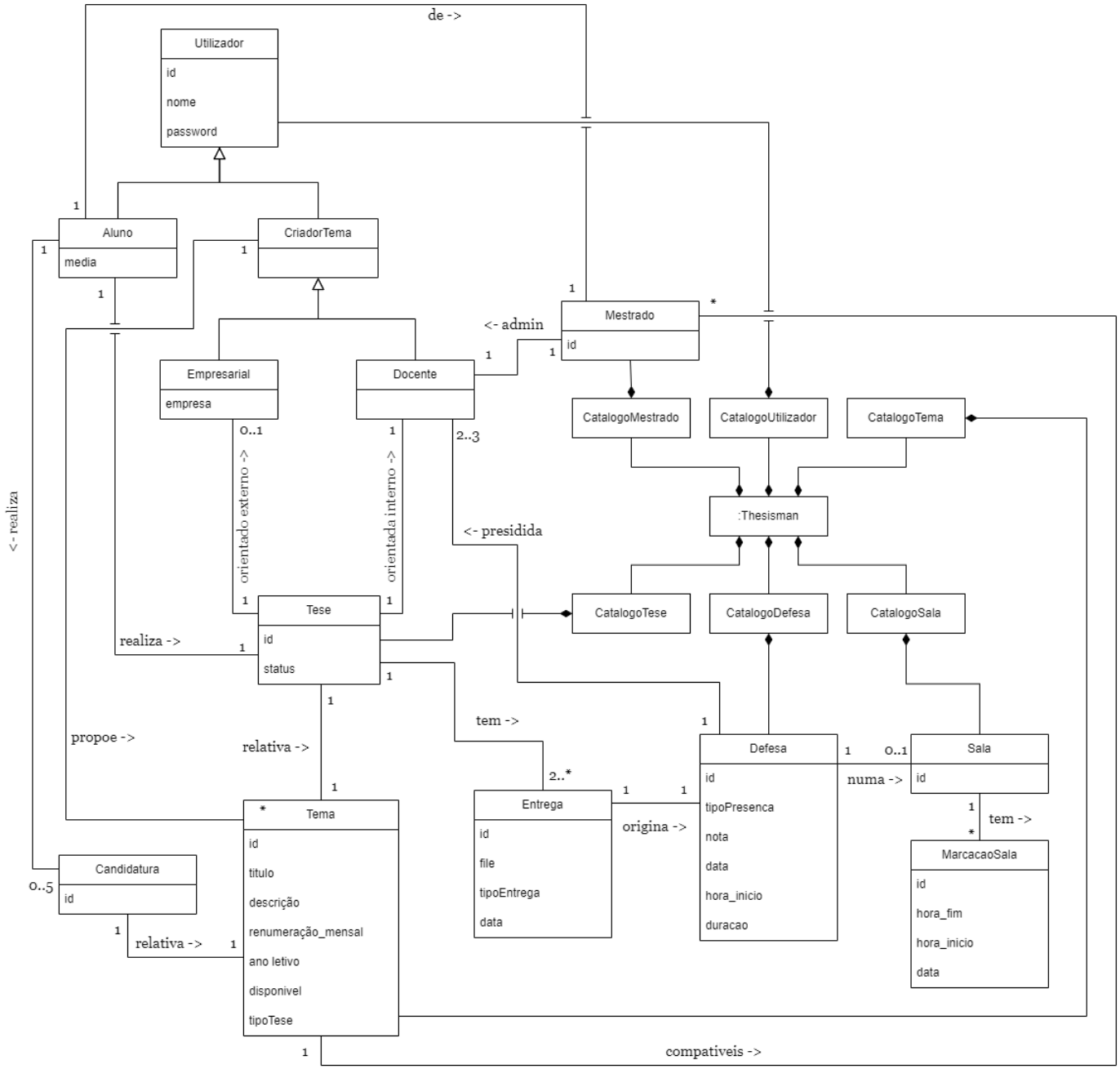
Relatório do projeto Thesisman

Disciplina de Construção de Sistemas de Software:

- Martim Pereira fc58223
- Daniel Nunes fc58257
- João Pereira fc58189

Modelo Domínio



SSD do caso de uso K

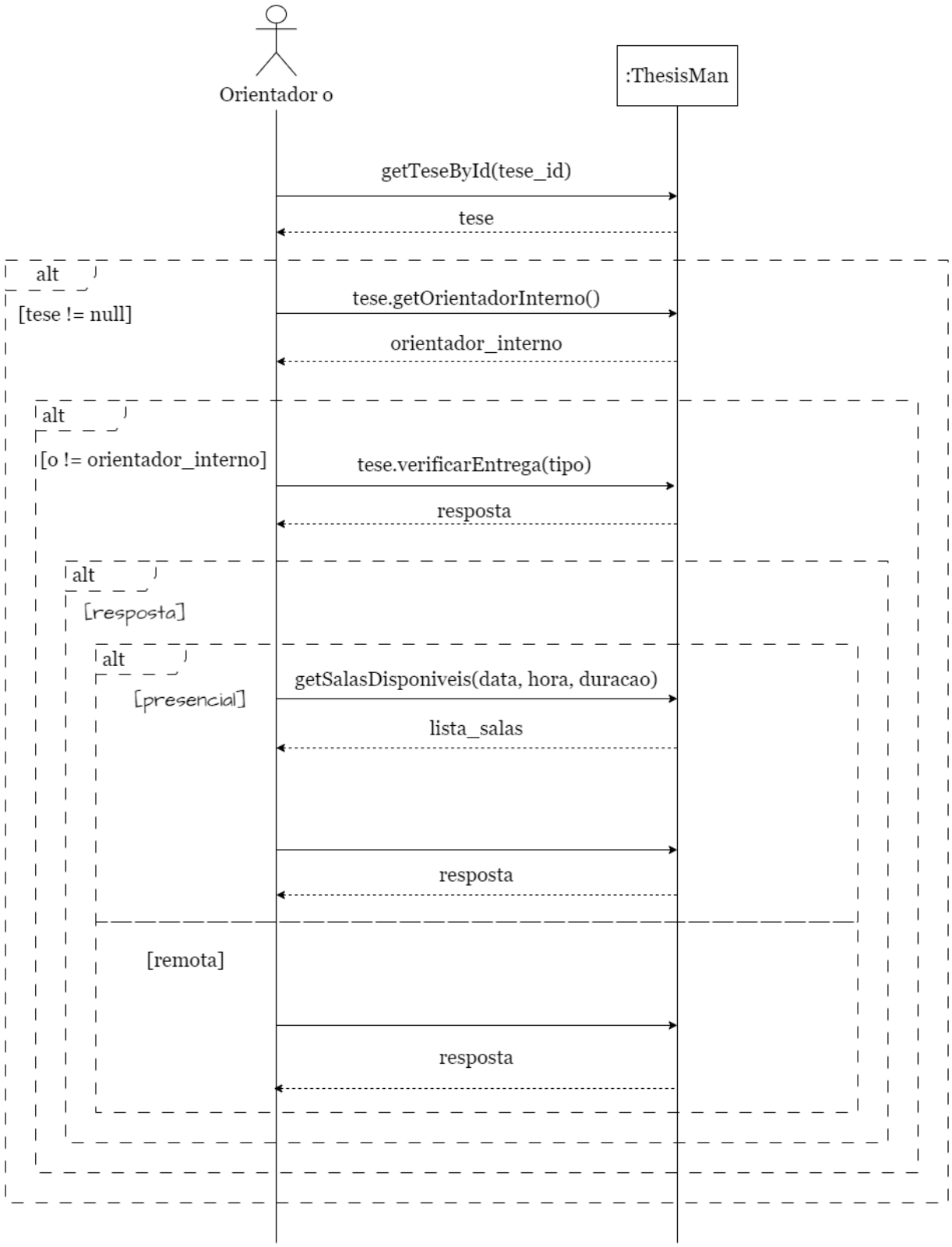
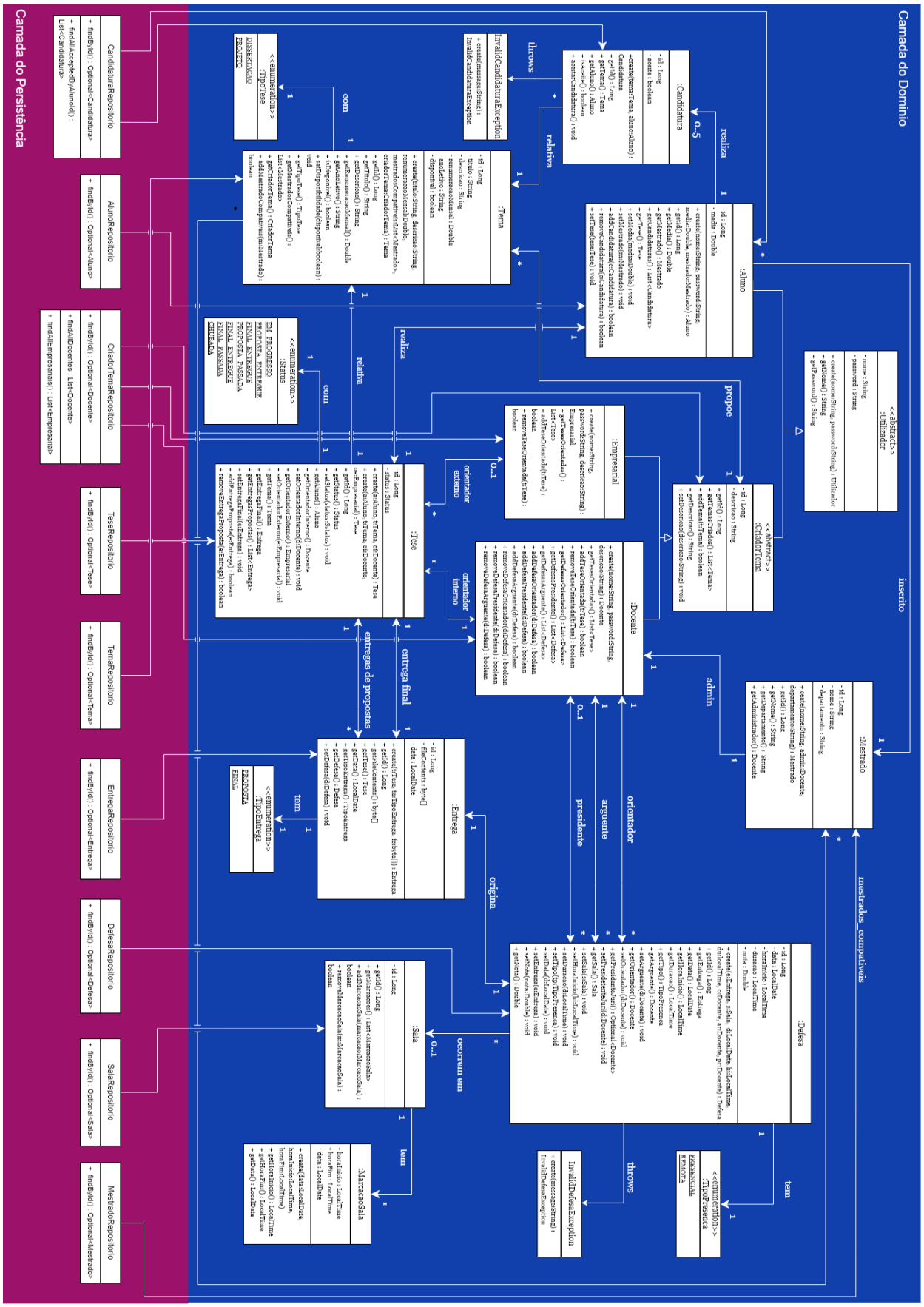


Diagrama de classes

- Optámos por não incluir quaisquer handlers nesta versão do diagrama de classes visto que esta fase ainda não tinha como objetivo a realização de qualquer caso de uso nem das respetivas interfaces de interação (tal como foi especificado no seguinte trecho do enunciado: "***Nesta entrega não se pretende ainda a realização dos casos de uso, nem nenhuma interface de interacção***")



Entidades

Utilizador

```java
@MappedSuperclass
public abstract class Utilizador {

  @NonNull private String nome;

  @NonNull private String password;
```

Na classe abstrata *Utilizador* utilizamos as anotações *@MappedSuperclass* pois esta vai ser uma superclasse para entidades JPA de modo que os seus atributos sejam mapeados para as tabelas das classes filhas.

Esta anotação vai ser útil no contexto do *Utilizador* pois todas as classes que a estendam (tipos diferentes de utilizadores na aplicação) vão compartilhar um conjunto comum de atributos (nome, password) e por isso, em vez de ter a repetição destes em cada classe, podem ser herdados da classe *Utilizador*.

Relativamente aos atributos ambos (nome e password) encontram-se anotados com *@NonNull* de modo a garantir que estes dois parâmetros existam.

Aluno

```java
@Entity
public class Aluno extends Utilizador {
  @Id
  @GeneratedValue(strategy = GenerationType.AUTO)
  private Long id;

  private Double media;

  @ManyToOne
  @JoinColumn(name = "mestrado_id", nullable = false)
  private Mestrado mestrado;

  @OneToMany(cascade = CascadeType.ALL)
  private List<Candidatura> candidaturas = new ArrayList<>();

  @OneToOne(cascade = CascadeType.ALL)
  @JoinColumn(name = "tese_id")
  private Tese tese;
```

Na classe *Aluno*, que estende *Utilizador*, usamos a anotação *@Entity* por se tratar de uma entidade na nossa aplicação.

O atributo *Id* possui a anotação *@Id*, uma vez que se trata da primary key da tabela e será gerada pela base de dados através de *@GeneratedValue*, servindo-se da estratégia *AUTO.* Esta estratégia vai deixar que o JPA escolha a estratégia a usar

O atributo *media* representa a media atual do aluno e como considerámos que num dado momento o aluno pode ainda não ter uma média atribuída que este não necessitaria da anotação *@NonNull.*

O atributo *mestrado* representa a relação entre o *Aluno* e um *Mestrado*. Possui a anotação *@ManyToOne* para indicar que muitos alunos podem estar inscritos em um único mestrado. Inclui também a anotação *@JoinColumn* para especificar qual a coluna é usada para a junção entre o *Aluno* e *Mestrado*, sendo criada a coluna *mestrado_id* na tabela *Aluno* que irá conter a foreign key do *Mestrado* (Padrão Foreign Key Mapping). Também é usado *nullable=false* para indicar que um aluno precisa sempre de estar inscrito num mestrado.

O atributo candidaturas representa a relação entre o *Aluno* e *Candidatura*. Possui a anotação *@OneToMany* para indicar que um aluno pode realizar várias candidaturas. Também é usado *cascade = CascadeType.ALL* de modo a propagar as operações realizadas sobre o *Aluno* para as *Candidatura* associadas.

O atributo *tese* representa a relação entre o *Aluno* e *Tese*. Possui a anotação *@OneToOne* para indicar que um aluno realiza uma tese num dado momento. Inclui também a anotação *@JoinColumn* para especificar qual a coluna é usada para a junção entre o *Aluno* e *Tese*, sendo criada a coluna *tese_id* na tabela *Aluno* que irá conter a foreign key da *Tese* (Padrão Foreign Key Mapping). Além disso é usado *cascade = CascadeType.ALL* de modo a propagar as operações realizadas sobre o *Aluno* para a *Tese* associada.

CriadorTema

```java
@Entity
@Inheritance(strategy = InheritanceType.SINGLE_TABLE)
public abstract class CriadorTema extends Utilizador {

  @Id
  @GeneratedValue(strategy = GenerationType.AUTO)
  private Long id;

  @OneToMany(mappedBy = "criadorTema", cascade = CascadeType.ALL)
  private List<Tema> temas_criados;

  private String descricao;
```

Na classe abstrata *CriadorTema*, que estende *Utilizador*, utilizamos as anotações *@Entity* pois esta vai ser uma entidade na qual queremos que tenha uma tabela associada. Usamos a anotação *@Inheritance* com a *strategy = InheritanceType.SINGLE_TABLE*, isto vai fazer com que todas as classes filhas vão ficar na mesma tabela *CriadorTema*, sendo criada a coluna que guarda o tipo de criador de tema (no nosso caso *Docente* e *Empresarial*). Esta anotação vai ser útil no sentido de querer guardar todas as pessoas que criaram um tema sem necessariamente saber o seu tipo.

O atributo *Id* possui a anotação *@Id*, uma vez que se trata da primary key da tabela e será gerada pela base de dados através de *@GeneratedValue*, servindo-se da estratégia *AUTO*. Esta estratégia vai deixar que o JPA escolha a estratégia a usar. Devido á estratégia de herança mencionada em cima, este Id vai ser partilhado entre as classes filhas.

O atributo *temas_criados* representa a ligação anotada por *@OneToMany* entre os *Tema*s e os *CriadorTema*, pois um único *CriadorTema* pode criar vários *Tema*, mas um *Tema* tem apenas um único *CriadorTema*. Além disso é usado *cascade = CascadeType.ALL* de modo a propagar as operações realizadas sobre o *CriadorTema* para todos *Tema*s associados àquele *CriadorTema*. Por fim usámos também o *mappedBy=criadorTema* que indica qual o lado inverso da relação.

O atributo *descrição* representa a descrição do *CriadorTema*, na qual pode ser tanto a empresa de um criador ou algo relacionado com um docente

Docente

```java
@Entity
public class Docente extends CriadorTema {

  @OneToMany(mappedBy = "orientadorInterno")
  private List<Tese> tesesOrientadas;

  @OneToMany(mappedBy = "orientador")
  private List<Defesa> defesasComoOrientador;

  @OneToMany(mappedBy = "presidente")
  private List<Defesa> defesasComoPresidente;

  @OneToMany(mappedBy = "arguente")
  private List<Defesa> defesasComoArguente;
```

Na classe *Docente*, que estende *CriadorTema*; utilizamos as anotações *@Entity* pois esta vai ser uma entidade na nossa aplicação.

O atributo *tesesOrientadas*, anotado com *@OneToMany*, representa uma ligação entre um *Docente* e várias *Tese* sendo estas aquelas nas quais o docente é orientador interno. É *OneToMany* uma vez que um único *Docente* pode orientar várias *Tese*, mas esta apenas é orientada por um único *Docente*, neste caso interno.

O atributo *defesasComoOrientador*, anotado com *@OneToMany,* representa uma ligação entre um *Docente* e várias *Defesa*, sendo estas aquelas nas quais o docente pertence no júri como orientador. É *OneToMany* uma vez que um único *Docente* pode orientar várias *Defesa*, mas uma *Defesa* apenas é orientada por um único *Docente*.

O atributo *defesasComoPresidente*, anotado com *@OneToMany*, representa uma ligação entre um *Docente* e várias *Defesa*s, sendo estas aquelas nas quais o docente pertence no júri como

presidente. É *OneToMany* uma vez que um único *Docente* pode ser presidente de várias *Defesa*, mas uma *Defesa* apenas é presidida por um único *Docente*.

O atributo defesasComoArguente, anotado com @*OneToMany*, representa uma ligação entre um *Docente* e várias *Defesa*s. sendo estas aquelas nas quais o docente pertence no júri como arguente. É *OneToMany* uma vez que um único *Docente* arguir em várias *Defesa*, mas uma *Defesa* apenas tem um *Docente* com o papel de arguente.

Em todos os atributos é usado o parâmetro mappedBy=xxx onde xxx é o nome do lado inverso da relação. Neste caso os nomes são: orientadorInterno, orientador, presidente e arguente respetivamente à ordem dos atributos acima referidos.

Empresarial

```
@Entity
public class Empresarial extends CriadorTema {

  @OneToMany(mappedBy = "orientadorExterno")
  private List<Tese> tesesOrientadas;
```

Na classe *Empresarial*, que estende *CriadorTema*, utilizamos as anotações @*Entity* pois esta vai ser uma entidade na nossa aplicação.

O atributo *tesesOrientadas*, anotado com @*OneToMany* representa uma ligação entre um *Empresarial* e várias *Tese*, sendo estas aquelas nas quais o empresarial é orientador externo. É *OneToMany* uma vez que um único *Empresarial* pode orientar várias *Tese*, mas uma *Tese* apenas é orientada por um único *Docente*. Neste caso usamos o atributo mappedBy=orientadorExterno pois um *Empresarial* vai corresponder a um orientadorExterno na classe de *Tese*.

Candidatura

```
@Entity
public class Candidatura {
  @Id
  @GeneratedValue(strategy = GenerationType.AUTO)
  private Long id;

  @ManyToOne
  @JoinColumn(name = "tema_id", nullable = false)
  private Tema tema;
```

Na classe *Candidatura* utilizamos a anotação @*Entity* pois esta vai ser uma entidade na nossa aplicação.

O atributo *Id* possui a anotação @*Id*, uma vez que se trata da primary key da tabela e será gerada pela base de dados através de @*GeneratedValue*, servindo-se da estratégia *AUTO*. Esta estratégia vai deixar que o JPA escolha a estratégia a usar.

O atributo *tema* é a ligação anotada por *@ManyToOne* entre *Candidatura* e *Tema* representado o facto de poder haver várias candidaturas respetivas a um tema. Inclui também a anotação *@JoinColumn* para especificar qual a coluna é usada para a junção entre o *Candidatura* e *Tema*, sendo criada a coluna tema_id na tabela *Candidatura* que irá conter a foreign key do *Tema* (Padrão Foreign Key Mapping). Além disso é usado nullable = false, pois, uma *Candidatura* tem de ter sempre um *Tema* associado.

Tema

```java
@Entity
public class Tema {

  @Id
  @GeneratedValue(strategy = GenerationType.AUTO)
  private Long id;

  @NonNull
  private String titulo;

  @NonNull
  private String descricao;

  private Double renumeracaoMensal;

  @NonNull
  private String anoLetivo;
```

Na classe *Tema* usámos a anotação *@Entity* pois esta vai ser uma entidade na nossa aplicação.

O atributo *Id* possui a anotação *@Id*, uma vez que se trata da primary key da tabela e será gerada pela base de dados através de *@GeneratedValue*, servindo-se da estratégia *AUTO*. Esta estratégia vai deixar que o JPA escolha a estratégia a usar.

O atributo *titulo* representa o titulo/nome de uma instancia de *Tema*, este anotado com *@NonNull* uma vez que uma tese precisa de ter obrigatoriamente um titulo.

O atributo *descricao* representa a descrição de uma *Tema*, este anotado com *@NonNull* uma vez que uma tese precisa de ter obrigatoriamente uma descrição.

O atributo *renumeracaoMensal* representa a renumeração mensal de uma *Tema*, não estando anotado com *@NonNull* pois podem existir temas que não tenham uma renumeração.

O atributo *anoLetivo* representa o ano letivo em que o *Tema* foi proposto, este anotado com *@NonNull* uma vez que um *Tema* tem sempre um ano letivo correspondente.

```java
private boolean disponivel;

@Enumerated(EnumType.STRING)
@NonNull
private TipoTese tipoTese;

@ManyToMany
private List<Mestrado> mestradosCompativeis;

@ManyToOne
@JoinColumn(name = "criadorTema_id")
private CriadorTema criadorTema;
```

O atributo *disponivel* indica se o *Tema* ainda se encontra disponível para ser atribuído (se não existe uma tese relativa a esse tema).

O atributo *tipoTese* representa o tipo da tese na qual o *Tema* corresponde, no nosso caso Dissertação ou Projeto. Tem a anotação *@Enumerated(EnumType.STRING)* que indica que este atributo é um enumerado e que vai ser tratado como string na base de dados. Tem também a anotação *@NonNull* uma vez que um tema precisa sempre de ter um tipo de tese associado.

O atributo *mestradoCompativeis*, com a anotação *@ManyToMany* representa a relação entre a compatibilidade de um *Tema* com um *Mestrado*, pois um *Tema* pode ser compatível com vários *Mestrado* assim como um *Mestrado* pode ser compatível com vários *Tema*.

O atributo *criadorTema* representa a ligação anotada por *@ManyToOne* entre os *Tema* e os *CriadorTema*, pois um único *CriadorTema* pode criar vários *Tema*, mas um *Tema* tem apenas um único *CriadorTema*. Inclui também a anotação *@JoinColumn* para especificar qual a coluna é usada para a junção entre o *CriadorTema* e *Tema*, sendo criada a coluna criador*Tema*_id na tabela *Tema* que conterá a foreign key do *CriadorTema* (Padrão Foreign Key Mapping).

Tese

```java
@Entity
public class Tese {

  @Id
  @GeneratedValue(strategy = GenerationType.AUTO)
  private Long id;

  @NonNull
  @Enumerated(EnumType.STRING)
  private Status status;

  @OneToOne(optional = false)
  @JoinColumn(name = "aluno_id", nullable = false)
  private Aluno aluno;

  @ManyToOne
  @JoinColumn(name = "orientador_interno_id", nullable = false)
  private Docente orientadorInterno;
```

Na classe *Tese* usámos a anotação *@Entity* pois esta vai ser uma entidade na nossa aplicação.

O atributo *Id* possui a anotação @*Id*, uma vez que se trata da primary key da tabela e será gerada pela base de dados através de @*GeneratedValue*, servindo-se da estratégia *AUTO*. Esta estratégia vai deixar que o JPA escolha a estratégia a usar

O atributo *status* representa o estado atual da tese, isto é, por exemplo, se esta já tem uma defesa de proposta entregue, uma defesa final entregue, etc...

Este encontra-se anotado com *Enumerated(EnumType.STRING)* que indica que este atributo será guardado na tabela como uma string. Tem também a anotação @*NonNull* uma vez que uma tese precisa sempre de ter um status associado.

O atributo *aluno*, com a anotação *OneToOne(optional=false)* representa a relação entre *Tese* e *Aluno*, pois uma *Tese* é sempre realizado por apenas um *Aluno* e este encontra-se num dado momento sempre a realizar no máximo uma *Tese*. Também tem a anotação @*JoinColumn* para especificar que deverá ser criada uma coluna *aluno_id* que conterá o id do aluno como foreign key (Padrão Foreign Key Mapping). Também se encontra especificado *nullable* = *false* pois uma tese tem de ter sempre um aluno a realizá-la.

O atributo *orientadorInterno* representa a ligação anotada por @@*ManyToOne* entre as *Tese* e os *Docente*, pois uma *Tese* tem apenas um *Docente* como orientador interno e este pode estar a orientar várias *Tese*. Inclui também a anotação @*JoinColumn* para especificar qual a coluna é usada para a junção entre o *Docente* e *Tese*, sendo criada a coluna orientador_interno_id na tabela *Tese* que conterá a foreign key (Padrão Foreign Key Mapping). O atributo nullable = false indica que uma *Tese* tem sempre um *Docente* como orientador interno.

```
@ManyToOne
@JoinColumn(name = "orientador_externo_id")
private Empresarial orientadorExterno;

@OneToOne
@JoinColumn(name = "tema_id", nullable = false)
private Tema tema;

@OneToOne(cascade = CascadeType.ALL)
@JoinColumn(name = "entrega_final_id")
private Entrega entregaFinal;

@OneToMany(mappedBy = "tese", cascade = CascadeType.ALL)
private List<Entrega> entregasPropostas;
```

O atributo *orientadorExterno* representa a ligação anotada por @@*ManyToOne* entre as *Tese* e os *Docente*, pois uma *Tese* pode ter um *Empresarial* como orientador externo e este pode estar a orientar várias *Tese*. Inclui também a anotação @*JoinColumn* para especificar qual a coluna é usada para a junção entre o *Empresarial* e *Tese*, sendo criada a coluna orientador_externo_id na tabela *Tese* que conterá a foreign key (Padrão Foreign Key Mapping). A não inclusão de *nullable* = *false* deve-se a uma tese não ter um orientador externo caso se trate de um tema de dissertação.

O atributo *tema*, com a anotação @*OneToOne* representa a relação entre *Tese* e *Tema*, pois uma *Tese* é sempre relativa a um *Tema* e este no máximo está associado a uma *Tese*.

Também tem a anotação *@JoinColumn* para especificar que deverá ser criada uma coluna tema_id que conterá o id do tema como foreign key (Padrão Foreign Key Mapping). Também se encontra especificado nullable = false pois uma tese tem de ter sempre um tema associado.

O atributo *entregaFinal*, com a anotação *OneToOne* representa uma das relações entre *Tese* e *Entrega*, pois uma *Tese* terá sempre no máximo uma *Entrega* que é final e esta é relativa a apenas uma *Tese*. O uso de *cascade = CascadeType.ALL* deve-se a queremos propagar alterações na *Tese* para a *Entrega* (por exemplo, ao deletar uma tese nao faria sentido manter a entrega associada). Também tem a anotação *@JoinColumn* para especificar que deverá ser criada uma coluna entrega_final_id que conterá o id da entrega como foreign key (Padrão Foreign Key Mapping). Também se encontra especificado nullable = false pois uma tese tem de ter sempre um tema associado.

O atributo *entregasPropostas* representa uma das relações entre *Tese* e *Entrega*. Possui a anotação *@OneToMany* para indicar que uma *Tese* pode realizar várias *Entrega* relativas a propostas de defesa. Também é usado *cascade = CascadeType.ALL* de modo a propagar as operações realizadas sobre o *Tese* para as *Entrega* associadas. Por fim usámos também o *mappedBy=tese* que indica qual o lado inverso da relação.

Entrega

```java
@Entity
public class Entrega {
    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    private Long id;

    @Lob
    @NonNull
    private byte[] fileContents;

    @Enumerated(EnumType.STRING)
    private TipoEntrega tipoEntrega;

    @Column(columnDefinition = "DATE")
    private LocalDate data;

    @ManyToOne
    @JoinColumn(name = "tese_id", nullable = false)
    private Tese tese;

    @OneToOne
    private Defesa defesa;
```

Na classe *Entrega* usámos a anotação *@Entity* pois esta vai ser uma entidade na nossa aplicação.

O atributo *Id* possui a anotação *@Id*, uma vez que se trata da primary key da tabela e será gerada pela base de dados através de *@GeneratedValue*, servindo-se da estratégia *AUTO*. Esta estratégia vai deixar que o JPA escolha a estratégia a usar

O atributo *fileContents* é um array de bytes que representa o conteúdo do ficheiro. Encontra-se anotado com *@Lob* pois este corresponde a um ficheiro binário com tamanho possivelmente grande que pretendemos persistir.

O atributo *tipoEntrega* é um enumerado que indica o tipo da entrega, isto é, se uma entrega é uma proposta de entrega ou uma entrega final. Este encontra-se anotado com *@Enumerated(EnumType.STRING)* que indica que este atributo será guardado na tabela como uma string.

O atributo *data* indica o dia em que a *Entrega* foi realizada. Este foi anotado com *@Column*(*columnDefinition* = *DATE*) de modo a podermos guardar este atributo numa coluna com o tipo date.

O atributo *tese*, com a anotação *@ManyToOne* representa a relação entre *Entrega* e *Tese*, pois uma *Tese* pode ter várias *Entrega* associadas e esta está associado a apenas uma *Tese*. Também tem a anotação *@JoinColumn* para especificar que deverá ser criada uma coluna tese_id que conterá o id do tema como foreign key (Padrão Foreign Key Mapping). Também se encontra especificado nullable = false, pois, uma entrega tem de ter sempre uma tese associada.

O atributo *defesa*, com a anotação *@OneToOne* representa a relação entre *Defesa* e *Entrega*, pois uma *Defesa* é sempre relativa a uma *Entrega* e esta origina uma única *Defesa*.

Sala

```java
@Entity
public class Sala {

  @Id
  @GeneratedValue(strategy = GenerationType.AUTO)
  private Long id;

  @ElementCollection private List<MarcacaoSala> marcacoes;
```

Na classe *Sala* usámos a anotação *@Entity* pois esta vai ser uma entidade na nossa aplicação.

O atributo *Id* possui a anotação *@Id*, uma vez que se trata da primary key da tabela e será gerada pela base de dados através de *@GeneratedValue*, servindo-se da estratégia *AUTO*.

O atributo *marcacoes* representa a lista de *MarcacaoSala*. Esta tem a anotação *@ElementCollection* de modo a criar uma tabela para que inclua os atributos de *MarcacaoSala* e do id *Sala* como foreign key (Padrão Foreign Key Mapping). Isso permite o mapeamento de marcações a uma sala especifica.

Defesa

```java
@Entity
public class Defesa {
  @Id
  @GeneratedValue(strategy = GenerationType.AUTO)
  private Long id;

  @NonNull
  @Column(columnDefinition = "DATE")
  private LocalDate data;

  @NonNull
  @Column(columnDefinition = "TIME")
  private LocalTime horaInicio;

  @NonNull
  @Column(columnDefinition = "TIME")
  private LocalTime duracao;

  @NonNull
  @Enumerated(EnumType.STRING)
  private TipoPresenca tipo;
```

Na classe *Defesa* usámos a anotação *@Entity* pois esta vai ser uma entidade na nossa aplicação.

O atributo *Id* possui a anotação *@Id*, uma vez que se trata da primary key da tabela e será gerada pela base de dados através de *@GeneratedValue*, servindo-se da estratégia *AUTO*. Esta estratégia vai deixar que o JPA escolha a estratégia a usar.

O atributo *data* indica se o dia em que a *Defesa* acontecerá. Este foi anotado com *@NonNull* pois uma defesa ocorre sempre numa certa data e *@Column*(columnDefinition = date) de modo a podermos guardar este atributo numa coluna com o tipo date.

O atributo *horaInicio* indica se a hora de inicio da *Defesa*. Este foi anotado com *@NonNull* pois uma defesa tem sempre uma hora de inicio e *@Column*(columnDefinition = Time) de modo a podermos guardar este atributo numa coluna com o tipo TIME.

O atributo *duracao* indica se a duração da *Defesa*. Este foi anotado com *@NonNull* pois uma defesa tem sempre uma duraçãon e *@Column*(columnDefinition = Time) de modo a podermos guardar este atributo numa coluna com o tipo TIME.

O atributo *tipo* representa como será realizada a *Defesa*, ou seja, se esta acontecerá presencialmente ou remotamente. Este encontra-se anotado com *@Enumerated(EnumType.STRING)* que indica que este atributo será guardado na tabela como uma string. Tem também a anotação *@NonNull* uma vez que uma defesa precisa sempre de ter um status associado.

```java
@ManyToOne
@JoinColumn(name = "orientador_id", nullable = false)
private Docente orientador;

@ManyToOne
@JoinColumn(name = "arguente_id", nullable = false)
private Docente arguente;

@ManyToOne
@JoinColumn(name = "presidente_id")
private Docente presidente;

@ManyToOne
@JoinColumn(name = "sala_id")
private Sala sala;

@OneToOne(mappedBy = "defesa")
private Entrega entrega;

private Double nota;
```

O atributo *orientador* representa uma das relações entre *Defesa* e um *Docente*. Possui a anotação *@ManyToOne* para indicar que um *Docente* pode estar a participar como orientador em várias *Defesa* e esta apenas pode ter um *Docente* como orientador. Inclui também a anotação *@JoinColumn* para especificar qual a coluna é usada para a junção entre o *Docente* e *Defesa*, sendo criada a coluna orientador_id na tabela *Defesa* que conterá a foreign key (Padrão Foreign Key Mapping). Também é usado nullable=false para indicar que uma defesa precisa sempre de ter um orientador no júri.

O atributo *arguente* representa uma da s relações entre *Defesa* e um *Docente*. Possui a anotação *@ManyToOne* para indicar que um *Docente* pode estar a participar como arguente em várias *Defesa* e esta apenas pode ter um *Docente* como arguente. Inclui também a anotação *@JoinColumn* para especificar qual a coluna é usada para a junção entre o *Docente* e *Defesa*, sendo criada a coluna orientador_id na tabela *Defesa*. Também é usado *nullable=false* para indicar que uma defesa precisa sempre de ter um arguente no júri.

O atributo *presidente* representa uma das relações entre *Defesa* e um *Docente*. Possui a anotação *@ManyToOne* para indicar que um *Docente* pode estar a participar como presidente em várias *Defesa* e esta apenas pode ter um *Docente* como presidente. Inclui também a anotação *@JoinColumn* para especificar qual a coluna é usada para a junção entre o *Docente* e *Defesa*, sendo criada a coluna orientador_id na tabela *Defesa* que conterá a foreign key do *Docente* (Padrão Foreign Key Mapping). Note-se o não uso de *nullable=false* para indicar que uma defesa não precisa sempre de ter um presidente no júri (no nosso caso apenas uma defesa final precisa de ter presidente de júri enquanto uma defesa de proposta de tese pode apenas ter orientador e arguente).

O atributo *sala* representa uma das relações entre *Defesa* e uma *Sala*. Possui a anotação *@ManyToOne* para indicar que uma *Sala* pode alojar várias *Defesa* e esta apenas pode ocorrer numa *Sala*. Inclui também a anotação *@JoinColumn* para especificar qual a coluna é usada para a junção entre a *Sala* e *Defesa*, sendo criada a coluna sala_id na tabela *Defesa* que conterá a foreign key da *Sala* (Padrão Foreign Key Mapping). Note-se o não uso de *nullable=false* para indicar que

uma defesa não precisa sempre de ter um presidente no júri (no caso de uma defesa ser remota não existe a necessidade de ter uma sala associada).

O atributo *entrega* representa uma das relações entre *Defesa* e uma *Sala*. Possui a anotação *@OneToOne* para indicar que uma *Defesa* é relativa a uma *Entrega* que foi feita e que esta apenas originará uma única *Defesa*. Inclui também *mappedBy = defesa* para indicar qual o lado inverso da relação.

O atributo *nota* é usado para representar a nota dada á defesa após esta ocorrer.

MarcacaoSala

```java
@Embeddable
public class MarcacaoSala {

  @NonNull
  @Column(name = "hoaraInicio", columnDefinition = "TIME")
  private LocalTime horaInicio;

  @NonNull
  @Column(name = "horaFim", columnDefinition = "TIME")
  private LocalTime horaFim;

  @NonNull
  @Column(name = "data", columnDefinition = "DATE")
  private LocalDate data;
```

A classe *MarcacaoSala* representa uma ocupação de uma sala numa determinada data e hora.

Esta é anotada com @Embeddable para poder ser guardada com uma parte intrínseca de classe *Sala* como já foi referido anteriormente.

Esta guarda um atributo *horaIncio* que tem a anotação *@Column* com o parâmetro *columnDefinition=Time* pois este representa a hora que a sala começa a marcação respetiva.

Guarda também o atributo *horaFim* que vai ter a anotação *@Column* com o parâmetro *columnDefinition=Time* pois este representa a hora que a sala acaba com a marcação respetiva.

Finalmente, guarda também um atributo *data* que vai ter a anotação *@Column* com o parâmetro *columnDefinition=Date* pois este representa a data da marcação.

Todos estes parâmetros estão anotados com *@NonNull*.

Mestrado

```java
@Entity
public class Mestrado {

  @Id
  @GeneratedValue(strategy = GenerationType.AUTO)
  private Long id;

  @NonNull
  @Column(unique = true)
  private String nome;

  @NonNull private String departamento;

  @OneToOne
  @JoinColumn(name = "docente_id", nullable = false)
  private Docente administrador;
```

Na classe *Mestrado* usamos a notação *@Entity* por se tratar de uma entidade da nossa implementação.

O atributo *Id* possui a anotação *@Id*, uma vez que se trata da primary key da tabela e será gerada pela base de dados através de *@GeneratedValue*, servindo-se da estratégia *AUTO*. Esta estratégia vai deixar que o JPA escolha a estratégia a usar.

O atributo *nome* representa o nome do mestrado e tem as anotações *@NonNull* e Column com o atributo *unique=True* pois queremos que este nome seja único

O atributo *departamento* representa o departamento na qual o mestrado está enquadrado. Foi anotado com *@NonNull* pois este parâmetro tem de existir obrigatoriamente.

O atributo *administrador* representa uma ligação entre um *Mestrado* e um *Docente* que é o administrador/coordenador do *Mestrado*. Este tem a anotação *@OneToOne* pois a ligação representa o facto de um único docente só poder administrar um único mestrado e um mestrado só poder ter um administrador/coordenador.

Inclui também a anotação *@JoinColumn* para especificar qual a coluna é usada para a junção entre o *Mestrado* e *Docente*, sendo criada a coluna docente_id na tabela *Docente* que conterá a foreign key do *Docente*(Padrão Foreign Key Mapping). Usamos também o parâmetro nullable=false pois queremos que haja sempre um Coordenador/Administrador de *Mestrado*.