



UNIVERSIDADE LUSÓFONA
de Humanidades e Tecnologias
Humani nihil alienum

Relatório de Trabalho Final de Curso
do Curso de
Licenciatura em Engenharia Informática (LEI)
Ano Lectivo 2011/2012

- Controlo de Acessos -

Projecto realizado sob a orientação de:
Prof. Pedro Malta
Prof. Luís Gomes

Relatório de Trabalho Final de Curso
do Curso de
Licenciatura em Engenharia Informática (LEI)
Ano Lectivo 2011/2012

- Controlo de Acessos -

Trabalho realizado por:
André Teles nº a20091051
Pedro Gonçalves nºa20091971

Índice

1. Introdução	5
1.1 Objectivo do Trabalho	5
1.2 Descrição do Problema	5
2. Use Cases – Controlos de Acesso	6
2.1 Use Case nº 1	6
2.2 Use Case nº 2	6
2.3 Use Case nº 2 Alternativo	7
3. Diagramas	8
3.1 Diagramas de Use Cases	8
3.1.1 Use Case nº1	8
3.1.2 Use Case nº2	10
3.1.3 Use Case nº2 Alternativo	12
3.2 Diagramas de Actividade	14
3.2.1 Use Case nº1	14
3.2.2 Use Case nº2	15
3.2.3 Use Case nº2 (continuação)	16
3.3 Diagramas de Estado	17
3.3.1 Diagramas de Estados de Presença	17
3.3.2 Diagramas de Estados de Marcação Nova Presença	18
3.4 Diagramas de Sequência	19
3.4.1 Use Case nº1	19
3.4.2 Use Case nº2	20
3.4.3 Use Case nº2 Alternativo	22
3.5 Diagrama de Classes	24
4. Testes Efectuados	26
5. Conclusão	27

6. Bibliografia	28
6.1 Urls	28
6.2 Referências	28
7. Anexos	29
7.1 Código Fonte da Aplicação	29

Índice de Figuras

Figura 1: Diagrama de Use Cases, Use Case nº1	7
Figura 2: Diagrama de Use Cases, Use Case nº2	9
Figura 3: Diagrama de Use Cases, Use Case nº2 Alternativo	11
Figura 4: Diagrama de Actividade do Use Case nº1	13
Figura 5: Diagrama de Actividade do Use Case nº2	14
Figura 6: Diagrama de Actividade do Use Case nº2 (continuação)	15
Figura 7: Diagrama de Estados de Presença	16
Figura 8: Diagrama de Estados de Marcação Nova Presença	17
Figura 9: Diagrama de Sequência do Use Case nº1	18
Figura 10: Diagrama de Sequência do Use Case nº2	19
Figura 11: Diagrama de Sequência do Use Case nº2 Alternativo	21
Figura 12: Diagrama de Classes	23
Figura 13: Grelha de Testes	26

1. Introdução

O Trabalho Final de Curso representa o culminar de três anos de aprendizagem que a Licenciatura em Engenharia Informática nos proporcionou ao longo da faculdade, como tal, o grupo escolheu o tema de Controlo de Acessos sugerido na listagem de TFC's para este ano lectivo. O tema tem a particularidade de ser aplicado à fundação CEBI que presta a sua ajuda aos mais desfavorecidos na nossa sociedade. Neste trabalho é implementada uma aplicação que gere os acessos da fundação, controlando assim entradas e saídas de alunos e encarregados de educação. Esta vai estar disponível nas portarias e salas de aulas, mostrando como forma de ajuda os nomes, fotos e informação relevante para o controlo que é desejado.

Esperemos que gostem do trabalho desenvolvido!

1.1 Objectivo do Trabalho

Este trabalho tem como objectivo o desenvolvimento de uma aplicação que permite à escola controlar os acessos dos seus alunos e encarregados de educação. Podemos afirmar que o está dividido em duas partes significativas, o controlo pelas portarias e o controlo nas salas de aulas.

A aplicação foi desenvolvida utilizando a linguagem de programação C# e como base dados o SQL Server.

1.2 Descrição do Problema

Com o desenvolvimento deste projecto é criada uma solução que lê o cartão de aluno ou o cartão do(s) encarregado(s) de educação (para alunos com idade inferior a cinco anos) para obter o número associado à pessoa através de um leitor de cartões e também gere as respectivas entradas e saídas. É importante informar que não vai ser utilizado nenhum cartão de aluno ou leitor, apenas é simulado o seu uso na aplicação.

As situações que se podem revelar problemáticas começam logo na utilização do cartão. Apesar de ser obrigatória, tem como solução na entrada e saída a introdução do número do cartão na portaria, desta forma não vai impossibilitar os acessos ao recinto escolar mas sim ajuda a ter outra alternativa para quando existe o esquecimento ou perda do mesmo. Além disso é acrescentado na base de dados uma

hora de saída obrigatória, sendo ela às 22 horas. A esta situação acresce o facto de ser necessário ficar registado as presenças e ausências dos alunos. Para isso os docentes terão de registar se o aluno está presente ou ausente da(s) aula(s), para ajudar, no ecrã têm a lista de alunos que fazem parte da turma, a sua foto e os respectivos encarregados de educação e ainda informação que possa ser pertinente.

Para melhor compreendermos melhor as situações que vão surgir, em baixo estão descritos os Use Cases.

2. Use Cases – Controlos de Acesso

2.1 Use Case nº1

O aluno chega à escola e **passa** o cartão pelo leitor. O leitor **recolhe** o número do cartão do aluno que está a entrar na escola e **confirma** a sua presença diária. No final do horário escolar o aluno **passa** novamente o cartão pelo leitor e este **recolhe** o número do cartão do aluno sabendo assim que o aluno saiu da escola. Caso esta situação não aconteça, no próprio dia até a escola fechar é **adicionado** automaticamente nos registos do aluno a sua saída às 22:00 horas, hora pré-definida para este tipo de situações.

Nota: O aluno neste use case tem mais de 5 anos pelo que não necessita obrigatoriamente do Encarregado de Educação (EE).

2.2 Use Case nº2

O Encarregado de Educação (EE) chega à escola e **passa** o cartão pelo leitor. O leitor **recolhe** o número do cartão do(s) aluno(s) que **pode(m) ser** filho(s) e/ou aluno(s) que **está(estão)** ao encargo deste EE. O leitor **coloca** a informação recolhida em "stand-by". É então iniciado o tempo de alarme.

Após o(s) aluno(s) chegar(em) à sala de aula acompanhado pelo seu EE a(s) professora(s) **confirma(m)** que este(s) encontram-se na sala de aula, é então **confirmada a presença** do(s) aluno(s) na escola. Caso esta situação não aconteça dentro de um tempo de alarme, o(s) aluno(s) **fica(m) numa lista de alerta** em que existe um tempo máximo para ser considerada falta escolar. Se esse tempo máximo for atingido então o(s) aluno(s) **fica(m)** com o estado de falta, caso seja **confirmada** a(s) presença(s) do(s) aluno(s) no tempo de alarme, é então considerado que o(s)

aluno(s) **está(estão)** presente(s) na aula.

No final do dia de aulas o EE **passa** novamente o cartão pelo leitor e este **recolhe** o número do cartão à saída. Caso esta situação não aconteça, no próprio dia até a escola fechar é **adicionado** automaticamente no(s) registo(s) do(s) aluno(s) a(s) sua(s) saída(s) às 22 horas, hora pré-definida para este tipo de situação.

Nota: O(s) aluno(s) neste use case tem menos de 5 anos pelo que necessita(m) obrigatoriamente do Encarregado de Educação (EE).

2.3 Use Case nº2 Alternativo

Pode(m) existir vários EE com **o(s) mesmo(s) aluno(s) associado(s)** a um cartão. Sendo que no final do horário escolar, um dos EE pode não levar um do(s) aluno(s) que lhe está encarregado. Na **saída** da sala de aula a professora deve de **confirmar** no sistema qual(uais) o(s) aluno(s) que sai(saem) com o EE. É iniciado o tempo de alarme. Assim, quando o EE **passa** o cartão pelo leitor à **saída**, no registo fica a informação que aluno(s) **saiu(saíram)** da escola.

Caso esta situação não aconteça, ou seja, o EE não saia no tempo limite ou não passe o cartão, o estado fica de **presença**.

No próprio dia até a escola fechar é **adicionado** automaticamente no(s) registo(s) do(s) aluno(s) a(s) sua(s) saída(s) às 22 horas, hora pré-definida para este tipo de situação.

Nota: O(s) aluno(s) neste use case tem menos de 5 anos pelo que necessita(m) obrigatoriamente do Encarregado de Educação (EE).

3. Diagramas

3.1 Diagrama de Use Cases

3.1.1 Use Case nº1

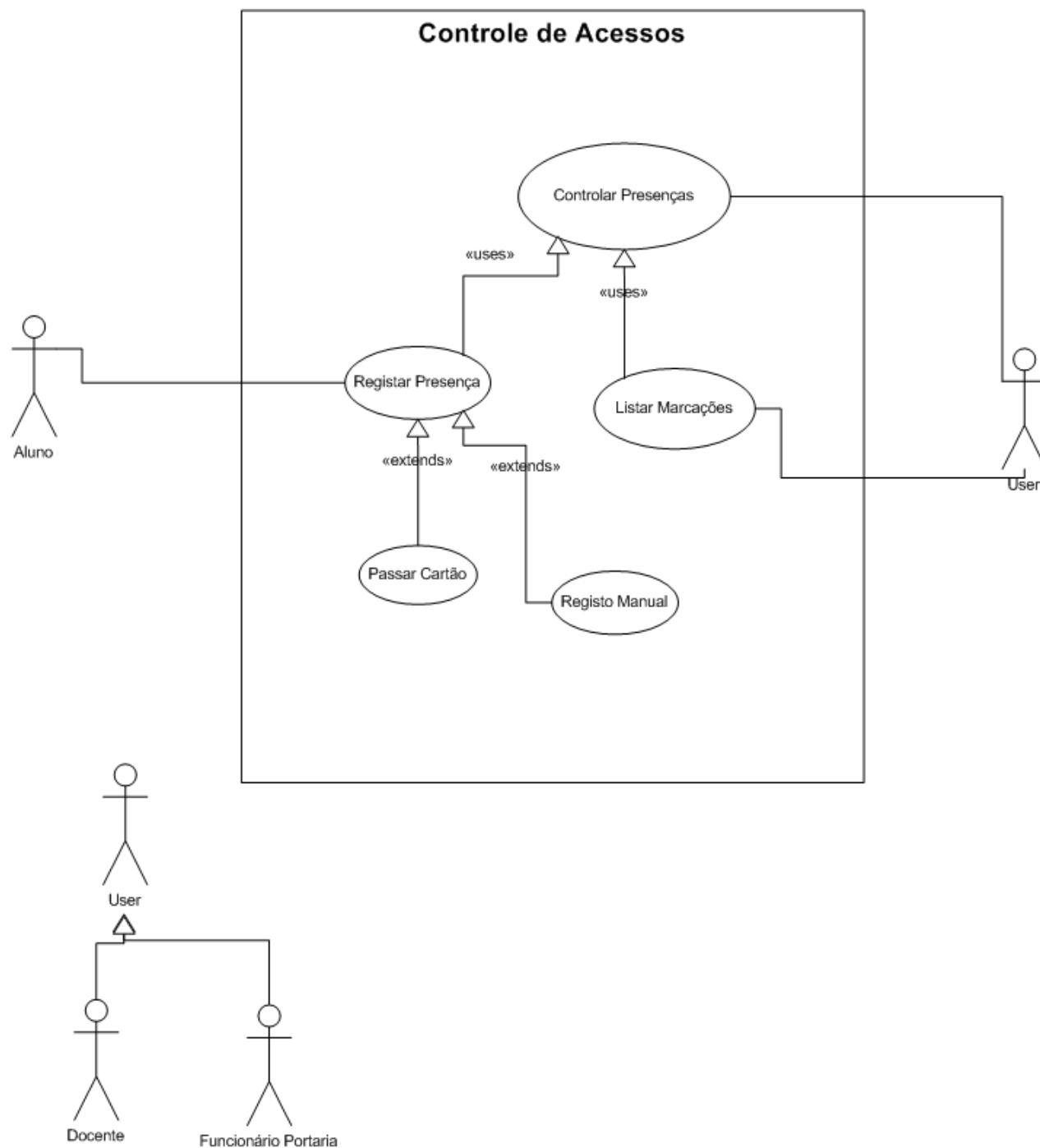


Figura 1: Diagrama de Use Cases, Use Case nº1

Na identificação dos actores existem dois actores que interagem com o sistema, são eles Aluno e User.

User – é uma generalização, pois tanto o Docente como Funcionário estão encarregues de tarefas semelhantes.

Aluno – Pessoa responsável por passar o cartão na entrada e saída da escola.

Docente – Pessoa responsável por controlar as presenças e ausências diárias dos alunos, utiliza também como ajuda uma listagem para o efeito.

Funcionário Portaria – Pessoa responsável por registar os cartões e controlar as entradas e saídas.

Descrições de Acções:

No use case Registrar Presença é definido um «extends» Passar Cartão e Registo Manual.

Registrar a presença do aluno pode ser feita através da passagem do cartão ou através do registo manual inserindo o nº do cartão.

Controlar Presenças vai ser do domínio do User, os «user» são utilizados para demonstrar que o controlo de presenças pode ser efectuado com a ajuda de Listar Marcações e através do use case Registrar Presença, que vai utilizar a passagem do cartão ou registo manual.

3.1.2 Use Case nº2

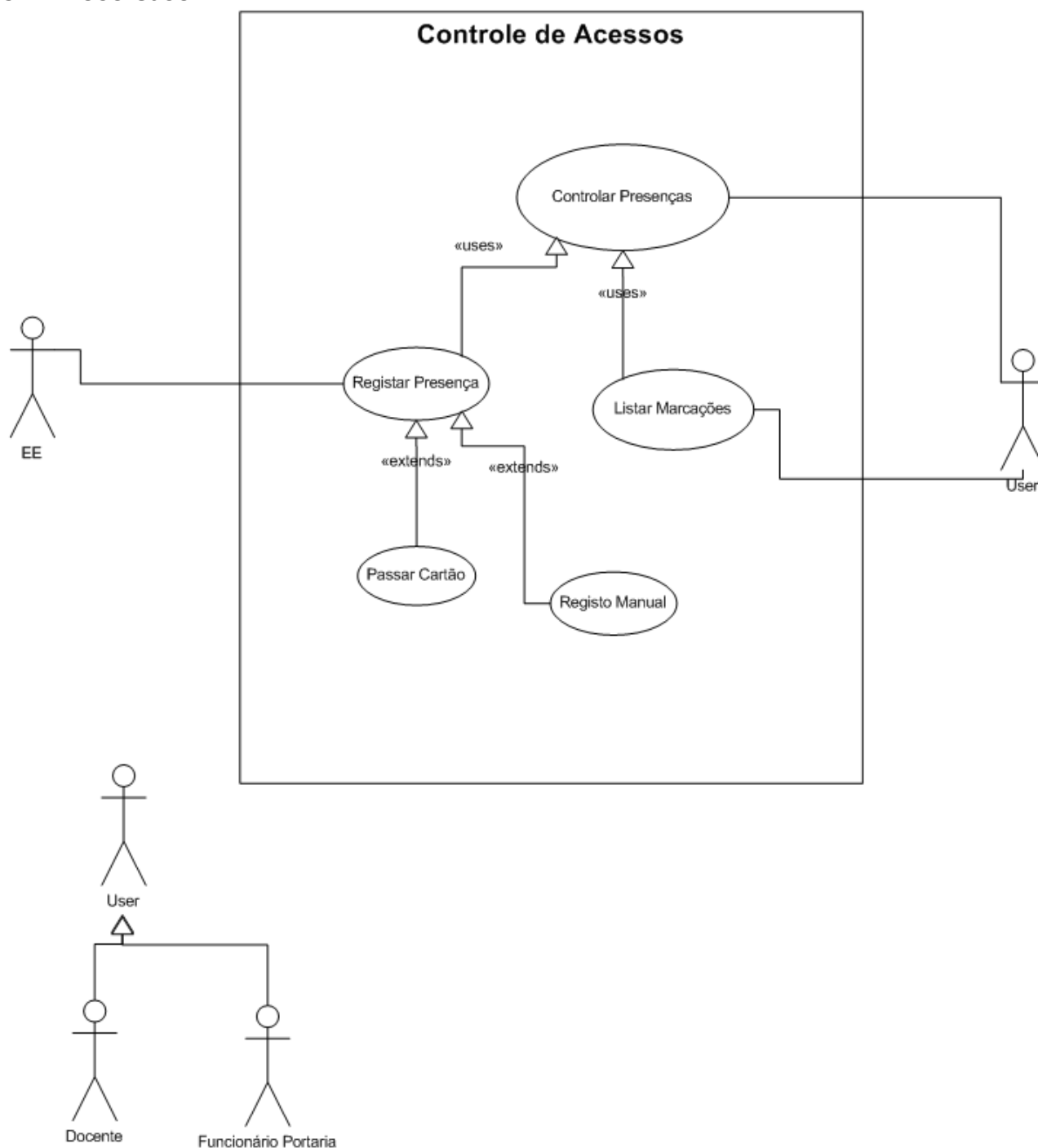


Figura 2: Diagrama de Use Cases, Use Case nº2

Na identificação dos actores existem dois actores que interagem com o sistema, são eles EE e User.

User - é uma generalização, pois tanto o Docente como Funcionário estão encarregues de tarefas semelhantes.

EE – Pessoa responsável pelo aluno, portanto é responsável por passar o cartão na entrada e saída da escola.

Docente – Pessoa responsável por controlar as presenças e ausências diárias dos alunos, utiliza também como ajuda uma listagem para o efeito.

Funcionário Portaria – Pessoa responsável por registar os cartões e controlar as entradas e saídas.

Descrições de Acções:

No use case Registrar Presença é definido uma extensão Passar Cartão e Registo Manual.

Registrar a presença do aluno, desta vez a semelhança do use case anterior, é feito pelo EE e pode ser feita através da passagem do cartão ou através do registo manual inserindo o nº do cartão.

Controlar Presenças vai ser do domínio do User, os «user» são utilizados para demonstrar que o controlo de presenças pode ser efectuado com a ajuda de Listar Marcações e através do use case Registrar Presença, que vai utilizar a passagem do cartão ou registo manual.

O user, podendo ser o docente ou o funcionário da portaria, vai controlar as presenças do aluno, para isso vai utilizar como auxílio uma listagem de quem entrou ou saiu da escola e também pode ser feito através da passagem de cartão ou do registo manual.

3.1.3 Use Case nº2 Alternativo

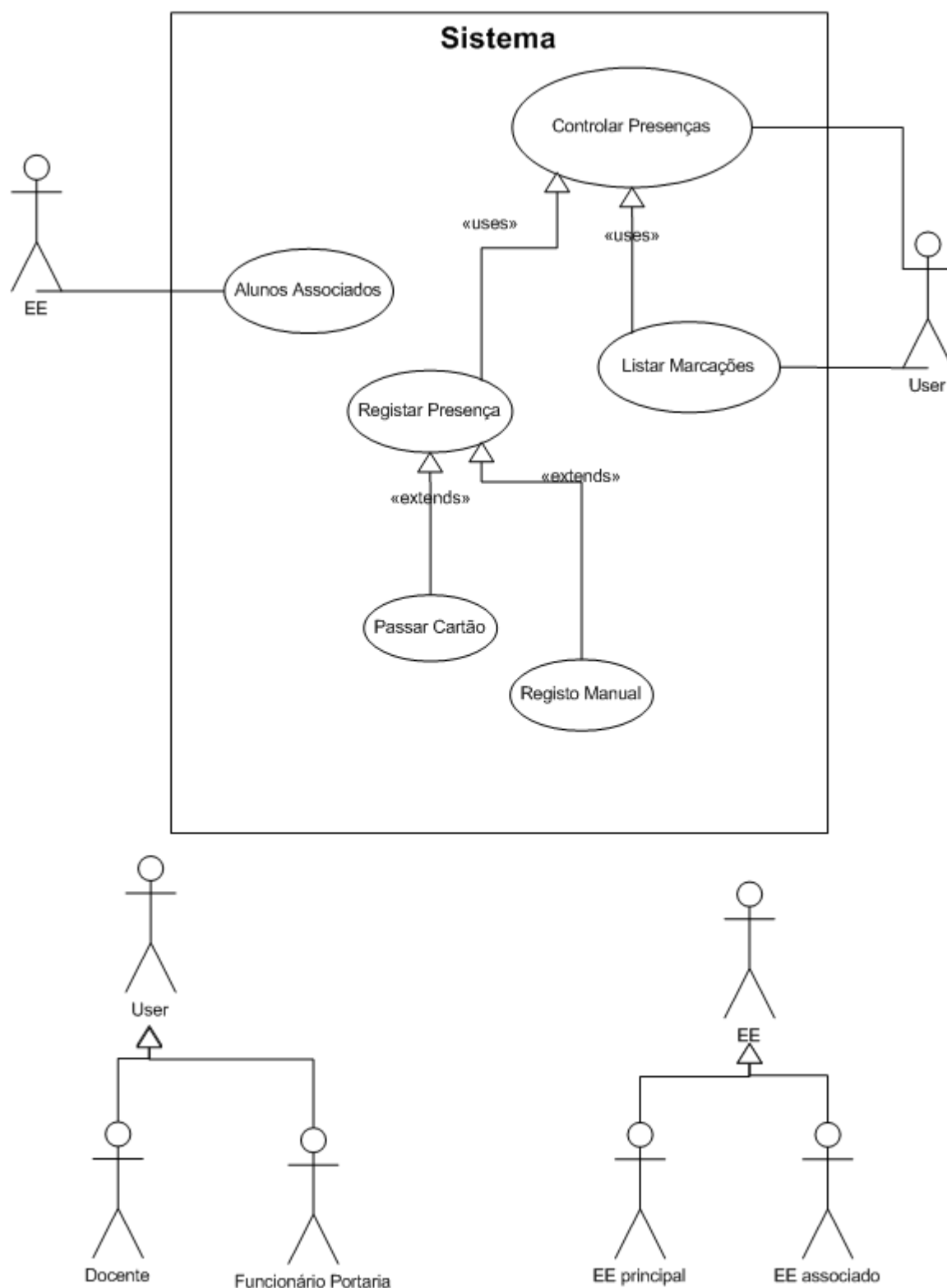


Figura 3: Diagrama de Use Cases, Use Case nº2 Alternativo

Na identificação dos actores existem dois actores que interagem com o sistema, são eles EE e User.

User – é uma generalização, pois tanto o Docente como Funcionário estão encarregues de tarefas semelhantes.

EE – é uma generalização, pois são pessoas responsáveis pelo alunos, portanto são responsável por passar o cartão na entrada e saída da escola e por coordenar quem leva qual aluno.

Descrições de Acções:

Para melhor distinção neste caso específico dividimos o EE principal como alguém que é do agregado familiar e EE associado alguém que foi depois associado ao aluno como seu EE.

Novamente no use case Registrar Presença é definido uma extensão Passar Cartão e Registo Manual.

Registrar a presença do aluno, é feito pelo User e pode ser feita através da passagem do cartão ou através do registo manual inserindo o nº do cartão.

Controlar Presenças vai ser do domínio do User, os «user» são utilizados para demonstrar que o controlo de presenças pode ser efectuado com a ajuda de Listar Marcações e através do use case Registrar Presença, que vai utilizar a passagem do cartão ou registo manual.

O user, podendo ser o docente ou o funcionário da portaria, vai controlar as presenças do aluno, para isso vai utilizar como auxílio uma listagem de quem entrou ou saiu da escola e também pode ser feito através da passagem de cartão ou do registo manual.

No use case Alunos Associados tem como pré-condição que os EE estejam associados aos alunos. Pois na saída o user, neste caso a professora, vai escolher que EE leva o aluno.

3.2 Diagramas de Actividade

3.2.1 Use Case nº1

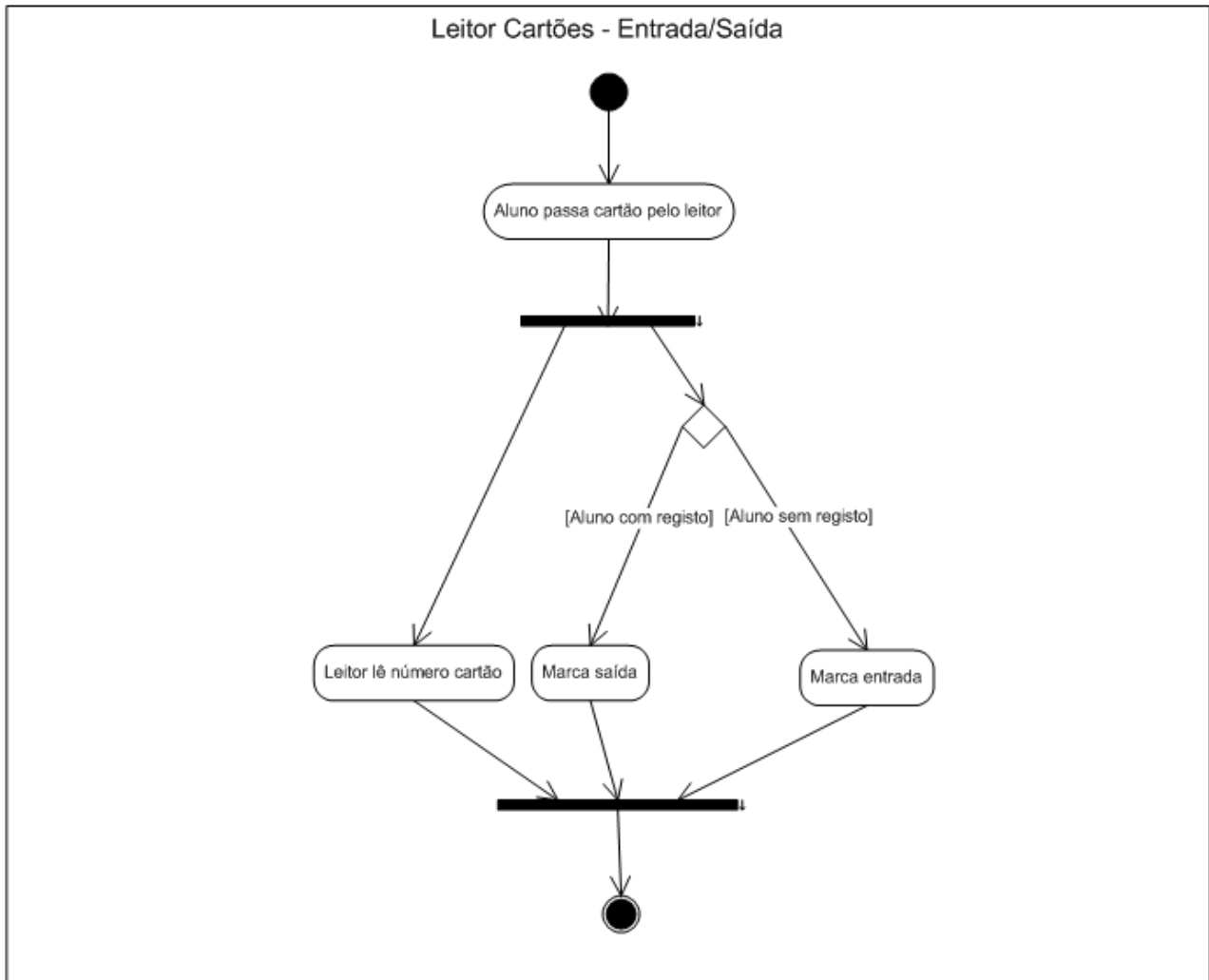


Figura 4: Diagrama de Actividade do Use Case nº1

No Use Case nº1 o Aluno vai passar o cartão pelo leitor, este vai ler o número do cartão e confirmar a presença ou ausência do aluno na escola. Portanto se o aluno já tiver feito um registo quer dizer que já entrou logo marca saída. O aluno está sem registo, então vai marcar uma entrada pois este ainda não entrou.

3.2.2 Use Case nº2

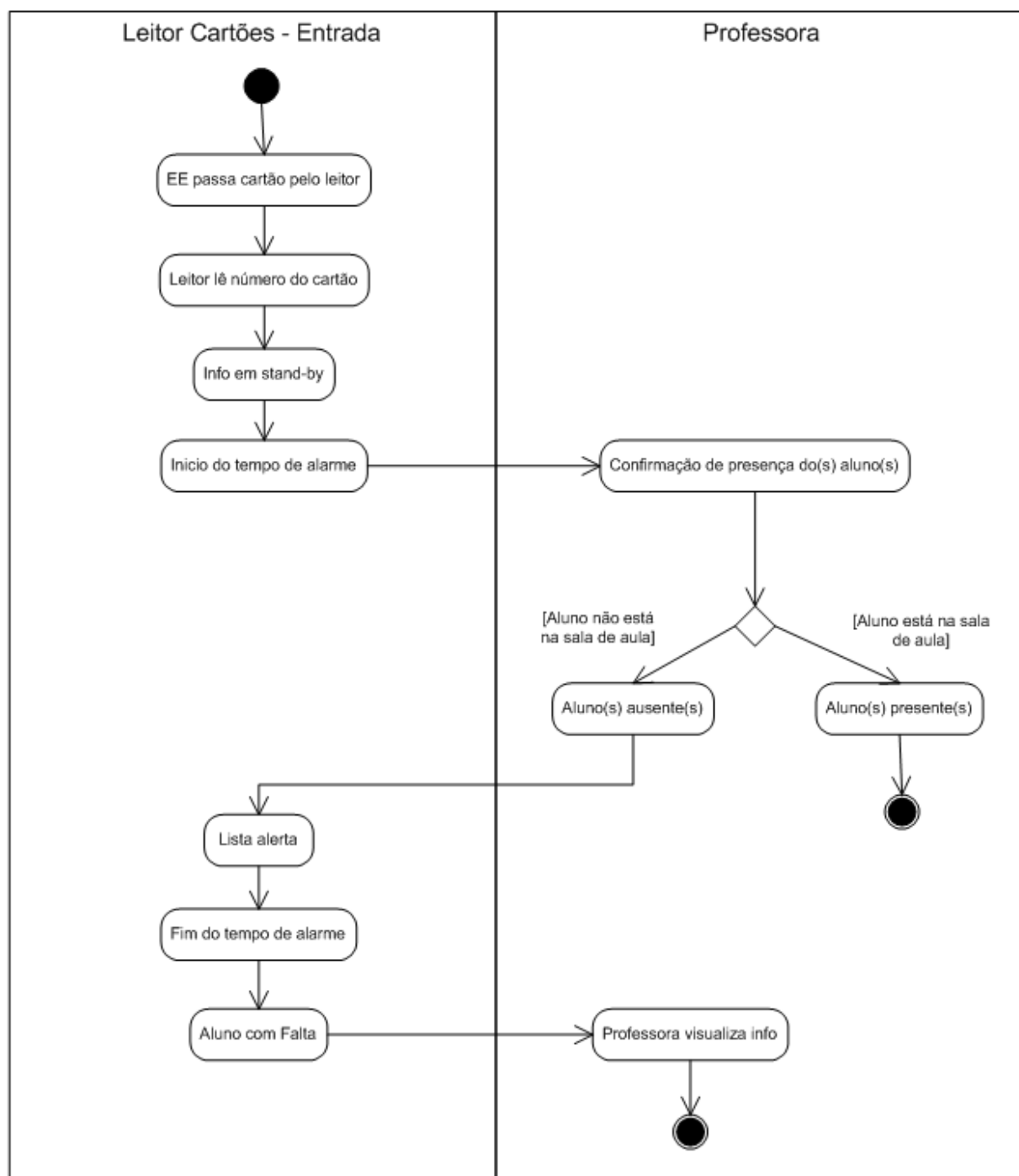


Figura 5: Diagrama de Actividade do Use Case nº2

No Use Case nº2 o EE para entrar na escola terá de passar o cartão pelo leitor onde depois é iniciado um tempo de alarme para controlar a chegada do EE e/ou Aluno à sala de aula. Estando o aluno na sala de aula dentro do tempo de alerta então será considerada uma presença, caso isto não acontece vai ser considerado ausente e colocado numa lista de alerta que por sua vez vai ter um tempo definido para ser considerada falta.

3.2.3 Use Case nº2 (continuação)

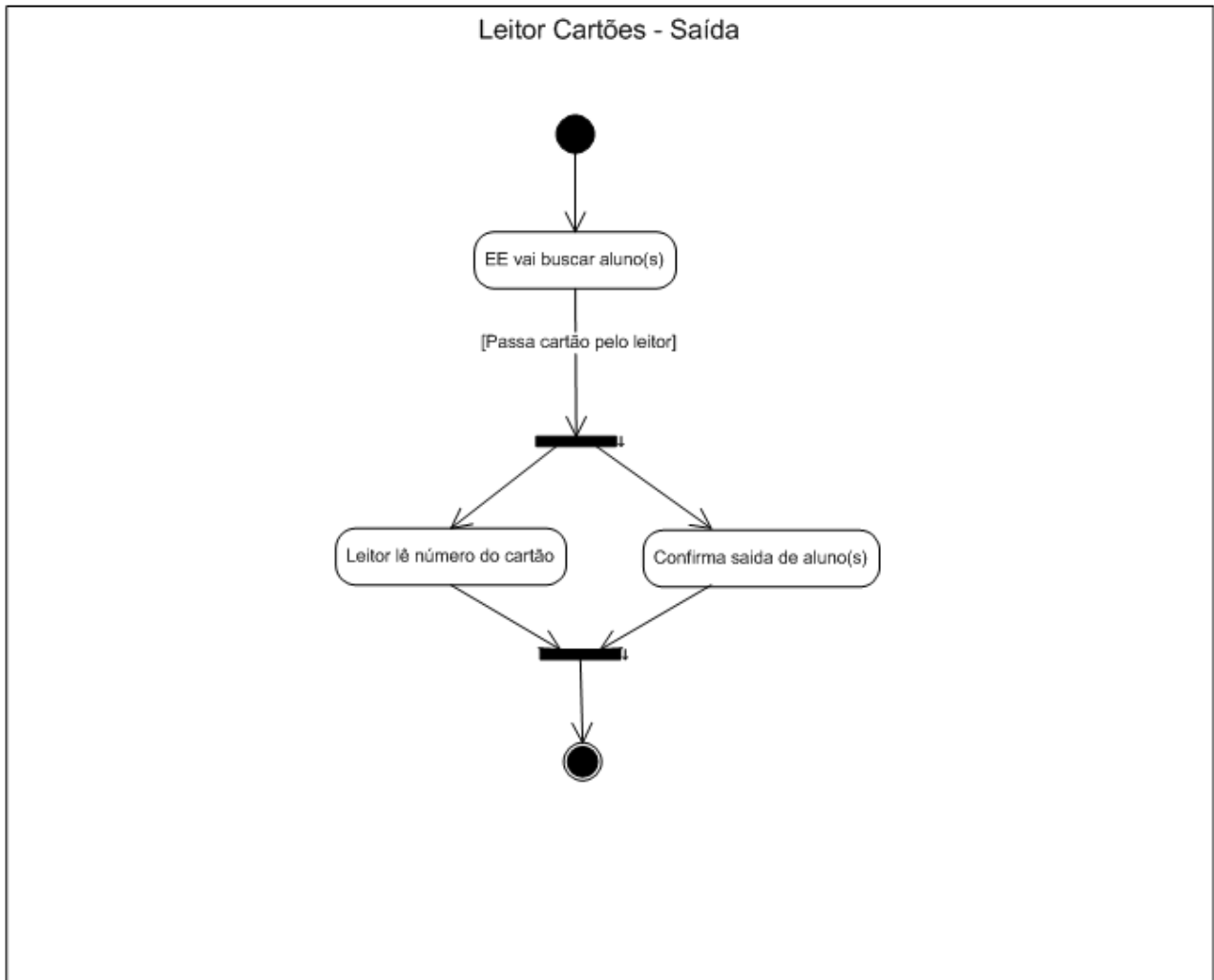


Figura 6: Diagrama de Actividade do Use Case nº2 (continuação)

Quando for a saída da escola o EE vai buscar o aluno e tem de passar novamente o cartão pelo leitor para ficar registado que o aluno já não se encontra na escola. A não passagem do cartão vai implicar que seja registada uma saída pelo sistema, na hora definida pela escola para o seu encerramento.

3.3 Diagrama de Estados

3.3.1 Diagrama de Estados de Presença

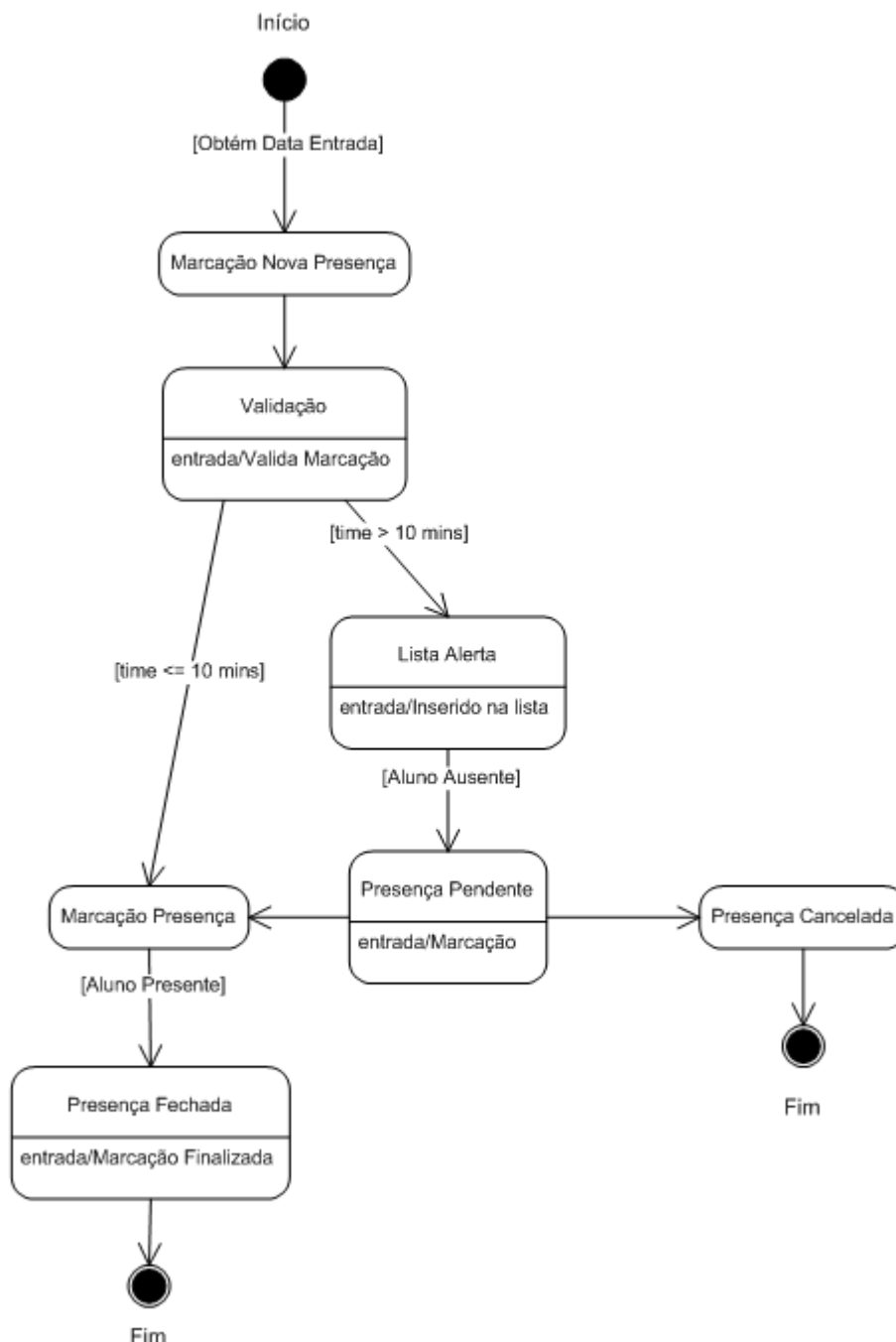


Figura 7: Diagrama de Estados de Presença

O diagrama de estados de presença começa por obter uma data de entrada que é referente à passagem do cartão pelo leitor, essa informação é recepcionada e colocada no estado Marcação Nova Presença, após a conclusão desse estado passamos para o estado de Validação, aqui é necessário efectuar uma validação da

presença do aluno. Caso o tempo até chegar à sala de aula seja inferior ou igual a 10 minutos, então o aluno está presente e por isso encontra-se finalizada o processo de marcação de presença. No caso de o tempo exceder os 10 minutos, passa para o estado Lista Alerta onde um conjunto de informações é inserido numa lista, passando o aluno a ficar com a sua presença ausente, uma vez pendente esta pode ser cancelada e por isso chega ao fim ou pede ser confirmada ficando o aluno com a marcação finalizada.

3.3.2 Diagrama de Estados de Marcação Nova Presença

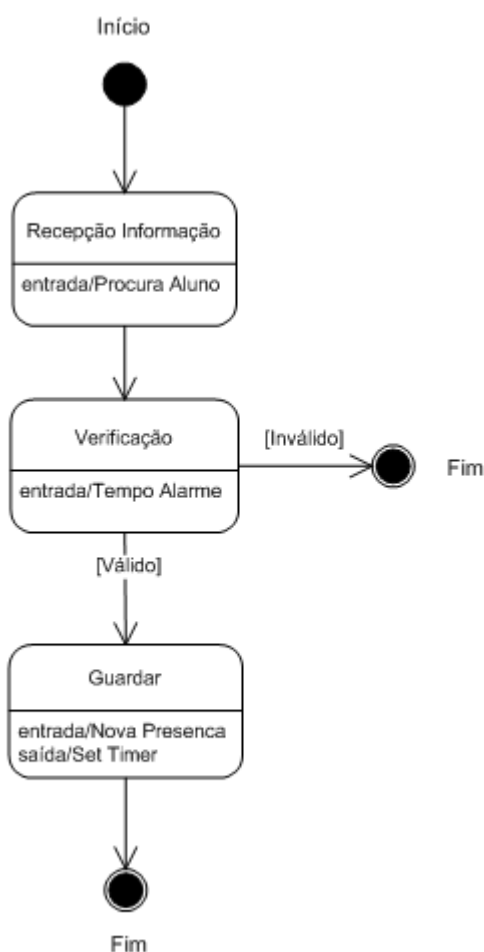


Figura 8: Diagrama de Estados de Marcação Nova Presença

Este diagrama é iniciado com o estado de recepção da informação e é iniciada a procura do aluno, depois a verificação onde é iniciado o tempo de alarme, se for inválido acaba se for válido passa para o estado guardar onde é iniciada uma nova presença e conclui com o Set Timer que vai "buscar" a hora de entrada.

3.4 Diagramas de Sequência

3.4.1 Use Case nº1

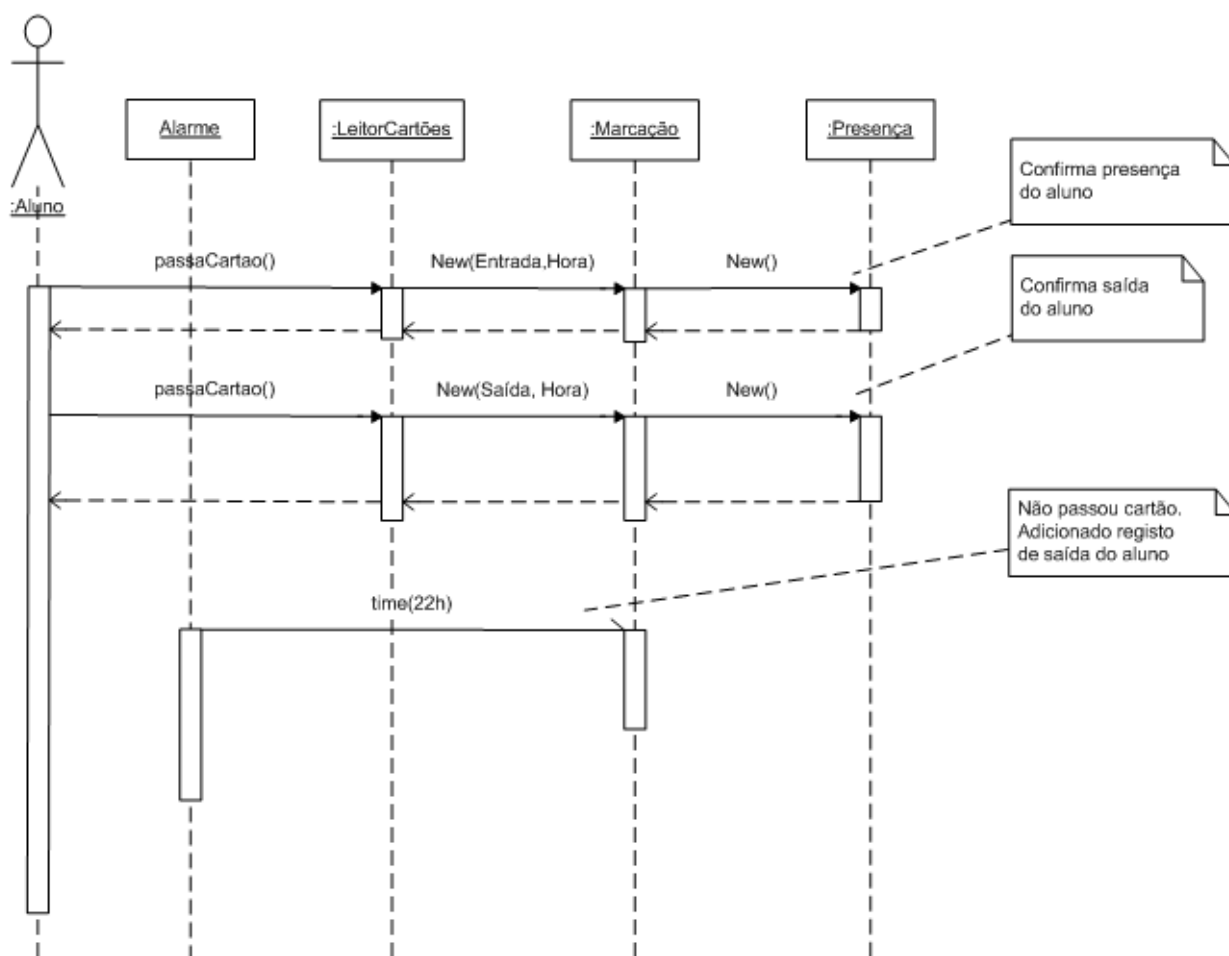


Figura 9: Diagrama de Sequência do Use Case nº1

Na identificação dos objectos podemos identificar Aluno, Alarme, Leitor Cartões, Marcação e Presença.

Aluno é o actor responsável por despoletar o use case. Leitor Cartões representa a interface com o actor, onde este pode marcar a presença. Marcação representa o estado e hora a que o actor passa o cartão. Presença representa a gravação e confirmação do estado.

O use case começa quando o aluno passa o cartão na entrada da escola. O leitor lê o número e verifica que é entrada e a hora respectiva, é marcada presença do aluno e gravada a mesma com o seu estado, neste caso vai ser de entrada. O mesmo acontece para a saída, no entanto caso o actor não tenha passado o cartão à saída, às 22 horas é despoletado pelo timer um alarme que vai efectuar a marcação da saída do aluno.

3.4.2 Use Case nº2

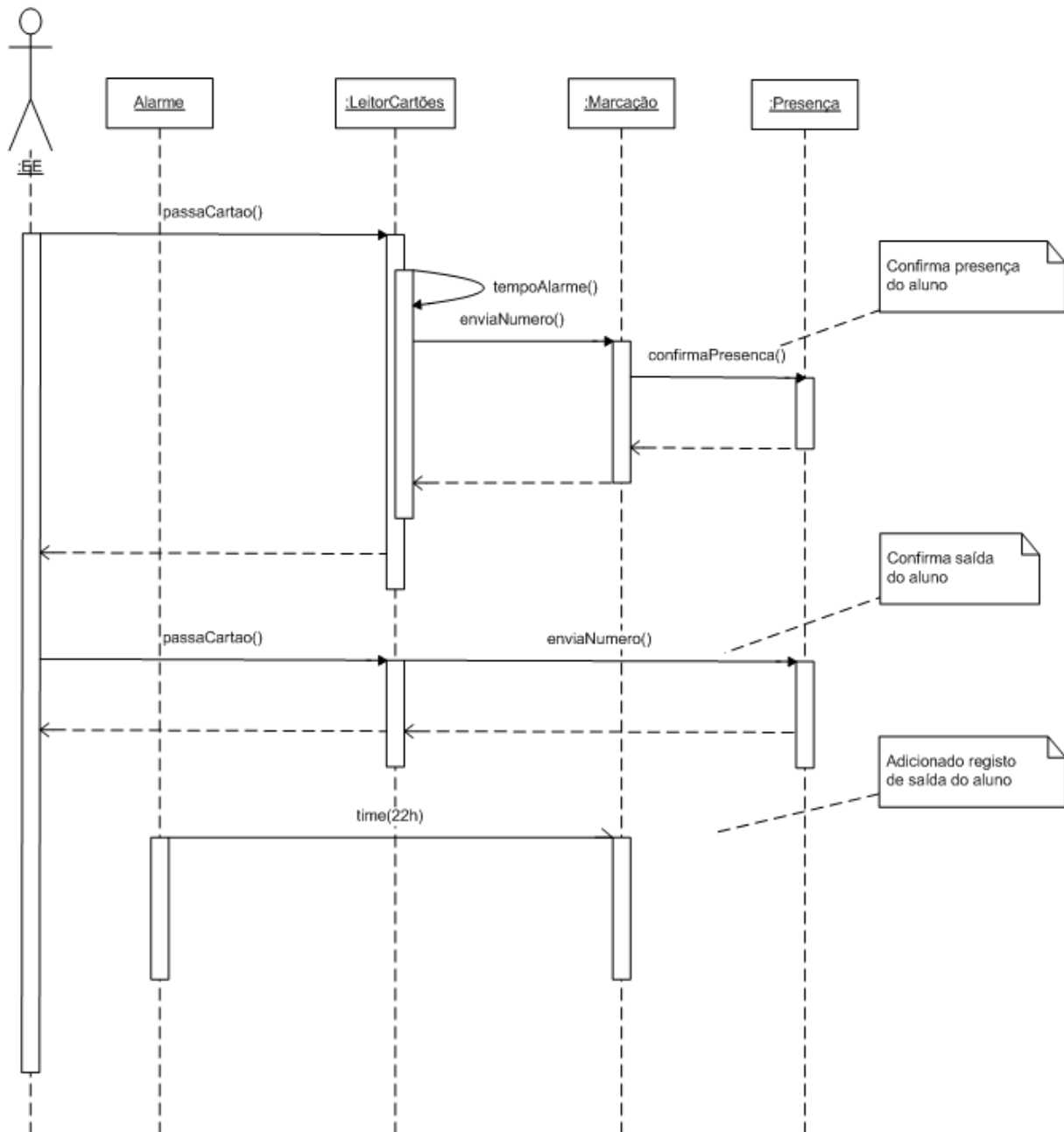


Figura 10: Diagrama de Sequência do Use Case nº2

Na identificação dos objectos podemos identificar EE, Alarma, Leitor Cartões, Marcação e Presença.

EE é o actor responsável por despoletar o use case. Leitor Cartões representa a interface com o actor, onde este pode marcar a presença. Marcação representa o estado e hora a que o actor passa o cartão. Presença representa a gravação e confirmação do estado.

O use case começa quando o EE passa o cartão na entrada da escola. O leitor lê o

número e é iniciado o tempo de alarme, verifica que é entrada e a hora respectiva, é marcada presença do aluno e gravada a mesma com o seu estado, neste caso vai ser de entrada. No entanto a confirmação de presença vai ser efectuada pelo Docente. O mesmo acontece para a saída, no entanto caso o actor não tenha passado o cartão à saída, às 22 horas é despoletado pelo timer um alarme que vai efectuar a marcação da saída do aluno.

3.4.3 Use Case nº2 Alternativo

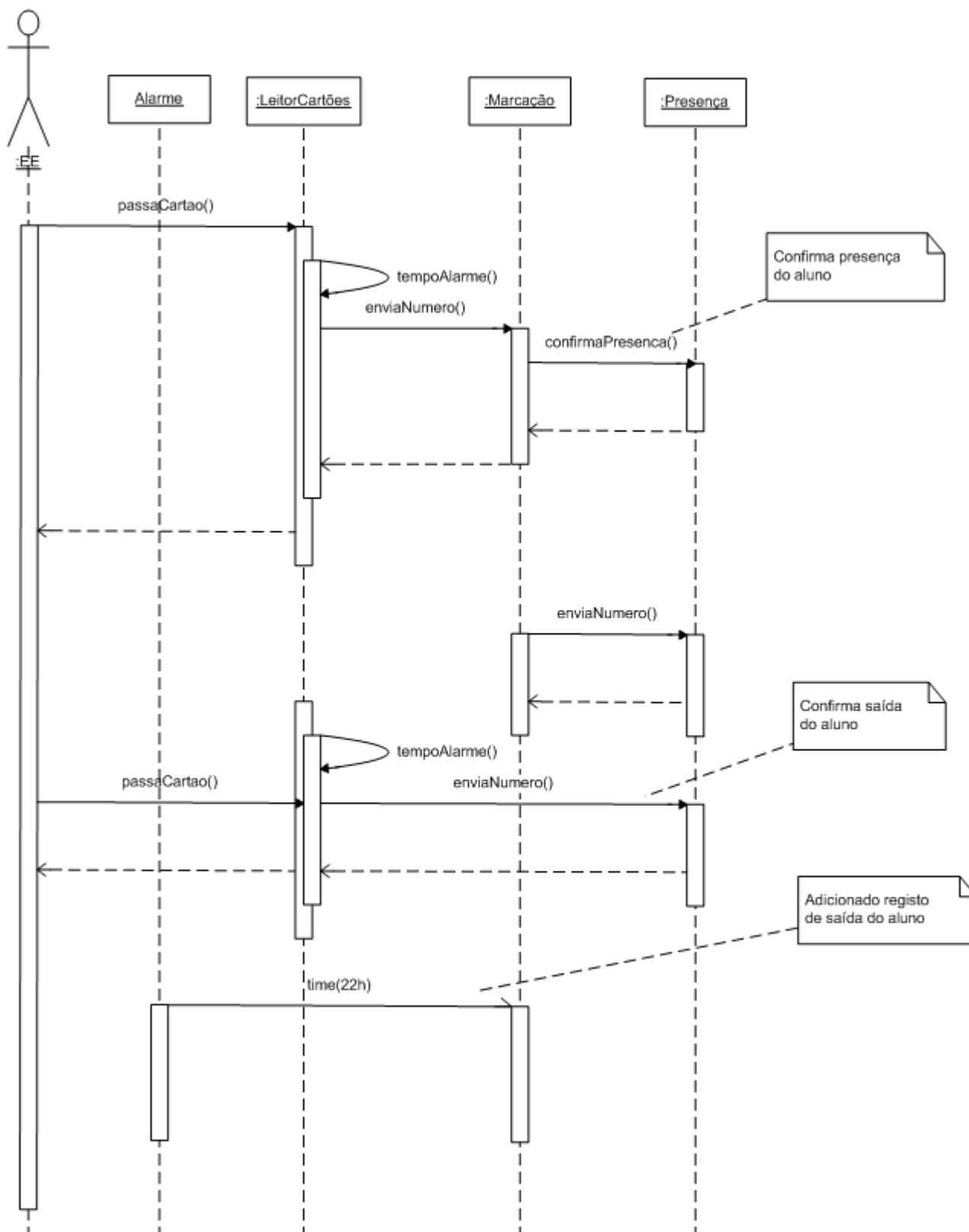


Figura 11: Diagrama de Sequência do Use Case nº2 Alternativo

Na identificação dos objectos podemos identificar EE, Alarme, Leitor Cartões, Marcação e Presença.

EE é o actor responsável por despoletar o use case. Leitor Cartões representa a interface com o actor, onde este pode marcar a presença. Marcação representa o estado e hora a que o actor passa o cartão. Presença representa a gravação e confirmação do estado.

O use case começa quando o EE passa o cartão na entrada da escola. O leitor lê o número e é iniciado o tempo de alarme, verifica que é entrada e a hora respectiva, é marcada presença do aluno e gravada a mesma com o seu estado, neste caso vai ser de entrada. No entanto a confirmação de presença vai ser efectuada pelo Docente.

Na saída, ainda na sala de aula o docente selecciona qual é o EE que leva o aluno, é iniciado o tempo de alarme e assim quando o EE passar o cartão ao sair é logo confirmada a sua saída. Caso o actor não tenha passado o cartão à saída, às 22 horas é despoletado pelo timer um alarme que vai efectuar a marcação da saída do aluno.

3.5 Diagrama de Classes

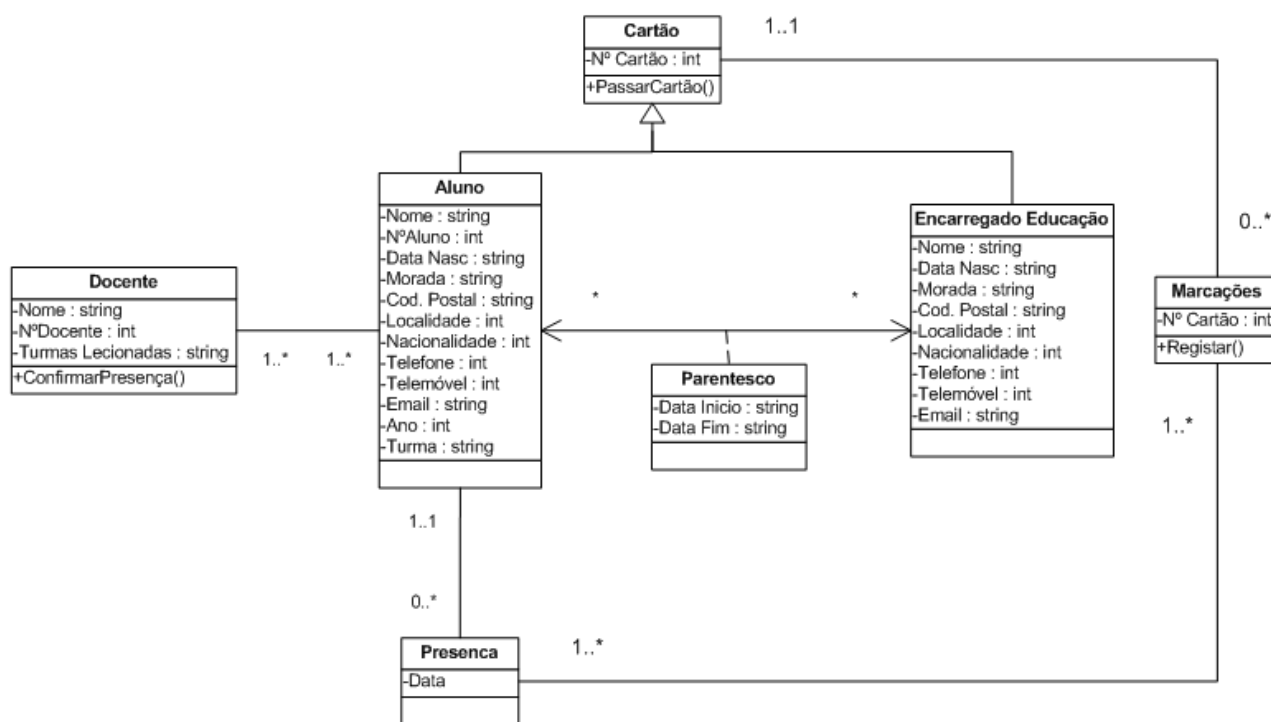


Figura 12: Diagrama de Classes

No diagrama de classes estão presentes as classes Docente, Aluno, Cartão, Parentesco, Encarregado Educação, Marcações e Presença.

Como é possível visualizar, Cartão é uma Super-Classe, Aluno e Encarregado de Educação são subclasses, estando assim presente no diagrama o conceito de herança. Esta situação acontece pois o Aluno e o Encarregado de Educação têm de ter um cartão associado a si próprios. Cartão tem como operação PassarCartão() que também vai ser utilizada pelas duas subclasses.

A classe Parentesco é uma classe Associativa, pois é necessário saber em detalhe o número de vezes que o Aluno e Encarregado de Educação vão entrar na escola. A sua multiplicidade é de muitos para muitos pois cada aluno pode ter vários encarregados de educação e vice-versa.

A classe Docente tem como operação ConfirmarPresença(), pois é uma acção necessária para controlar as presenças e ausências dos alunos e com quem vieram. A sua multiplicidade é de um para muitos. Um docente tem de ter um ou vários alunos e

um aluno tem de ter um ou vários professores.

A classe Presença é composta apenas pelo atributo Data, essencial para manter um controle sobre a presença do aluno. A sua multiplicidade com a classe Aluno é de um para um, pois para existir uma presença é obrigatório existir um aluno. E a multiplicidade de Aluno com Presença é de zero para muitos, pois por exemplo um aluno pode não ir á escola, logo não tem presença nesse dia.


A classe Marcações é constituída nos atributos pelo Nº Cartão, pois para existir uma marcação tem de existir um número de cartão. Tem como operação Registrar(), que tal como o nome indica, regista a presença do aluno. A sua multiplicidade com a classe Presença é de um para muitos e vice-versa, a presença pode ser uma ou várias marcações, dependendo por exemplo do número de aulas que o aluno possui e o contrário também é aplicado, ou seja, nas presenças pode existir uma ou várias marcações. Relativamente à multiplicidade com a classe Cartão, esta é de um para um porque para existir uma marcação tem de existir um cartão, sem este a marcação não será feita. E de zero para muitos pois um cartão pode não ter nenhuma marcação ou pode ter várias. A título de exemplo, se o aluno não for à escola, o cartão não vai ter nenhuma marcação efectuada nesse dia.

4. Testes Efectuados

Durante o desenvolvimento do projecto surgiu a necessidade de realizar vários testes no sentido de melhorar mapear a evolução do projecto e o comportamento da aplicação. Assim criamos a seguinte tabela que foi sendo preenchida de acordo com o sucesso ou insucesso da aplicação em cumprir determinada funcionalidade.

Legenda:

Sucesso - 

Insucesso - 

Incompleto - 












	Aluno	EE	Professor	Portaria	Administrador
Entradas			-	-	-
Saídas			-	-	-
Controle de Entradas	-	-			-
Controle de Saídas	-	-			-
Controle de Presenças	-	-		-	-
Gestão de utilizadores	-	-	-	-	
Relatórios	-	-	-	-	

Figura 13: Grelha de Testes

5. Conclusão

No decorrer deste projecto foram aplicadas as matérias leccionadas ao longo da Licenciatura em Engenharia de Informática, assim, o grupo tinha como objectivo implementar todos os conhecimentos que aprendeu, deste a modelação até à programação, passando pelo cuidado a estética da página.

Os resultados obtidos foram bastante satisfatórios, pois apesar de no início existir alguma dificuldade em como aplicar os conceitos em determinados processos, conseguimos ultrapassá-los por nós próprios mas também com a ajuda dos professores.

Este trabalho revelou-se muito satisfatório, pois apesar de ser um tema proposto, esta é uma área que o grupo não tinha qualquer conhecimento e foram necessárias várias reuniões e visita às instalações da CEBI para perceber os objectivos pretendidos e aquilo que poderia ser o produto final. O nosso intuito é que a aplicação possa vir a ser utilizada num futuro próximo, mesmo que não seja na sua totalidade possa servir como exemplo para futuras referências.

Para uma melhor percepção podemos concluir que os objectivos podem ser listados da seguinte forma:

- Identificação e descrição dos Use Cases (processos principais identificados no âmbito do case study e stakeholders);
- O docente deve:
 - Controlar os acessos dos alunos e encarregado de educação;
 - Marca presenças e ausências dos alunos;
 - Visualiza as entradas e saídas dos alunos;
 - Acede à informação das várias turmas que lecciona;
- A portaria deve:
 - Controlar os acessos dos alunos e encarregado de educação;
 - Visualiza as entradas e saídas dos alunos;
- O administrador de sistema deve:
 - Gerir a informação dos alunos;
 - Gerir a informação dos docentes;
 - Gerir a informação dos encarregados de educação;

6. Bibliografia

6.1 Urls

- <http://www.fcebi.org>
- <http://forums.asp.net/>
- <http://msdn.microsoft.com>
- <http://www.w3schools.com>
- <http://pplware.sapo.pt>
- <http://ajax.net-tutorials.com/controls/updatepanel-control/>
- www.macoratti.net/07/07/ajax_tim.htm
- www.codigofonte.net/dicas/aspnet/403_adicionando-campos-dinamicamente-no-gridview

6.2 Referências

- Fundamental de UML, 2ª Edição, Mauro Nunes e Henrique O'Neill
- Slides das Aulas de Análise e Concepção de Sistemas
- Slides das Aulas de Sistemas de Informação Multimédia
- Slides das Aulas de Engenharia de Software

7. Anexos

7.1 Código Fonte da aplicação desenvolvida

Portaria

```
protected void btn_inserir_Click(object sender, EventArgs e)
{
    con.Close();
    con.Open();
    SqlCommand cmd = new SqlCommand("SELECT * FROM [Alunos-EE] WHERE idCartao='" +
    TextBox1.Text + "'", con);

    dr = cmd.ExecuteReader();

    while (dr.Read())
    {
        if (dr["tipo"].ToString() == "A")
        {
            tipo = "A";
            entrou_aluno = dr["entrou"].ToString();
            procura_aluno_geral(Convert.ToInt32(TextBox1.Text));
            break;
        }
        else if (dr["tipo"].ToString() == "EE")
        {
            tipo = "EE";

            procura_EE_geral(Convert.ToInt32(TextBox1.Text));
        }
    }

    con.Close();

    dt = new DataTable();
    MakeDataTable();
    AddToDataTable();
    BindGrid();

    dt = (DataTable)ViewState["DataTable"];
}

#region "procura_aluno"
void procura_aluno_geral(int id)
{
    con2.Close();
    con2.Open();
    SqlCommand cmd2 = new SqlCommand("SELECT * FROM [Alunos-EE] WHERE idCartao="+
id, con2);

    dr2 = cmd2.ExecuteReader();

    while (dr2.Read())
    {
        id_aluno = dr2["idCartao"].ToString();
        nome_aluno = dr2["nome"].ToString();
        turma_aluno = dr2["turma"].ToString();
        ano_aluno = dr2["ano"].ToString();
        entrou_aluno = dr2["entrou"].ToString();
    }

    con2.Close();
}
```

```
if (entrou_aluno == "0")
{
    string query3 = "insert into Regis-
to(idCartao,nome,tipo,turma,ano,dataEntrada,entrou,dataDia, autorSair) Values" +
    "(@paridCartao, @parnome,@partipo, @parturma,
@parano,@parEntrada,@parEntrou,@pardataDia,@parautoSair)";

    SqlCommand comando3 = new SqlCommand(query3, con3);

    SqlParameter par1 = new SqlParameter("@paridCartao", SqlDbType.Int);
    par1.Value = id_aluno;
    comando3.Parameters.Add(par1);

    SqlParameter par2 = new SqlParameter("@parnome", SqlDbType.NVarChar);
    par2.Value = nome_aluno;
    comando3.Parameters.Add(par2);

    SqlParameter par9 = new SqlParameter("@partipo", SqlDbType.NVarChar);
    par9.Value = "A";
    comando3.Parameters.Add(par9);

    SqlParameter par3 = new SqlParameter("@parturma", SqlDbType.NVarChar);
    par3.Value = turma_aluno;
    comando3.Parameters.Add(par3);

    SqlParameter par4 = new SqlParameter("@parano", SqlDbType.Int);
    par4.Value = ano_aluno;
    comando3.Parameters.Add(par4);

    SqlParameter par5 = new SqlParameter("@parEntrada", SqlDbType.DateTime);
    par5.Value = DateTime.Now;
    comando3.Parameters.Add(par5);

    SqlParameter par7 = new SqlParameter("@parEntrou", SqlDbType.Int);

    par7.Value = entrou_aluno = "1";

    comando3.Parameters.Add(par7);

    SqlParameter par8 = new SqlParameter("@pardataDia", SqlDbType.NVarChar);
    par8.Value = Convert.ToString(DateTime.Today.Day) + "-" + Con-
vert.ToString(DateTime.Today.Month) + "-" + Convert.ToString(DateTime.Today.Year);
    comando3.Parameters.Add(par8);

    SqlParameter par10 = new SqlParameter("@parautoSair", SqlDbType.Int);
    par10.Value = 1;
    comando3.Parameters.Add(par10);

    con3.Close();
    con3.Open();

    int num = comando3.ExecuteNonQuery();

    con3.Close();

    Db.ExecuteNonQuery("update [Alunos-EE] set entrou = 1 where idCartao='" +
id_aluno + "'");

    lblNomeShow.Text = nome_aluno;
    lblAnoShow.Text = ano_aluno;
    lblTurmaShow.Text = turma_aluno;
```

```
        inserir_imagens(Convert.ToInt32(id_aluno));
    }
    else
    {
        validar_autorizacao_Professora();
        if (autorizacao_Sair == "0")
        {
            string query4 = "insert into Regis-
to(idCartao,nome,tipo,turma,ano,dataEntrada,dataSaida,entrou,dataDia) Values" +
            "(@paridCartao, @parnome,@partipo, @parturma, @parano,@parEntrada,
@parSaida,@parEntrou,@pardataDia)";

            SqlCommand comando4 = new SqlCommand(query4, con4);

            SqlParameter par1 = new SqlParameter("@paridCartao", SqlDbType.Int);
            par1.Value = id_aluno;
            comando4.Parameters.Add(par1);

            SqlParameter par2 = new SqlParameter("@parnome", SqlDbType.NVarChar);
            par2.Value = nome_aluno;
            comando4.Parameters.Add(par2);

            SqlParameter par9 = new SqlParameter("@partipo", SqlDbType.NVarChar);
            par9.Value = "A";
            comando4.Parameters.Add(par9);

            SqlParameter par3 = new SqlParameter("@parturma", SqlDbType.NVarChar);
            par3.Value = turma_aluno;
            comando4.Parameters.Add(par3);

            SqlParameter par4 = new SqlParameter("@parano", SqlDbType.Int);
            par4.Value = ano_aluno;
            comando4.Parameters.Add(par4);

            proc_dataEntrada(Convert.ToInt32(id_aluno));

            SqlParameter par5 = new SqlParameter("@parEntrada", SqlDbType.DateTime);
            par5.Value = Convert.ToDateTime(data_entrada);
            comando4.Parameters.Add(par5);

            SqlParameter par6 = new SqlParameter("@parSaida", SqlDbType.DateTime);
            par6.Value = DateTime.Now;
            comando4.Parameters.Add(par6);

            SqlParameter par7 = new SqlParameter("@parEntrou", SqlDbType.Int);

            par7.Value = entrou_aluno = "0";

            comando4.Parameters.Add(par7);

            SqlParameter par8 = new SqlParameter("@pardataDia", SqlDbType.NVarChar);
            par8.Value = Convert.ToString(DateTime.Today.Day) + "-" + Con-
vert.ToString(DateTime.Today.Month) + "-" + Convert.ToString(DateTime.Today.Year);
            comando4.Parameters.Add(par8);

            con4.Close();
            con4.Open();

            int num = comando4.ExecuteNonQuery();

            con4.Close();
```

```
Db.ExecuteNonQuery("update [Alunos-EE] set entrou = 0 where idCartao='" + id_aluno + "'");

    lblNomeShow.Text = nome_aluno;
    lblAnoShow.Text = ano_aluno;
    lblTurmaShow.Text = turma_aluno;

    inserir_imagens(Convert.ToInt32(id_aluno));
}

}

}

#endregion

#region "procura_EE"

void procura_EE_geral(int id_EE_)
{
    con5.Close();
    con5.Open();
    SqlCommand cmd5 = new SqlCommand("SELECT * FROM [Alunos-EE] WHERE idCartao=" + id_EE_, con5);

    dr5 = cmd5.ExecuteReader();

    while (dr5.Read())
    {
        id_EE = dr5["idCartao"].ToString();
        nome_EE = dr5["nome"].ToString();
        entrou_EE = dr5["entrou"].ToString();
    }

    con5.Close();

    if (entrou_EE == "0")
    {
        string query6 = "insert into Regis-
to(idCartao,nome,tipo,dataEntrada,entrou,dataDia) Values" +
        "(@paridCartao, @parnome,@parTipo, @parEntrada, @parEntrou,
@pardataDia)";

        SqlCommand comando6 = new SqlCommand(query6, con6);

        SqlParameter par1 = new SqlParameter("@paridCartao", SqlDbType.Int);
        par1.Value = id_EE;
        comando6.Parameters.Add(par1);

        SqlParameter par2 = new SqlParameter("@parnome", SqlDbType.NVarChar);
        par2.Value = nome_EE;
        comando6.Parameters.Add(par2);

        SqlParameter par6 = new SqlParameter("@partipo", SqlDbType.NVarChar);
        par6.Value = tipo;
        comando6.Parameters.Add(par6);

        SqlParameter par3 = new SqlParameter("@parEntrada", SqlDbType.DateTime);
        par3.Value = DateTime.Now;
        comando6.Parameters.Add(par3);

        SqlParameter par5 = new SqlParameter("@parEntrou", SqlDbType.Int);
```



```
        if (entrou_EE == "0")
        {
            par5.Value = entrou_EE = "1";
        }

        comando6.Parameters.Add(par5);

        SqlParameter par8 = new SqlParameter("@pardataDia", SqlDbType.NVarChar);
        par8.Value = Convert.ToString(DateTime.Today.Day) + "-" + Con-
vert.ToString(DateTime.Today.Month) + "-" + Convert.ToString(DateTime.Today.Year);
        comando6.Parameters.Add(par8);

        con6.Close();
        con6.Open();

        int num = comando6.ExecuteNonQuery();

        con6.Close();

        Db.ExecuteNonQuery("update [Alunos-EE] set entrou = 1 where idCartao='" +
id_EE + "'");

        lblNomeShow.Text = nome_EE;

        inserir_imagens(Convert.ToInt32(id_EE));

        alarmeEE_Inserir();

        procura_R_EE_Aluno(id_EE_);
    }
    else
    {
        string query6 = "insert into Regis-
to(idCartao,nome,tipo,dataEntrada,dataSaida,entrou,dataDia) Values" +
        "(@paridCartao, @parnome,@partipo,@parEntrada,
@parSaida,@parEntrou,@pardataDia)";

        SqlCommand comando6 = new SqlCommand(query6, con6);

        SqlParameter par1 = new SqlParameter("@paridCartao", SqlDbType.Int);
        par1.Value = id_EE;
        comando6.Parameters.Add(par1);

        SqlParameter par2 = new SqlParameter("@parnome", SqlDbType.NVarChar);
        par2.Value = nome_EE;
        comando6.Parameters.Add(par2);

        SqlParameter par9 = new SqlParameter("@partipo", SqlDbType.NVarChar);
        par9.Value = "EE";
        comando6.Parameters.Add(par9);

        proc_dataEntrada(Convert.ToInt32(id_EE));

        SqlParameter par5 = new SqlParameter("@parEntrada", SqlDbType.DateTime);
        par5.Value = Convert.ToDateTime(data_entrada);
        comando6.Parameters.Add(par5);

        SqlParameter par6 = new SqlParameter("@parSaida", SqlDbType.DateTime);
        par6.Value = DateTime.Now;
        comando6.Parameters.Add(par6);

        SqlParameter par7 = new SqlParameter("@parEntrou", SqlDbType.Int);
```

```

        par7.Value = entrou_EE = "0";

        comando6.Parameters.Add(par7);

        SqlParameter par8 = new SqlParameter("@pardataDia", SqlDbType.NVarChar);
        par8.Value = Convert.ToString(DateTime.Today.Day) + "-" + Con-
vert.ToString(DateTime.Today.Month) + "-" + Convert.ToString(DateTime.Today.Year);
        comando6.Parameters.Add(par8);

        con6.Close();
        con6.Open();

        int num = comando6.ExecuteNonQuery();

        con6.Close();

        Db.ExecuteNonQuery("update [Alunos-EE] set entrou = 0 where idCartao='" +
id_EE + "'");

        lblNomeShow.Text = nome_EE;

        inserir_imagens(Convert.ToInt32(id_EE));

        procuraR_EE_A();
    }

}

#endregion

#region "procura_EE_Aluno"
void procura_R_EE_Aluno(int id_EE_A)
{
    con7.Close();
    con7.Open();
    SqlCommand cmd7 = new SqlCommand("SELECT * FROM R_EE_Aluno WHERE idEE=" +
id_EE_A, con7);

    dr7 = cmd7.ExecuteReader();

    while (dr7.Read())
    {
        id_aluno = dr7["idA"].ToString();
        procura_aluno_geral(Convert.ToInt32(id_aluno));
    }

    con7.Close();
}
#endregion

void procuraR_EE_A()
{
    con8.Close();
    con8.Open();
    SqlCommand cmd8 = new SqlCommand("SELECT idEE, idASS FROM dbo.R_EE_ASS", con8);

    dr8 = cmd8.ExecuteReader();

    while (dr8.Read())
    {
        if (TextBox1.Text == dr8["idEE"].ToString())
        {
            procura_R_EE_Aluno(Convert.ToInt32(TextBox1.Text));

```

```

    }
}
con8.Close();

con9.Close();
con9.Open();
SqlCommand cmd9 = new SqlCommand("SELECT idEE, idASS FROM dbo.R_EE_ASS", con9);

dr9 = cmd9.ExecuteReader();

while (dr9.Read())
{
    if(textBox1.Text == dr9["idASS"].ToString())
    {
        procura_R_EE_Aluno(Convert.ToInt32(dr9["idEE"].ToString()));
    }
}
con9.Close();
}
#region "CarregarGRID"

private void MakeDataTable()
{
    dt.Columns.Add("Nº Aluno");
    dt.Columns.Add("Nome");
    dt.Columns.Add("Tipo");
    dt.Columns.Add("Ano");
    dt.Columns.Add("Turma");
    dt.Columns.Add("Data Entrada");
    dt.Columns.Add("Data Saída");
    dt.Columns.Add("ID");
}

private void AddToDataTable()
{
    con14.Close();
    con14.Open();
    int countBD = 0;

    SqlCommand cmd14 = new SqlCommand("SELECT Registo.idCartao AS Nº, Registo.nome
AS Nome, Registo.tipo AS Tipo, Registo.turma AS Turma, Registo.ano AS Ano, Regis-
to.dataEntrada AS [Data Entrada], Registo.dataSaida AS [Data Saída], Registo.idRegisto AS ID
FROM Registo INNER JOIN [Alunos-EE] ON Registo.idCartao = [Alunos-EE].idCartao ORDER BY ID
DESC", con14);

    dr14 = cmd14.ExecuteReader();

    while (dr14.Read())
    {
        if (countBD != 10)
        {
            nG = dr14["Nº"].ToString();
            nomeG = dr14["Nome"].ToString();
            tipoG = dr14["Tipo"].ToString();
            turmaG = dr14["Turma"].ToString();
            anoG = dr14["Ano"].ToString();
            dataInicioG = dr14["Data Entrada"].ToString();
            dataSaidaG = dr14["Data Saída"].ToString();
            idG = dr14["ID"].ToString();

            dt.Rows.Add(nG, nomeG, tipoG, anoG, turmaG, dataInicioG, dataSaidaG,
idG);

            countBD++;
        }
    }
}

```

```

    }
}

}

private void BindGrid()
{
    GridView1.DataSource = dt;
    GridView1.DataBind();
}

#endregion
protected void Timer2_Tick(object sender, EventArgs e)
{
    UpdateLabel();
    alarmeEE_Ler();
    if (System.DateTime.Now.Hour == 22 && System.DateTime.Now.Minute == 00)
    {
        procuraAlunosEESair();
    }
}
#region "AtribuirAlarme"

void alarmeEE_Inserir()
{
    string vazio = "";
    if (entrou_EE == "1")
    {
        DateTime data10 = DateTime.Now.AddMinutes(10);

        Db.ExecuteNonQuery("update [Registo] set alarme = '" + Convert.ToString(data10) + "' where idCartao='" + id_EE + "'");
    }
    else
    {
        Db.ExecuteNonQuery("update [Registo] set alarme = '" + vazio + "' where id-Cartao='" + id_EE + "'");
    }
}

void alarmeEE_Ler()
{
    string dataDia = Convert.ToString(DateTime.Today.Day) + "-" + Convert.ToString(DateTime.Today.Month) + "-" + Convert.ToString(DateTime.Today.Year);

    con15.Close();
    con15.Open();

    SqlCommand cmd15 = new SqlCommand(" SELECT dbo.Registo.*, dbo.[Alunos-EE].* " +
        " FROM    dbo.[Alunos-EE] INNER JOIN " +
        "         dbo.Registo ON dbo.[Alunos-EE].idCartao = dbo.Registo.idCartao " +
        "         WHERE (dbo.[Alunos-EE].tipo = N'EE') AND " +
        "         dbo.[Alunos-EE].idCartao = '" + id_EE + "' AND dbo.[Alunos-EE].entrou = 1 AND " +
        "         dbo.Registo.dataDia = '" + dataDia + "' ", con15);

    dr15 = cmd15.ExecuteReader();

    while (dr15.Read())
    {
        alarmeEE = dr15["alarme"].ToString();
    }
}

```

```

        if (alarmeEE == Convert.ToString(DateTime.Now))
        {
            alarmeEE_Erro();
        }
    }
    con15.Close();
}

```

Professor

#region ""carregadrop e Ler"

```

private void carregardropdownlist1()
{
    con.Close();
    con.Open();

    int x = 0;
    string ano = "";

    int i = Convert.ToInt32(Session["idProfessor"].ToString());

    SqlCommand cmd = new SqlCommand("SELECT Ano.ano FROM R_P_AT INNER JOIN " +
        " Professores ON R_P_AT.idP = Professores.idProfessor INNER JOIN " +
        " Turma ON R_P_AT.idTA = Turma.idTA INNER JOIN " +
        " Ano ON Turma.idA = Ano.idAT " +
        " WHERE Professores.idProfessor = " + i, con);

    dr = cmd.ExecuteReader();

    while (dr.Read())
    {
        if (ano != dr["ano"].ToString())
        {
            DropDownList1.Items.Add(dr["ano"].ToString());
            ano = dr["ano"].ToString();

            if (x == 0)
            {
                ddl1 = dr["ano"].ToString();
                x++;
            }
        }
        else { }
    }

    con.Close();
}

private void carregardropdownlist2()
{
    con.Close();
    con.Open();

    string turma = "";
    string y = "";

    SqlCommand cmd = new SqlCommand("SELECT dbo.Turma.turma FROM dbo.R_P_AT INNER
JOIN " +

```

```

        "dbo.Professores ON dbo.R_P_AT.idP = dbo.Professores.idProfessor INNER
JOIN " +
        "dbo.Turma ON dbo.R_P_AT.idTA = dbo.Turma.idTA INNER JOIN " +
        "dbo.Ano ON dbo.Turma.idA = dbo.Ano.idAT " +
        " WHERE dbo.Professores.idProfessor = '" + Ses-
sion["idProfessor"].ToString() + "' AND (dbo.Ano.ano = '" + ddl1 + "')", con);

        dr = cmd.ExecuteReader();

        while (dr.Read())
        {
            if (turma != dr["turma"].ToString())
            {
                DropDownList2.Items.Add(dr["turma"].ToString());
                turma = dr["turma"].ToString();

                if (y == "")
                {
                    y = ddl2 = dr["turma"].ToString();
                }
            }
            else { }
        }

        con.Close();
    }
protected void DropDownList1_SelectedIndexChanged(object sender, EventArgs e)
{
    ListBox1.Items.Clear();
    ListBox2.Items.Clear();

    ddl1 = DropDownList1.SelectedValue;
    DropDownList2.Items.Clear();
    carregardropdownlist2();

    carregalistboxinfo();
    carregarListAlunos();

}

protected void DropDownList2_SelectedIndexChanged(object sender, EventArgs e)
{
    ListBox1.Items.Clear();
    ListBox2.Items.Clear();

    ddl1 = DropDownList1.SelectedValue;
    ddl2 = DropDownList2.SelectedValue;

    carregalistboxinfo();
    carregarListAlunos();

}
#endregion

#region "CarregarListAlunos"
void carregarListAlunos()

```

```

{
    CheckBoxList1.Items.Clear();
    con.Close();
    con.Open();

    SqlCommand cmd = new SqlCommand("SELECT * FROM dbo.[Alunos-EE] WHERE (turma = '"
+ dd12 + "') AND (ano = '" + dd11 + "')", con);

    dr = cmd.ExecuteReader();
    while (dr.Read())
    {
        ListBox2.Items.Add(new ListItem(dr["Nome"].ToString()));
        CheckBoxList1.Items.Add(dr["Nome"].ToString());
    }
    con.Close();
}

#endregion

#region "CarregaListEE"

void carregaListEE()
{
    con.Open();
    SqlCommand cmd2 = new SqlCommand("SELECT * FROM [Alunos-EE] INNER JOIN
R_EE_Aluno ON [Alunos-EE].idCartao = dbo.R_EE_Aluno.idEE" +
    " WHERE (R_EE_Aluno.idA = '" + id_aluno + "')", con);

    dr = cmd2.ExecuteReader();

    while (dr.Read())
    {
        DropDownList3.Items.Add(dr["nome"].ToString());
        // ListBox3.Items.Add(dr["nome"].ToString());
    }
    con.Close();
}

#endregion

protected void ListBox2_SelectedIndexChanged(object sender, EventArgs e)
{
    id_aluno = "";
    con.Close();
    con.Open();

    SqlCommand cmd = new SqlCommand("SELECT * " +
    " FROM [Alunos-EE] INNER JOIN Fotos ON [Alunos-EE].idCartao =
Fotos.id_alunoe " +
    " WHERE ([Alunos-EE].nome = '" + ListBox2.SelectedItem.ToString() + "')",
con);

    Image1.ImageUrl = "";
    dr = cmd.ExecuteReader();

    while (dr.Read())
    {
        id_imagem = dr["idFoto"].ToString();
    }
}

```

```
        Image1.ImageUrl = "~/Image.ashx?imgid=" + id_imagem;
        id_aluno = dr["idCartao"].ToString();

    }
    con.Close();

}

#region "Ausente e Presente"

void verificarAusente(string idAluno)
{
    string dataDia2 = Convert.ToString(DateTime.Today.Day) + "-" + Convert.ToString(DateTime.Today.Month) + "-" + Convert.ToString(DateTime.Today.Year);

    con.Close();
    con.Open();

    SqlCommand cmd = new SqlCommand("SELECT * FROM [Presencas] WHERE data = '" + dataDia2 + "' and idCartao = '" + idAluno + "'", con);

    dr = cmd.ExecuteReader();

    while (dr.Read())
    {
        valorPresente = dr["presente"].ToString();
        valorAusente = dr["ausente"].ToString();
    }

    if (valorAusente == "1")
    {
        valorAusente = "0";
        Db.ExecuteNonQuery("update [Presencas] set ausente = " + valorAusente + ", presente=1 where idCartao='" + idAluno + "' AND data = '" + dataDia2 + "'");
    }
    if (valorPresente == "1")
    {
    }
    else
    {
        inserirPresente();
    }
}

void procurarAluno(string nome)
{
    con.Close();

    con2.Open();

    SqlCommand cmd = new SqlCommand("SELECT * FROM [Alunos-EE] WHERE nome = '" + nome + "'", con2);

    dr2 = cmd.ExecuteReader();

    while (dr2.Read())
    {
        id_aluno = dr2["idCartao"].ToString();
        verificarAusente(id_aluno);
    }
    con2.Close();

}
```



```

void inserirPresente()
{
    con.Close();
    con.Open();

    string dataDia3 = Convert.ToString(DateTime.Today.Day) + "-" + Con-
vert.ToString(DateTime.Today.Month) + "-" + Convert.ToString(DateTime.Today.Year);

    string query = "insert into Presencas(idCartao,data,presente) Values (" + Con-
vert.ToInt32(id_aluno) + ", '" + dataDia3 + "', "+ 1 +")";

    SqlCommand comando = new SqlCommand(query, con);

    int num = comando.ExecuteNonQuery();
    con.Close();
}

void marcarPresente()
{
    String values = "";
    for (int i = 0; i < CheckBoxList1.Items.Count; i++)
    {
        if (CheckBoxList1.Items[i].Selected)
        {
            values= CheckBoxList1.Items[i].Value;

            procurarAluno(values);
        }
    }
}

private void MakeDataTable()
{
    dt.Columns.Add("Nº Aluno");
    dt.Columns.Add("Nome");
    dt.Columns.Add("Ano");
    dt.Columns.Add("Turma");
    dt.Columns.Add("Data do Dia");
    dt.Columns.Add("Presente");
    dt.Columns.Add("Ausente");
}

private void AddToDataTable()
{
    string dataDia = Convert.ToString(DateTime.Today.Day) + "-" + Con-
vert.ToString(DateTime.Today.Month) + "-" + Convert.ToString(DateTime.Today.Year);
    string nomeA = "";
    con.Close();
    con.Open();

    SqlCommand cmd = new SqlCommand("SELECT dbo.[Alunos-EE].idCartao as ID,
dbo.[Alunos-EE].nome as Nome, dbo.[Alunos-EE].turma as Turma, dbo.[Alunos-EE].ano as Ano,
dbo.Presencas.data as Data, dbo.Presencas.presente as Presente, "+
        " dbo.Presencas.ausente as Ausente FROM dbo.[Alunos-EE] INNER JOIN
dbo.Presencas ON dbo.[Alunos-EE].idCartao = dbo.Presencas.idCartao " +
        " WHERE (dbo.[Alunos-EE].turma = '" + dd12 + "') AND (dbo.[Alunos-
EE].ano = '" + dd11 + "') AND data = '" + dataDia + "' ORDER BY [Alunos-EE].nome", con);

    dr = cmd.ExecuteReader();
}

```

```
while (dr.Read())
{
    if (nome != dr["Nome"].ToString())
    {
        id = dr["ID"].ToString();
        nome = nomeA = dr["Nome"].ToString();
        data = dr["Data"].ToString();

        if (dr["Presente"].ToString() == "0")
        {
            presente = "Não";
        }
        else
        {
            presente = "Sim";
        }

        if (dr["Ausente"].ToString() == "0")
        {
            ausente = "Não";
        }
        else
        {
            ausente = "Sim";
        }

        turma = dr["Turma"].ToString();
        ano = dr["Ano"].ToString();

        dt.Rows.Add(id, nome, ano, turma, data, presente, ausente);
    }
}

con.Close();
}

void inserir_ausente(string id)
{
    con.Close();
    con.Open();

    string dataDia4 = Convert.ToString(DateTime.Today.Day) + "-" + Convert.ToString(DateTime.Today.Month) + "-" + Convert.ToString(DateTime.Today.Year);

    string query = "insert into Presencas(idCartao,data,ausente) Values (" + Convert.ToInt32(id) + ", '" + dataDia4 + "', " + 1 + ")";

    SqlCommand comando = new SqlCommand(query, con);

    int num = comando.ExecuteNonQuery();
    con.Close();
}

void procurar_alunoAusente(string nome)
{
    con.Close();

    con2.Open();

    SqlCommand cmd = new SqlCommand("SELECT * FROM [Alunos-EE] WHERE nome = '" + nome + "'", con2);
```

```

        dr2 = cmd.ExecuteReader();

        while (dr2.Read())
        {
            id_aluno = dr2["idCartao"].ToString();
            verificarPresente(id_aluno);
        }
        con2.Close();
    }

    void verificarPresente(string id_Aluno)
    {
        string dataDia2 = Convert.ToString(DateTime.Today.Day) + "-" + Convert.ToString(DateTime.Today.Month) + "-" + Convert.ToString(DateTime.Today.Year);

        con.Close();
        con.Open();

        SqlCommand cmd = new SqlCommand("SELECT * FROM [Presencas] WHERE data = '" + dataDia2 + "' and idCartao = '" + id_Aluno + "'", con);

        dr = cmd.ExecuteReader();

        while (dr.Read())
        {
            valorPresente = dr["presente"].ToString();
        }

        if (valorPresente == "1")
        { }
        else
        {
            inserir_ausente(id_Aluno);
        }
    }
}

protected void btn_inserir_Click(object sender, EventArgs e)
{
    if (TextBox1.Text == "")
    { }
    else
    {
        con.Close();
        con.Open();
        SqlCommand cmd = new SqlCommand("SELECT * FROM [Alunos-EE] WHERE idCartao='" + TextBox1.Text + "'", con);

        dr = cmd.ExecuteReader();

        while (dr.Read())
        {
            if (dr["tipo"].ToString() == "A")
            {
                tipo = "A";
                entrou_aluno = dr["entrou"].ToString();
                procura_aluno_geral(Convert.ToInt32(TextBox1.Text));
                break;
            }
            else if (dr["tipo"].ToString() == "EE")
            {
                tipo = "EE";
                procura_EE_geral(Convert.ToInt32(TextBox1.Text));
            }
        }
    }
}

```

```

        }
        else
        {
            tipo = "AS";
            procura_associado(Convert.ToInt32(TextBox1.Text));
        }
    }
    con.Close();
    Response.Redirect(Request.RawUrl);
}
}
}
#region "procura_EE"
void procura_EE_geral(int id_EE_)
{
    con.Close();
    con.Open();
    SqlCommand cmd = new SqlCommand("SELECT * FROM [Alunos-EE] WHERE idCartao=" +
id_EE_, con);

    dr = cmd.ExecuteReader();

    while (dr.Read())
    {
        id_EE = dr["idCartao"].ToString();
        nome_EE = dr["nome"].ToString();
        entrou_EE = dr["entrou"].ToString();
    }

    con.Close();

    string query = "insert into Registo(idCartao,nome, tipo
,dataEntrada,dataSaida,entrou) Values" +
        "(@paridCartao, @parnome, @parEntrada, @pardataSaida,@parEntrou)";

    SqlCommand comando = new SqlCommand(query, con);

    SqlParameter par1 = new SqlParameter("@paridCartao", SqlDbType.Int);
    par1.Value = id_EE;
    comando.Parameters.Add(par1);

    SqlParameter par2 = new SqlParameter("@parnome", SqlDbType.NVarChar);
    par2.Value = nome_EE;
    comando.Parameters.Add(par2);

    SqlParameter par6 = new SqlParameter("@partipo", SqlDbType.NVarChar);
    par6.Value = tipo;
    comando.Parameters.Add(par6);

    SqlParameter par3 = new SqlParameter("@parEntrada", SqlDbType.DateTime);
    par3.Value = DateTime.Now;
    comando.Parameters.Add(par3);

    SqlParameter par4 = new SqlParameter("@pardataSaida", SqlDbType.DateTime);
    if (entrou_EE == "0")
    {
        par4.Value = "";
    }
    else
    {
        par4.Value = DateTime.Now;
    }
}

```

```
comando.Parameters.Add(par4);

SqlParameter par5 = new SqlParameter("@parEntrou", SqlDbType.Int);
if (entrou_EE == "0")
{
    par5.Value = entrou_EE = "1";
}
else
{
    par5.Value = entrou_EE = "0";
}
comando.Parameters.Add(par5);

con.Open();

int num = comando.ExecuteNonQuery();

con.Close();

procura_R_EE_Aluno(id_EE_);
}

#endregion

#region "procura_EE_Aluno"
void procura_R_EE_Aluno(int id_EE_A)
{
    con.Close();
    con.Open();
    SqlCommand cmd = new SqlCommand("SELECT * FROM R_EE_Aluno WHERE idEE=" +
id_EE_A, con);

    dr = cmd.ExecuteReader();

    while (dr.Read())
    {
        id_aluno = dr["idA"].ToString();
        procura_aluno_geral(Convert.ToInt32(id_aluno));
    }

    con.Close();
}
#endregion

#region "procura_associado"

void procura_associado(int id_associadoo)
{
    con.Close();
    con.Open();
    SqlCommand cmd = new SqlCommand("SELECT * FROM Associado WHERE idAssociado=" +
id_associadoo, con);

    dr = cmd.ExecuteReader();

    while (dr.Read())
    {
        id_associado = dr["id_associado"].ToString();
        nome_associado = dr["nome"].ToString();
        entrou_associado = dr["entrou"].ToString();
    }

    con.Close();
}
```

```
string query = "insert into Regis-  
to(idCartao,nome,tipo,dataEntrada,dataSaida,entrou) Values" +  
"(@paridCartao, @parnome, @partipo, @parEntrada, @pardataSaida,@parEntrou);  
  
SqlCommand comando = new SqlCommand(query, con);  
  
SqlParameter par1 = new SqlParameter("@paridCartao", SqlDbType.Int);  
par1.Value = id_associado;  
comando.Parameters.Add(par1);  
  
SqlParameter par2 = new SqlParameter("@parnome", SqlDbType.NVarChar);  
par2.Value = nome_associado;  
comando.Parameters.Add(par2);  
  
SqlParameter par7 = new SqlParameter("@partipo", SqlDbType.NVarChar);  
par7.Value = tipo;  
comando.Parameters.Add(par7);  
  
SqlParameter par3 = new SqlParameter("@parEntrada", SqlDbType.DateTime);  
par3.Value = DateTime.Now;  
comando.Parameters.Add(par3);  
  
SqlParameter par4 = new SqlParameter("@pardataSaida", SqlDbType.DateTime);  
if (entrou_associado == "0")  
{  
    par4.Value = "";  
}  
else  
{  
    par4.Value = DateTime.Now;  
}  
comando.Parameters.Add(par4);  
  
SqlParameter par5 = new SqlParameter("@parEntrou", SqlDbType.Int);  
if (entrou_associado == "0")  
{  
    par5.Value = entrou_associado = "1";  
}  
else  
{  
    par5.Value = entrou_associado = "0";  
}  
comando.Parameters.Add(par5);  
  
con.Open();  
  
int num = comando.ExecuteNonQuery();  
  
con.Close();  
  
procura_R_EE_Aluno(id_associadoo);  
}  
  
#endregion
```