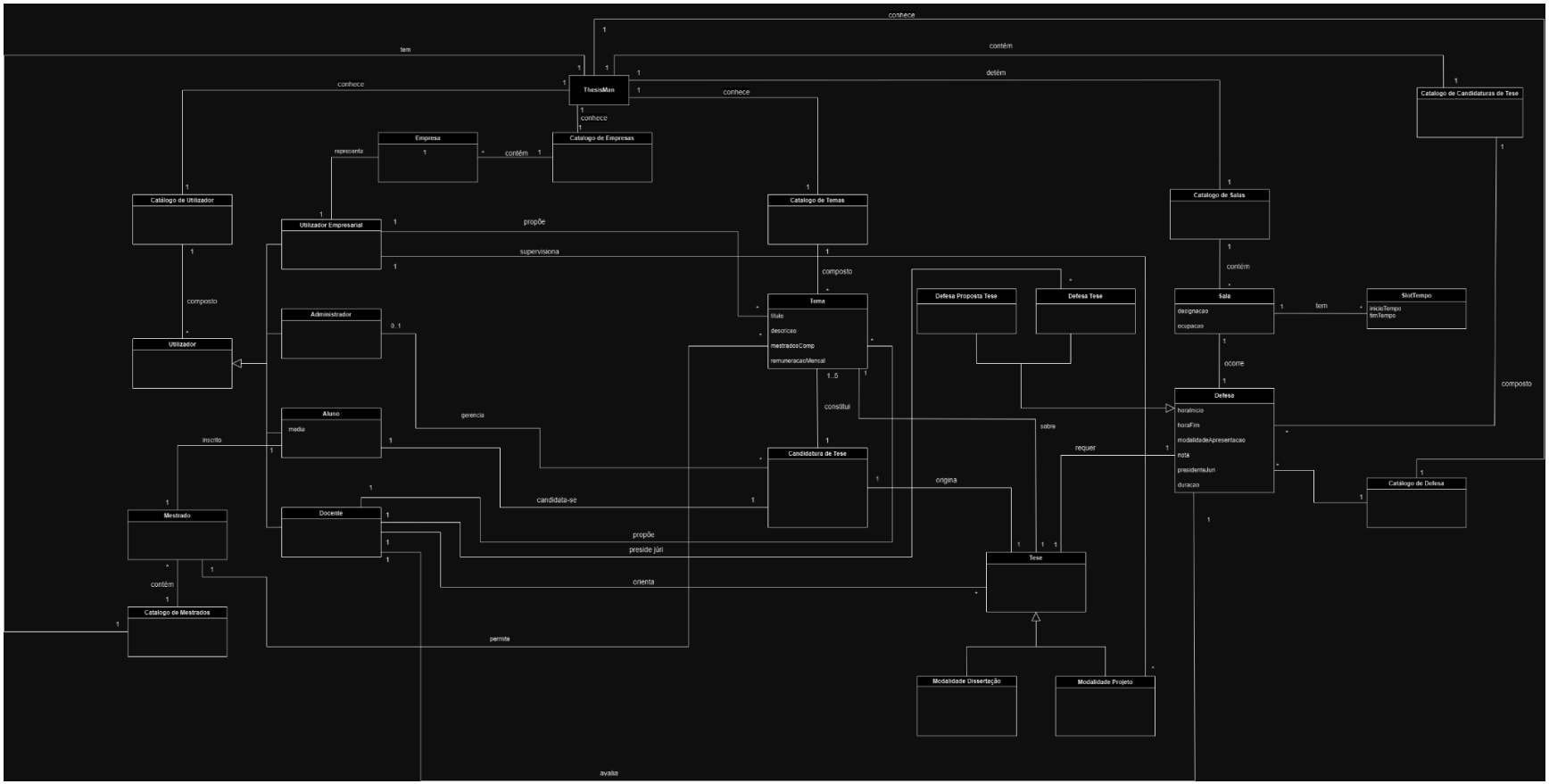
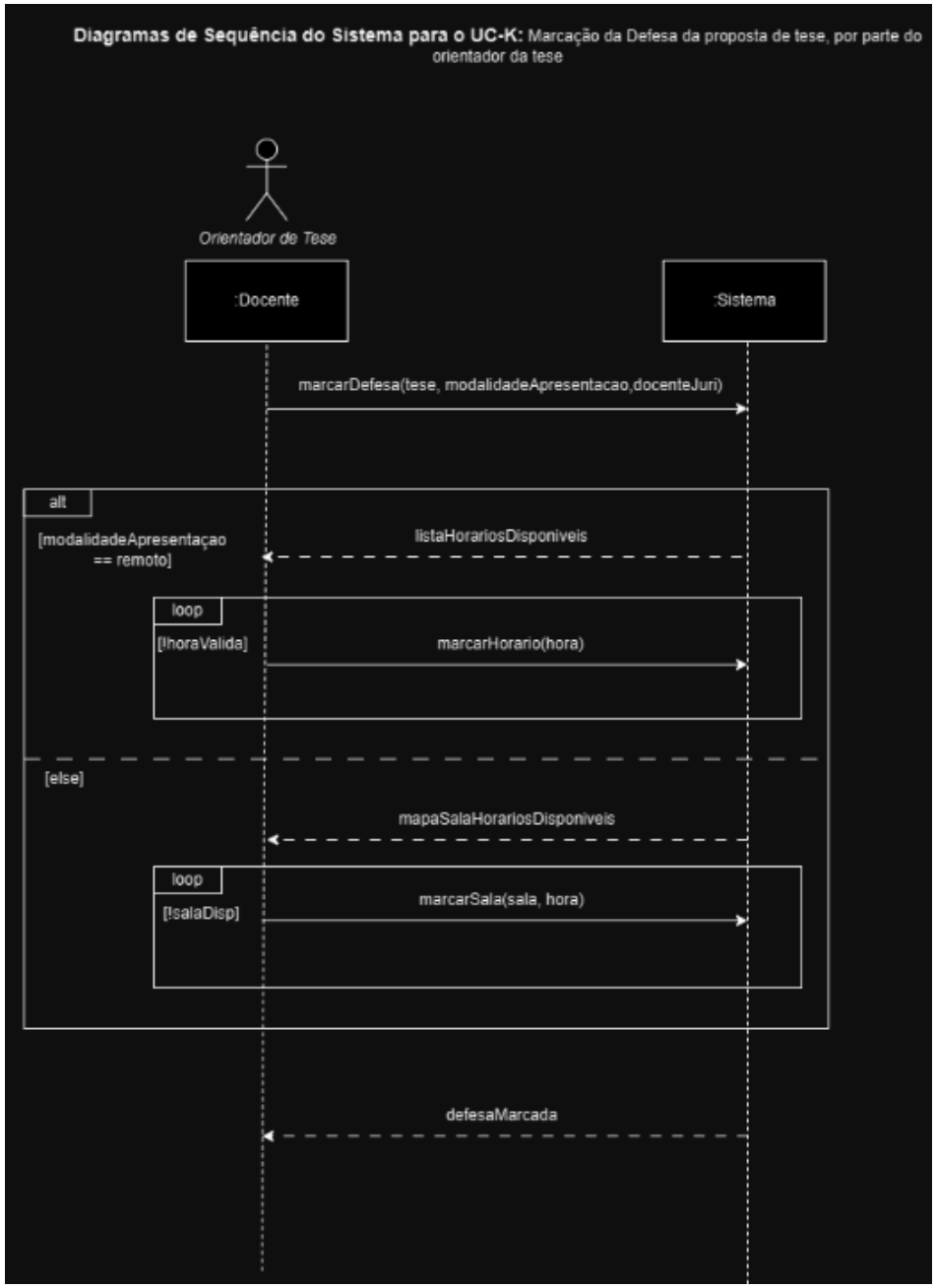

conhece
tem
contém
detém
conhece
ThesisMan
conhece
conhece
Catalogo de Candidaturas de Tese
representa
Empresa
contém
Catalogo de Empresas
Catálogo de Utilizador
Utilizador Empresarial
propõe
supervisiona
Catalogo de Temas
Catalogo de Salas
composto
contém
Tema
titulo
descricao
mestradosComp
remuneracaoMensal
Defesa Proposta Tese
Defesa Tese
Sala
designacao
ocupacao
tem
SlotTempo
inicioTempo
fimTempo
composto
Utilizador
Administrador
0..1
ocorre
composto
Defesa
horaInicio
horaFim
modalidadeApresentacao
nota
presidenteJuri
duracao
Aluno
media
gerencia
1..5
constitui
sobre
inscrito
Candidatura de Tese
requer
origina
Catálogo de Defesa
Mestrado
Docente
candidata-se
propõe
preside júri
Tese
contém
orienta
Catalogo de Mestrados
permite
Modalidade Dissertação
Modalidade Projeto
avalia


Diagramas de Sequência do Sistema para o UC-K: Marcação da Defesa da proposta de tese, por parte do orientador da tese
Orientador de Tese
:Docente
:Sistema
marcarDefesa(tese, modalidadeApresentacao,docenteJuri)
alt
[modalidadeApresentaçao == remoto]
listaHorariosDisponiveis
loop
[!horaValida]
marcarHorario(hora)
[else]
mapaSalaHorariosDisponiveis
loop
[!salaDisp]
marcarSala(sala, hora)
defesaMarcada

Contrato

Operação: marcarDefesa(tese, modalidadeApresentacao)

Casos de Uso:
* UC K

Pré-Condições:
* Existe uma tese:Tese que representa a proposta de Tese que irá ser apresentada na Defesa
* Existe uma associação entre tese e aluno, sendo que é o aluno que está a desenvolver a tese
* Existe uma associação entre tese e docente, sendo docente o orientador da tese
* Existe uma associação entre tese e utilizador empresarial, caso a tese seja da modalidade projeto
* Existe uma associação entre tese e candidaturaDeTese
* Existe uma modalidadeApresentacao que representa o tipo de apresentação da tese e que pode tomar os valores presencial ou remoto
* Existe um docenteJuri:Docente que representa o segundo docente que fará parte do júri da defesa
* Existe uma associação entre aluno e mestrado, sendo que o aluno está a realizar o mestrado

*Pós-Condições:**
* Existe uma defesa:Defesa que representa a defesa que se está a marcar e por sua vez
* existe uma defesaProposta:DefesaPropostaTese que representa a especialização da defesa que se está a marcar
* Existe uma associação entre aluno e defesa
* Existe uma associação entre docenteJuri e defesa
* defesa.presidenteJuri = tese.docenteOrientador
* defesa.modalidadeApresentacao = modalidadeApresentacao
* defesa.duracao = 60 (minutos)

Operação: marcarHorarios(hora)

Casos de Uso:
* UC K

Pré-Condições:
* Existe uma defesa:Defesa que representa a defesa que se está a marcar e por sua vez
* existe uma defesaProposta:DefesaPropostaTese que representa a especialização da defesa que se está a marcar
* Existe uma associação entre aluno e defesaProposta
* Existe uma associação entre docenteJuri e defesaProposta
* defesa.presidenteJuri = tese.docenteOrientador
* defesa.modalidadeApresentacao = modalidadeApresentacao (== remoto)
* defesa.duracao = 60 (minutos)

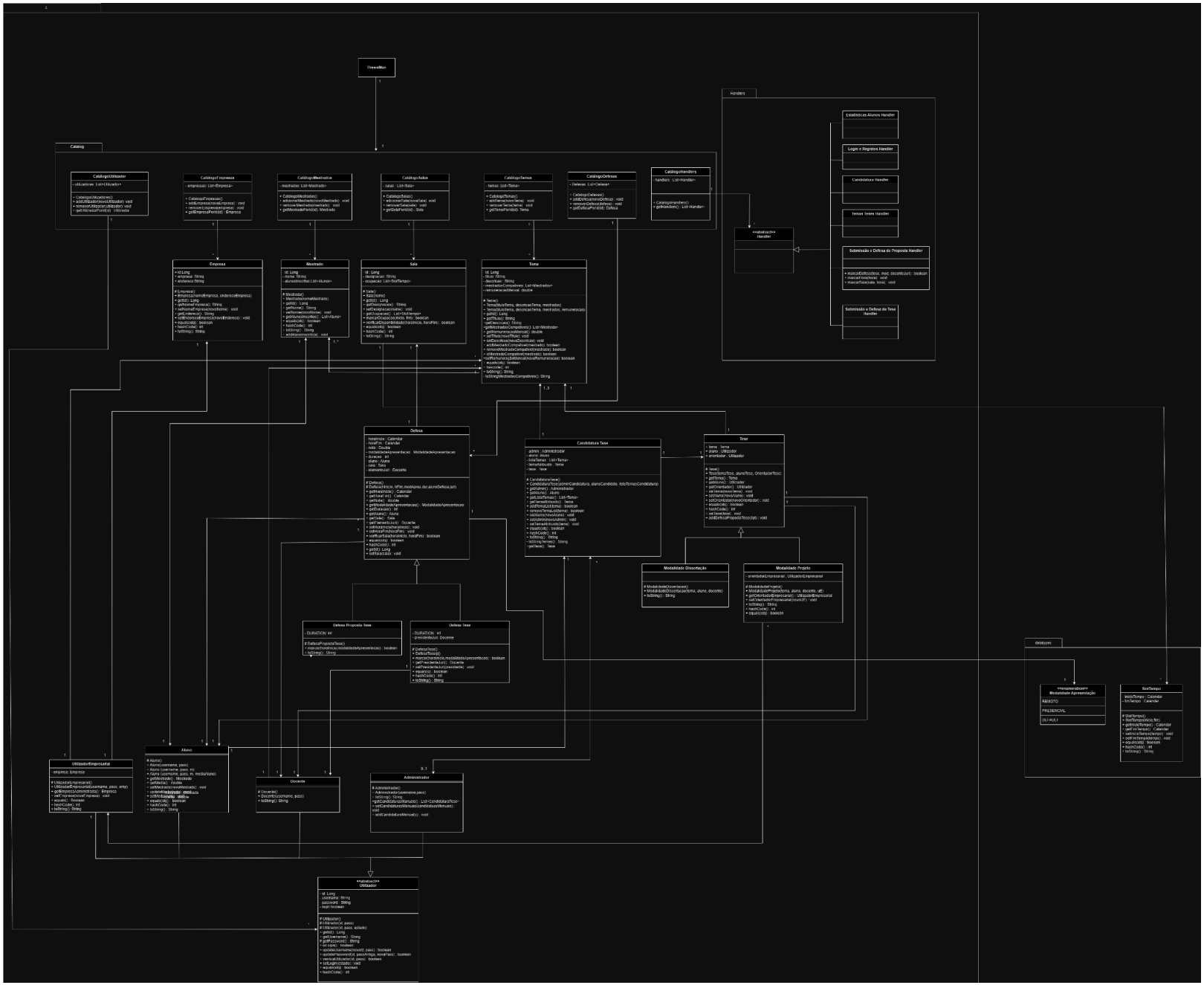
*Pós-Condições:**
* defesa.horaInicio = hora
* defesa.horaFim = hora + 60 minutos

Operação: marcarSala(sala,hora)

Casos de Uso:
* UC K

Pré-Condições:
* Existe uma defesa:Defesa que representa a defesa que se está a marcar e por sua vez
* existe uma defesaProposta:DefesaPropostaTese que representa a especialização da defesa que se está a marcar
* Existe uma associação entre aluno e defesaProposta
* Existe uma associação entre docenteJuri e defesaProposta
* defesa.presidenteJuri = tese.docenteOrientador
* defesa.modalidadeApresentacao = modalidadeApresentacao (== presencial)
* defesa.duracao = 60 (minutos)
* Existe um catSalas:CatalogoDeSalas que representa todas as salas da instituição
* Existe sala:Sala que representa a sala onde se pretende marcar a defesa, que pertence à catSalas e que está disponível à hora

*Pós-Condições:**
* defesa.horaInicio = hora
* defesa.horaFim = hora + 60 minutos
* Existe uma associação entre defesa e sala
* hora de marcação da defesa é removida da sala.disponibilidade


ItensaMan
Handlers
Estatísticas Alunos Handler
Logro e Registro Handler
Candidatura Handler
Tentus Ivens Handler
Submissão e Defesa de Proposta Handler
Submissão e Defesa de Tese Handler
«Validador» Handler
Catalog
CatálogoUtilizador
CatálogoEmpresa
CatálogoMestrados
CatálogoSalas
CatálogoTemas
CatálogoDefesas
CatálogoHandlers
Empresa
Mestrado
Sala
Tema
Defesa
Candidatura Tese
Tese
Modalidade Dissertação
Modalidade Projeto
Defesa Proposta Tese
Defesa Tese
Aluno
UtilizadorEmpresarial
Docente
Administrador
«Validador» Utilizador
«Implementador» ModalidadeApresentação
SlotTempo

Construção de Sistemas de Software
Licenciatura em Engenharia Informática
Ano Letivo: 2023/24

ThesisMan : Gestão de Teses de Mestrado

Vasco Baldé (58174), José Martins (58228) e Maria Silva(58243)

Introdução

A aplicação *ThesisMan* é uma aplicação que gere teses de mestrado e este relatório tem como objetivo justificar o mapeamento das diversas classes que constituem a aplicação.

Classes

Utilizador

Esta classe é abstrata e representa os vários tipos de utilizadores que a aplicação pode ter (stakeholders).

Anotações

- **@Entity**
  A classe é anotada com @Entity a fim de indicar que é uma entidade JPA que será mapeada para uma tabela na base de dados.

- **@Table(name = “Utilizadores”)**
  A classe Utilizador possui a anotação @Table(name = “Utilizadores”) que especifica o nome da tabela como "Utilizadores", seguindo convenções de nomenclatura.

- **@Inheritance(strategy = InheritanceType.SINGLE_TABLE)**
  Escolhemos utilizar a estratégia de herança de tabela única (SINGLE_TABLE), o que possibilita que todas as subclasses de Utilizador compartilhem a mesma tabela a fim de uma melhor organização de informação.

- **@DiscriminatorColumn(name = "TIPO_UTILIZADOR")**
  Recorremos ao discriminador de modo a especificar a coluna que distingue os diferentes tipos de utilizadores.

- **@Id**
  Serve para marcar o atributo da classe id como chave primária do tipo em questão.

- **@GeneratedValue(strategy = GenerationType.SEQUENCE)**
  Esta anotação específica que os valores de id que serão gerados automaticamente pela base de dados utilizarão uma estratégia sequencial.

- **@NonNull**
  Alguns atributos da classe foram anotados desta forma a fim de indicar que não podem tomar valor nulo.

Aluno

Esta classe representa os alunos e é uma extensão do Utilizador.

Anotações

- **@Entity**
  A classe é anotada com @Entity a fim de indicar que é uma entidade JPA que será mapeada para uma tabela na base de dados.

- **@DiscriminatorValue(“aluno")**
  Específica na coluna da tabela de utilizadores que se trata de um utilizador do tipo Aluno.

- **@ManyToOne()**
  Esta anotação faz compreender que o mestrado a que o aluno pertence também possui outros alunos.

- **@JoinColumn(name = “mestrado_id”)**
  Esta anotação específica que a classe Aluno utiliza a coluna “mestrado_id”como chave estrangeira para mapear essa relação na base de dados.

Docente

Esta classe representa os docentes e é uma extensão do Utilizador.

Anotações

- **@Entity**
  A classe é anotada com @Entity a fim de indicar que é uma entidade JPA que será mapeada para uma tabela na base de dados.

- **@DiscriminatorValue(“docente")**
  Específica na coluna da tabela de utilizadores que se trata de um utilizador do tipo Docente.

Administrador

Esta classe representa os administradores da aplicação e é uma extensão do Utilizador.

Anotações

- **@Entity**
  A classe é anotada com @Entity a fim de indicar que é uma entidade JPA que será mapeada para uma tabela na base de dados.

- **@DiscriminatorValue(“admin")**
  Específica na coluna da tabela de utilizadores que se trata de um utilizador do tipo Administrador.

- **@Nullable**
  O atributo candidaturasManuais da classe Tese possui esta anotação de modo a ficar explícito que pode tomar o valor nulo.

- **@OneToMany**
  Alguns atributos da classe foram anotados desta forma a fim de indicar que a sua relação com o outro tipo de objeto é sempre de um para muitos.

Utilizador Empresarial

Esta classe representa os representantes das empresas que fornecem temas para as teses e é uma extensão do Utilizador.

Anotações

- **@Entity**
  A classe é anotada com @Entity a fim de indicar que é uma entidade JPA que será mapeada para uma tabela na base de dados.

- **@DiscriminatorValue(“utilizadorEmpresarial")**
  Específica na coluna da tabela de utilizadores que se trata de um utilizador do tipo UtilizadorEmpresarial.

- **@NonNull**
  O atributo empresa foi anotado desta forma de modo a referir que não pode tomar valor nulo.

- **@OneToOne()**
  Esta anotação faz compreender que a empresa só pode ter um representante e que um representante só pode representar uma empresa.

Empresa

Esta classe representa as várias empresas que vão fornecer temas para as teses.

Anotações

- **@Entity**
  A classe é anotada com @Entity a fim de indicar que é uma entidade JPA que será mapeada para uma tabela na base de dados.

- **@Id**
  Serve para marcar o atributo da classe id como chave primária do tipo em questão.

- **@GeneratedValue(strategy = GenerationType.AUTO)**
  Esta anotação específica que os valores de id que serão gerados automaticamente pela base de dados utilizarão uma estratégia automática.

- **@NonNull**
  Alguns atributos da classe foram anotados desta forma a fim de indicar que não podem tomar valor nulo.

Mestrado

Esta classe representa os vários mestrados que a instituição de ensino possui.

Anotações

- **@Entity**
  A classe é anotada com @Entity a fim de indicar que é uma entidade JPA que será mapeada para uma tabela na base de dados.

- **@Table(name = “Mestrados”)**
  A classe Mestrados possui a anotação @Table(name = “Mestrados”) que especifica o nome da tabela como "Mestrados", seguindo convenções de nomenclatura.

- **@Id**
  Serve para marcar o atributo da classe id como chave primária do tipo em questão.

- **@GeneratedValue(strategy = GenerationType.SEQUENCE)**

Esta anotação específica que os valores de id que serão gerados automaticamente pela base de dados utilizarão uma estratégia sequencial.

- **@NonNull**
  Alguns atributos da classe foram anotados desta forma a fim de indicar que não podem tomar valor nulo.

- **@Collumn(name = "ALUNOS_INSCRITOS)**
  Esta anotação faz compreender que a coluna que mapeia os alunos inscritos se chama "ALUNOS_INSCRITOS".

Sala

Esta classe representa as várias salas que a instituição de ensino possui para a realização de defesas de tese nos diferentes estágios.

Anotações

- **@Entity**
  A classe é anotada com @Entity a fim de indicar que é uma entidade JPA que será mapeada para uma tabela na base de dados.

- **@Id**
  Serve para marcar o atributo da classe id como chave primária do tipo em questão.

- **@GeneratedValue(strategy = GenerationType.AUTO)**
  Esta anotação específica que os valores de id que serão gerados automaticamente pela base de dados utilizarão uma estratégia automática.

- **@NonNull**
  Alguns atributos da classe foram anotados desta forma a fim de indicar que não podem tomar valor nulo.

- **@ElementCollection**
  Esta anotação faz compreender que uma sala pode ter vários intervalos de tempo ocupados.

SlotTempo

Esta classe representa intervalos de tempo.

Anotações

- **@Embeddable**
  A classe possui esta anotação a fim de explicitar que a classe é incorporável em outras entidades como parte da sua estrutura, possibilitando isto a reutilização de campos complexos sem a necessidade de criar uma entidade separada.

Tema

Esta classe representa os temas das teses.

Anotações

- **@Entity**
  A classe é anotada com @Entity a fim de indicar que é uma entidade JPA que será mapeada para uma tabela na base de dados.

- **@Id**
  Serve para marcar o atributo da classe id como chave primária do tipo em questão.

- **@GeneratedValue(strategy = GenerationType.SEQUENCE)**
  Esta anotação específica que os valores de id que serão gerados automaticamente pela base de dados utilizarão uma estratégia sequencial.

- **@NonNull**
  Alguns atributos da classe foram anotados desta forma a fim de indicar que não podem tomar valor nulo.

- **@OneToMany(cascade = CascadeType.ALL)**
  Esta anotação faz compreender que cada tema pode ter mais do que um mestrado e que por sua vez cada alteração feita ao tema vai também ser compreendida no mestrado.

- **@JoinTable(name = “MESTRADOS_COMPATIVEIS”, joinColumns = @JoinColumn(name = “tema_id”)**
  A lista de mestrados compatíveis ao tema é anotada desta forma a fim de se definir uma tabela de junção em um relacionamento muitos para muitos entre entidades, especificando-se o nome da tabela como

“MESTRADOS_COMPATIVEIS” e tema_id será uma coluna da tabela à qual estão associados os mestrados.

Tese

Esta classe é abstrata e representa os dois tipos de teses que podem ser feitas.

Anotações

- **@Entity**
  A classe é anotada com @Entity a fim de indicar que é uma entidade JPA que será mapeada para uma tabela na base de dados.

- **@Table(name = “Teses”)**
  A classe Teses possui a anotação @Table(name = “Teses”) que especifica o nome da tabela como "Teses", seguindo convenções de nomenclatura.

- **@Inheritance(strategy = InheritanceType.SINGLE_TABLE)**
  Escolhemos utilizar a estratégia de herança de tabela única (SINGLE_TABLE), o que possibilita que todas as subclasses de Tese compartilhem a mesma tabela a fim de uma melhor organização de informação.

- **@DiscriminatorColumn(name = "MODALIDADE")**
  Recorremos ao discriminador de modo a especificar a coluna que distingue os diferentes tipos de teses que existem.

- **@Id**
  Serve para marcar o atributo da classe id como chave primária do tipo em questão.

- **@GeneratedValue(strategy = GenerationType.SEQUENCE)**
  Esta anotação específica que os valores de id que serão gerados automaticamente pela base de dados utilizarão uma estratégia sequencial.

- **@NonNull**
  Alguns atributos da classe foram anotados desta forma a fim de indicar que não podem tomar valor nulo.

- **@OneToOne**

Alguns atributos da classe foram anotados desta forma a fim de indicar que a sua relação com o outro tipo de objeto é sempre de um para um.

- **@Nullable**
  O atributo document da classe Tese possui esta anotação de modo a ficar explícito que pode tomar o valor nulo por agora.

- **@OneToMany**
  Alguns atributos da classe foram anotados desta forma a fim de indicar que a sua relação com o outro tipo de objeto é sempre de um para muitos.

ModalidadeDissertacao

Esta classe representa uma modalidade de Tese, nomeadamente a dissertação.

Anotações

- **@Entity**
  A classe é anotada com @Entity a fim de indicar que é uma entidade JPA que será mapeada para uma tabela na base de dados.

- **@DiscriminatorValue(“DISSERTACAO")**
  Específica na coluna da tabela de teses que se trata de um tese do tipo ModalidadeDissertacao.

ModalidadeProjeto

Esta classe representa uma modalidade de Tese, nomeadamente a projeto.

Anotações

- **@Entity**
  A classe é anotada com @Entity a fim de indicar que é uma entidade JPA que será mapeada para uma tabela na base de dados.

- **@DiscriminatorValue(“PROJETO")**
  Específica na coluna da tabela de teses que se trata de um tese do tipo ModalidadeProjeto.

- **@NonNull**
  Alguns atributos da classe foram anotados desta forma a fim de indicar que não podem tomar valor nulo.

- **@ManyToOne**

Alguns atributos da classe foram anotados desta forma a fim de indicar que a sua relação com o outro tipo de objeto é sempre de muitos para um.

CandidaturaTese

Esta classe representa uma candidatura de tese.

Anotações

- **@Entity**
  A classe é anotada com @Entity a fim de indicar que é uma entidade JPA que será mapeada para uma tabela na base de dados.

- **@Id**
  Serve para marcar o atributo da classe id como chave primária do tipo em questão.

- **@GeneratedValue(strategy = GenerationType.AUTO)**
  Esta anotação específica que os valores de id que serão gerados automaticamente pela base de dados utilizarão uma estratégia automática.

- **@ManyToOne**
  Alguns atributos da classe foram anotados desta forma a fim de indicar que a sua relação com o outro tipo de objeto é sempre de muitos para um.

- **@Nullable**
  O atributo admin da classe CandidaturaTese possui esta anotação de modo a ficar explícito que pode tomar o valor nulo.

- **@OneToOne**
  Alguns atributos da classe foram anotados desta forma a fim de indicar que a sua relação com o outro tipo de objeto é sempre de um para um.

- **@OneToMany**
  Alguns atributos da classe foram anotados desta forma a fim de indicar que a sua relação com o outro tipo de objeto é sempre de um para muitos.

Defesa

Esta classe é abstrata e representa os dois tipos de defesas que podem ser feitas.

Anotações

- **@Entity**
  A classe é anotada com @Entity a fim de indicar que é uma entidade JPA que será mapeada para uma tabela na base de dados..

- **@DiscriminatorColumn(name = "TYPE")**
  Recorremos ao discriminador de modo a especificar a coluna que distingue os diferentes tipos de defesas.

- **@Id**
  Serve para marcar o atributo da classe id como chave primária do tipo em questão.

- **@GeneratedValue(strategy = GenerationType.AUTO)**
  Esta anotação específica que os valores de id que serão gerados automaticamente pela base de dados utilizarão uma estratégia automática.

- **@NonNull**
  Alguns atributos da classe foram anotados desta forma a fim de indicar que não podem tomar valor nulo.

- **@OneToOne**
  Alguns atributos da classe foram anotados desta forma a fim de indicar que a sua relação com o outro tipo de objeto é sempre de um para um.

- **@Nullable**
  O atributo sala da classe Defesa possui esta anotação de modo a ficar explícito que pode tomar o valor nulo dependendo da modalidade de apresentação.

DefesaTese

Esta classe representa uma modalidade de Defesa, nomeadamente a defesa da tese.

Anotações

- **@Entity**

A classe é anotada com @Entity a fim de indicar que é uma entidade JPA que será mapeada para uma tabela na base de dados..

- **@DiscriminatorValue(“DEFESA_TESE”)**
  Específica na coluna da tabela de defesas que se trata de uma defesa do tipo DefesaTese.

- **@NonNull**
  Alguns atributos da classe foram anotados desta forma a fim de indicar que não podem tomar valor nulo.

- **@OneToOne**
  Alguns atributos da classe foram anotados desta forma a fim de indicar que a sua relação com o outro tipo de objeto é sempre de um para um.

DefesaPropostaTese

Esta classe representa uma modalidade de Tese, nomeadamente a defesa da proposta de tese.

Anotações

- **@Entity**
  A classe é anotada com @Entity a fim de indicar que é uma entidade JPA que será mapeada para uma tabela na base de dados..

- **@DiscriminatorValue(“DEFESA_PROPOSTA_TESE”)**
  Específica na coluna da tabela de defesas que se trata de uma defesa do tipo DefesaPropostaTese.