projeto Aplicado da Licenciatura de Engenharia de Sistemas Informáticos pós-laboral (LESI-PL) é constituído pelos seguintes membros de grupo: Pedro Simões, Gonçalo Cunha e João Apresentação.

O grupo tem como orientador o professor Eduardo Peixoto, que poderá participar nas reuniões de grupo, para auxiliar na concretização dos objetivos do projeto.

A nível de cargos distribuídos pelo grupo, temos:

* Gonçalo Cunha - Product Owner, Programador
* João Apresentação - Scrum Master, Programador
* Pedro Simões - Programador

**Product Owner** tem o papel de fazer a definição da visão do projeto, supervisionamento e avaliação das etapas de desenvolvimento.

**Scrum Master** tem o papel de facilitar as sprints, ajudar a equipa a manter-se concentrada nas sprints e fazer o planeamento e organização da mesma.

**Programador** tem a função desenvolver o software em si, obedecendo às tarefas fornecidas pelo Scrum master e de realizar as suas tarefas até ao fim de cada sprint.

Não está projetado a alteração de cargos entre membros de grupo, exceto em ocasiões especiais, das quais serão devidamente registadas com os membros que sofreram a alteração, data e razões da mesma.

s carregado por um hardware com uso de cartões para o ginásio.

# Diagramas

## Diagrama Entidade-Relação

Segue-se abaixo o diagrama de entidade-relação da base de dados do IPCA GYM:



Figura 1 - Diagrama de Entidade-Relação

Como entidades principais este diagrama possuí:

* **­Cliente** – dados de um cliente que está a utilizar a aplicação;
* **Funcionário** – dados de um funcionário do ginásio em causa, possuí o atributo “isAdmin” para determinar se este tem como role Gerente ou não;
* **Ginásio** – dados do ginásio em causa, entidade criada de forma que o projeto, mais tarde, tenha suporte para várias instituições académicas
* **Loja** – possui dados de todos os produtos disponíveis e indisponíveis na loja de cada ginásio
* **Atividade** – entidade criada com o propósito de analisar as entradas e saídas de cada cliente no ginásio (recebe informação do Arduíno)

De forma que fosse possível suportar alguns dados sobre outras funcionalidades, foram adicionadas as seguintes entidades:

* **Plano\_Nutricional** e Refeição – entidades que possuem dados sobre diferentes refeições e seus horários, cada ginásio define o seu plano nutricional;
* **Plano\_Treino** e Exercício – entidades que possuem dados sobre diferentes exercícios e suas descrições, cada ginásio define o seu plano de treino;
* Pedido e **Pedido\_Loja** – entidade que possui dados de cada encomenda feita pelo utilizador na loja do ginásio no qual este está inscrito;
* **Horario\_Funcionario** – regista o horário de cada funcionário, de forma a verificar a sua disponibilidade para as diferentes marcações;
* **Marcação** – possui a informação de todas as marcações marcadas pelo cliente com o funcionário, associadas a cada ginásio;
* **Classificação** – contém todas as avaliações feitas pelos clientes a cada ginásio.

# Código

## Programação por Camadas

Neste projeto, como forma de organizar o nosso código, decidimos programar em 4 camadas, isto para que que o código fique mais organizado, com melhor performance e mais seguro. A nós permite-nos também detetar anomalias e corrigir problemas de forma mais simples e direta, tudo isto porque é possível substituir partes das camadas (ou a camada toda) sem que o sistema fique todo ele comprometido.

Uma imagem com texto

Descrição gerada automaticamente

1ª Camada - Backend API

2ª Camada - BLL

3ª Camada - BOL

4ª Camada - DAL

Figura - Camadas

Na camada “Backend\_IPCA\_Gym” estão os controllers respetivos para todas as entidades do nosso sistema. Esta camada utiliza funções logic, sendo estas funções provindas da camada DAL.

Aqui é onde são executadas as chamadas à API. Também é nesta camada que está o nosso main, que é chamado quando executamos o nosso sistema.

### Uma imagem com texto Descrição gerada automaticamente13.1.1 Backend\_IPCA\_Gym

Figura - Lista dos controladores

Uma imagem com texto

Descrição gerada automaticamente

Figura - Main

Como referido na página anterior, em cada controlador corresponde a cada parte do nosso projeto e é onde são chamadas as funções que irão fazer a conexão com a bases de dados. Dentro dos controladores temos que colocar (antes de executarmos a chamada), o tipo de request que pretendemos fazer (httpget, httppost,hettpdelete…).

* Listagem de todos os clientes

### Uma imagem com texto Descrição gerada automaticamente

### 

Figura - Get dos clientes todos

* Listagem de um cliente através do seu ID

Uma imagem com texto

Descrição gerada automaticamente

Figura - Get de um cliente pelo ID

* Criação de um novo cliente

### Uma imagem com texto Descrição gerada automaticamente

### 

Figura - Adicionar Cliente

* Remoção de um cliente

Uma imagem com texto

Descrição gerada automaticamente

Figura - Remover Cliente

* Alteração dos dados relativos a um cliente

### Uma imagem com texto Descrição gerada automaticamente

Figura - Alteração dados de um cliente

### 13.1.2 LayerBLL

"BLL" significa "Business Logic Layer" e é outro termo comum usado no desenvolvimento de software para se referir a uma camada na arquitetura de uma aplicação. A camada BLL normalmente é responsável por implementar a lógica de negócios. Isso pode incluir tarefas como validar a entrada do utilizador, realizar cálculos e interagir com a camada de acesso a dados (DAL) para recuperar e armazenar dados e muito mais.

No C#, a camada de lógica de negócios foi implementada como um conjunto de classes que contém os métodos que executam a lógica de negócios da aplicação. Esses métodos são chamados pela camada de apresentação (como uma interface de utilizador) ou por outros componentes na aplicação para executar determinadas tarefas.

* Uma imagem com texto, eletrónica

  Descrição gerada automaticamenteNa parte *Logics* temos as funções logic (funções referidas anteriormente chamadas na camada Backend.

Figura - Camada BLL

* Uma imagem com texto

  Descrição gerada automaticamenteNa parte *Utils* temos o código que nos dá informação request está ou não a funcionar corretamente.

Figura - Exemplo Utils

Apresento agora um exemplo, para um cliente do nosso sistema, do que é necessário para que seja permitido à camada da apresentação fornecer métodos para a camada de negócios.

### Uma imagem com texto, captura de ecrã, ecrã, prateado Descrição gerada automaticamenteUma imagem com texto Descrição gerada automaticamente

Figura - Logics Amostrar cliente por ID

Figura - Logics lista de Clientes

Uma imagem com texto

Descrição gerada automaticamenteUma imagem com texto

Descrição gerada automaticamenteUma imagem com texto

Descrição gerada automaticamente

Nestas imagens podemos verificar que em todos os requests (neste caso para o cliente) necessitamos de comunicar com a camada BOL (ClienteService).

Figura - Logics Adicionar Cliente

Figura - Logics Remover Cliente

Figura - Logics Alterar dados do Cliente

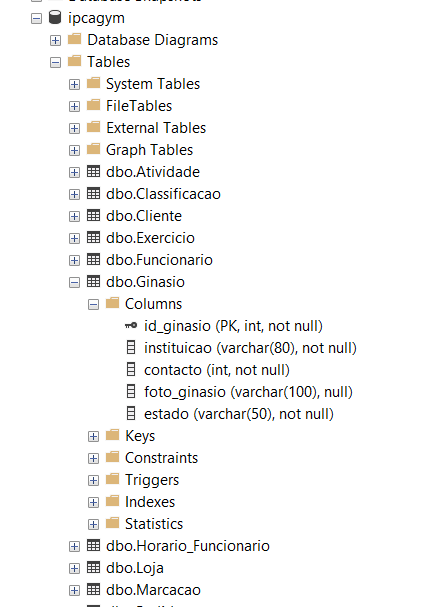
### 13.1.3 LayerBOL

A camada "BOL" significa "Business Object Layer" e é um termo comum usado no desenvolvimento de software referir-se a uma camada que é responsável por representar entidades de negócios e os seus relacionamentos.

O "BOL" normalmente fica entre a camada de apresentação (como uma interface de utilizador) e a camada de acesso a dados (que é responsável pela comunicação com a base de dados ou outro armazenamento de dados).

No C#, a camada do objeto de negócios foi também implementada como um conjunto de classes respetivos a cada entidade do negócio e os relacionamentos entre elas. Nestas classes podemos observar também as propriedades que correspondem aos atributos das entidades de negócios e métodos que executam a lógica de negócios.

Em baixo, e com o código já devidamente comentado, podemos observar as propriedades que a entidade Ginásio tem e estão a ser chamados, ao lado da figura podemos também observar na base de dados do projeto que os campos são os mesmos da tabela.

Uma imagem com texto

Descrição gerada automaticamente

Figura - Propriedades Ginásio DB

Figura - Propriedades Ginásio

### 13.1.3 LayerDAL

"DAL" significa "Data Access Layer" e é um termo comum usado no desenvolvimento de software para referir-se a uma camada na arquitetura de uma aplicação que é responsável pela comunicação com uma base de dados ou outro armazenamento de dados. A DAL normalmente é responsável por tarefas como executar consultas SQL e interagir com a base de dados para recuperar e armazenar dados.

No C#, a camada de acesso a dados foi, como as restantes, implementada como um conjunto de classes que contém os métodos que executam as interações da base de dados. Esses métodos são chamados pela Business Logic Layer (BLL) e outros componentes na aplicação para recuperar e armazenar dados na base de dados.

# Conclusão

O progresso atual do projeto foi possível com o recurso a ferramentas de suporte, com uma distribuição por sprints semanais e organização de reuniões para estruturação das tarefas. Com esta 3ª entrega foi possível terminar todos os diagramas e avançar para a implementação a ser avaliado em futuras entregas (4 e 5).

# Bibliografia

**Repositório GitHub**

<https://github.com/Presentation12/Ipca_Gym>

**Figma**

<https://www.figma.com/file/Q4tM34gl91b9fhrGvUeXRs/MileriuPT's-teamlibrary?node-id=0%3A1&t=p9cWit1VJHUNwf2m-1>

**Sites para produção dos diagramas:**

<https://app.diagrams.net>

<https://online.visual-paradigm.com/pt/login.jsp>