

**Programação Avançada**

Gestão de Projetos e Estágios



**Sérgio Costa 2020129026**

**Roberto Tarta 2020136419**

Licenciatura em Engenharia Informática – Curso Europeu

Coimbra, 25 de abril de 2022

Índice

[Introdução 3](#_Toc101789072)

[Organização 3](#_Toc101789073)

[Decisões tomadas na implementação 3](#_Toc101789074)

[Diagrama da máquina de estados 3](#_Toc101789075)

[*Diagramas de outros padrões de programação* 3](#_Toc101789076)

[Classes utilizadas 3](#_Toc101789077)

[Descrição do relacionamento entre as classes 3](#_Toc101789078)

[Funcionalidade da aplicação 3](#_Toc101789079)

# Introdução

Este trabalho prático está a ser desenvolvido no âmbito da unidade curricular de Programação Avançada. O projeto consiste na implementação de uma aplicação de apoio ao processo de gestão de projetos e estágios do Departamento de Engenharia Informática e de Sistemas do ISEC.

# Organização

Seguindo as indicações sobre a arquitetura do projeto, nós criamos a seguinte organização dentro da package “**pt.isec.pa.apoio\_poe**”, incluindo dentro desta as seguintes packages:

* **model** - contem todas as classes que implementam a gestão de PoE e toda a logica associada.
  + **fsm** - contém toda a hierarquia dos estados, incluindo a interface e a classe Context, para a interface poder aceder a todas as funcionalidades.
  + **data** - contém as classes que representam as estruturas de dados e que disponibilizam todas as funcionalidades.
* **ui** - contem todas as classes que implementam a interface, seja em modo texto (meta 1), ou em modo gráfico (meta 2)
  + **text** - classe que implementa a interface em modo texto.
  + **gui** - classes que implementam a interface em modo gráfico em JavaFx (apenas será desenvolvido na segunda meta).
* **utils** – contem a classe que vai ajudar a ler os dados introduzidos pelo utilizador, é usado pela interface

# Decisões tomadas na implementação

-- A completar --

# Diagrama da máquina de estados

Para ajudar na implementação deste projeto, nos criamos um diagrama de máquinas de estados para ajudar na interação UI – Aplicação. Com este processo, nos conseguimos aceder a toda a informação essencial e perceber quando é que o programa vai mudar de estado e como devemos lidar com o processo. Na imagem abaixo encontra-se o nosso diagrama de estados, que serviu como base para a nossa implementação. Os estados escolhidos foram com base nos momentos em que o utilizador tem que tomar uma decisão para que o programa possa continuar em funcionamento.

# *Diagramas de outros padrões de programação*

-- A completar --

# Classes utilizadas

-- A completar --

# Descrição do relacionamento entre as classes

-- A completar --

# Funcionalidade da aplicação

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Feito** | **Parcial** | **Por fazer** |
| Alternar entre modos de gestão | *X* |  |  |
| Inserir, consultar, editar e eliminar alunos | *X* |  |  |
| Inserir, consultar, editar e eliminar professores | *X* |  |  |
| Inserir, consultar, editar e eliminar propostas | *X* |  |  |
| Inserir, consultar, editar e eliminar candidaturas | *X* |  |  |
| Exportar alunos, professores, propostas e candidatura | *X* |  |  |
| Obter a lista de alunos personalizada |  | *X* |  |
| Obter a lista de propostas personalizada |  |  | *X* |
| Atribuição automática das autopropostas / propostas de professores |  |  | *X* |
| Atribuição automática de uma proposta disponível aos alunos ainda sem atribuições |  |  | *X* |
| Atribuição manual de propostas |  |  |  |
| Remoção manual de uma atribuição previamente realizada ou de todas as atribuições |  |  |  |
| Associação automática dos docentes proponentes de projetos como orientador |  |  |  |
| Atribuição, consulta, alteração e eliminação de um orientador do ISEC |  |  |  |
| Obter a lista de orientadores personalizada |  |  |  |
| Obtenção de dados diversos sobre todo o processo |  |  |  |
| Fechar a fase |  |  |  |
| Avançar a fase |  |  |  |
| Voltar a fase anterior |  |  |  |
|  |  |  |  |
|  |  |  |  |