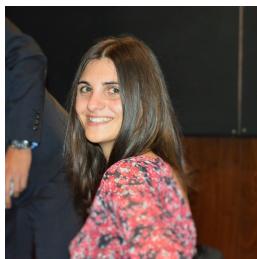


Escola de Engenharia
Universidade do Minho

DEPARTAMENTO DE ENGENHARIA INFORMÁTICA
Mestrado Integrado em Engenharia Informática
Desenvolvimento Sistemas de Software

System Cost – Partilha de Despesas num Apartamento

Fase Final



Célia Figueiredo
a67637



Gil Gonçalves
a67738



Márcia Costa
a67672



Daniel Rodrigues
a67634



Ricardo Lopes
a72062

Braga, 30 de Dezembro de 2016

Conteúdo

1	Introdução	2
1.1	Apresentação do Caso de Estudo	2
2	Requisitos do Sistema	4
2.1	Análise e Levantamento dos Requisitos	4
2.1.1	Criação do grupo com os elementos da casa/apartamento	4
2.1.2	Gestão das contas	4
2.2	Base de dados	5
2.2.1	Modelo conceitual	5
2.2.2	Modelo Lógico	6
2.2.3	Diagrama de Classes DAO	7
3	Arquitetura da Aplicação	8
3.1	Modelo de Domínio	8
3.2	Diagrama de Classes	9
3.3	Modelo de Use Case	10
3.3.1	Diagrama de Use Case	10
3.3.2	Subdiagrama Gerir Despesas	18
3.3.3	Subdiagrama Administrar Contas	24
3.3.4	Subdiagrama Interação com os Utilizadores	30
4	Implementação e Instalação do Sistema	34
4.1	Diagrama de Instalação	34
4.2	Diagrama de <i>Package</i>	35
4.3	Diagrama de Actividade	36
4.3.1	Diagrama de Atividade: Funcionamento da Aplicação	36
4.3.2	Adicionar Despesa/Fatura	37
4.3.3	Pagar Conta	38
4.3.4	Pagar Dívida	38
5	Interface do Sistema Gestão de Despesas - SGD	39
5.1	Máquinas de Estado	39
5.1.1	Efetuar Login	39
5.1.2	Criar Conta	39
5.1.3	Convidar Utilizadores	40
5.1.4	Remover Utilizadores	40
5.1.5	Inserir Fatura	41
5.1.6	Deposita Dinheiro	41

5.2	Mockups	42
5.3	Interação com utilizador	46
6	Conclusão	56

1. Introdução

No âmbito da unidade curricular de Desenvolvimento de Sistemas de Software do 3ºano do curso de MIEI, foi proposto o desenvolvimento de um projeto que visa a concepção de um sistema que serve de suporte à partilha de despesas num apartamento.

Neste relatório é descrito o processo de análise, modelação e concepção de um sistema que serve de suporte à partilha de despesas num apartamento. Foi-nos proposto desenvolver uma aplicação que fosse capaz de fazer o registo das despesas que são geradas num apartamento, assim como a gestão do pagamento feita por cada um dos moradores do apartamento em questão. Decidimos que seria interessante desenvolver um sistema que permitisse aos utilizadores usufruírem do controlo de terem as suas despesas devidamente divididas, onde os pagamentos das mesmas fossem o mais breve possível, organizado e ainda a facilidade de acesso através de um smartphone, tablet ou pc para a consulta desta mesma aplicação.

O trabalho será dividido em duas fases que se completam uma à outra. Na primeira fase será descrito o processo de análise de requisitos suportada pelo modelo de domínio do sistema, casos de uso e respectivas especificações e uma possível interface com o utilizador. Tudo isto de forma a enquadrar e descrever da forma mais detalhada possível o sistema a ser desenvolvido. Serão expostos os desenhos de planificação das interfaces e da sua correlação com as funcionalidades a serem implementadas no sistema.

Nesta segunda fase faremos com que a nossa aplicação ganhe vida e desta forma conseguirmos fazer chegar ao público alvo aquilo que seria o nosso produto final. Assim sendo, será necessário:

- criar uma base de dados que irá conter informação sobre os utilizadores da aplicação, bem como outras informações úteis como o histórico de despesas que deverá estar sempre atualizado;
- criar uma interface apelativa e funcional, a qual será criada à semelhança dos Mockups criados e apresentados na primeira fase;
- fazer a ligação dessa mesma interface à Base de Dados criada

Para além disso serão desenhados os modelos de Dominio, os Modelos de Domínio com DAO, os diagramas de package, os diagramas de instalação e os diagramas de atividade. Todos estes modelos servirão de base para a implementação do projeto.

1.1 Apresentação do Caso de Estudo

A aplicação terá como objetivo desenvolver um sistema de despesas num apartamento capaz de suportar o registo de despesas e a gestão do pagamento dessas mesmas despesas por parte dos moradores. Este é um sistema que proporciona aos seus utilizadores a possibilidade de

efetuarem os pagamentos das suas despesas, sejam estas recorrentes (por exemplo, água ou eletricidade) ou extraordinárias (por exemplo, necessidade de realizar alguma reparação no apartamento) fazendo com que o controlo de dívidas entre moradores estejam sempre atualizadas e visíveis para todos os utilizadores da aplicação. Neste sistema existe um morador previamente registado na aplicação necessitando de fornecer o nome, data de nascimento, e-mail e número de telefone/telemóvel e uma password para efetuar o registo. Terá a ele associada uma conta corrente que será uma espécie de fundo do qual mensalmente (ou quando necessário) é creditado o pagamento, ou seja, a quantia necessária afeta as despesas correspondentes. A conta corrente de cada morador contribui para o saldo global, ou seja, no nosso sistema existe um saldo que resulta do somatório de todas as contas correntes e que corresponde ao montante total que o apartamento tem como despesa nesse mês.

O saldo global é administrado por um Administrador que é responsável por verificar se o montante deste mesmo está completo e depois dessa forma, pagar a despesa. A todas as despesas está associado um valor assim como cada pagamento creditado da conta corrente de cada morador.

Ainda de salientar que a cada morador está associada uma fração que varia consoante o tipo de morador (exemplo: moradores que partilham quarto, terão uma fração do valor da renda menor relativamente a um morador com quarto individual), assim como relativamente às restantes despesas (água, luz, gás, internet, ou até mesmo despesas extraordinárias).

2. Requisitos do Sistema

2.1 Análise e Levantamento dos Requisitos

A aplicação a desenvolver deverá suportar o registo das despesas e a gestão do seu pagamento por parte de moradores registados.

2.1.1 Criação do grupo com os elementos da casa/apartamento

- Cada morador deve efetuar um registo fornecendo o seu nome, e-mail, número de telemóvel e data de nascimento.
- O morador assim como o utilizador necessitam de efetuar login na aplicação
- O utilizador que convidar os restantes será considerado administrador do sistema.

2.1.2 Gestão das contas

- Após o registo será associada uma conta corrente a cada utilizador.
- Cada conta corrente permitirá a gestão do saldo corrente (estão incluídas dívidas) de cada morador.
- O morador deverá efetuar um pagamento.
- Cada pagamento será creditado na conta corrente do morador.
- Cada pagamento efetuado não pode ultrapassar o valor da fatura
- O pagamento pode ser igual ou superior à quantia necessária para pagar as despesas em causa.
- Existirá um saldo global da casa, este que é o somatório de todas as contas correntes dos moradores
- O saldo global é administrado por um Administrador
- Existem dois tipos de despesas distintos: a despesa recorrente referente a despesas mensais como a água, luz, renda, etc; a despesa extra referente a por exemplo arranjos de material na casa
- Cada morador deverá pagar uma fração relativa à despesa
- Cada despesa a pagar tem a si associado um tipo (i.e. água, luz, cadeiras, etc)

2.2 Base de dados

2.2.1 Modelo conceitual

A aplicação necessitará de uma implementação de uma base de dados para gerir os elementos que pertencem ao grupo assim como as despesas efetuadas pelos moradores. A modelação conceitual consiste no processo de construção de um modelo de dados a partir de uma análise detalhada dos requisitos. Este modelo é completamente independente de todas as considerações físicas, ou seja é independente dos detalhes de implementação. De seguida está apresentado o esquema conceitual relativo ao projeto.

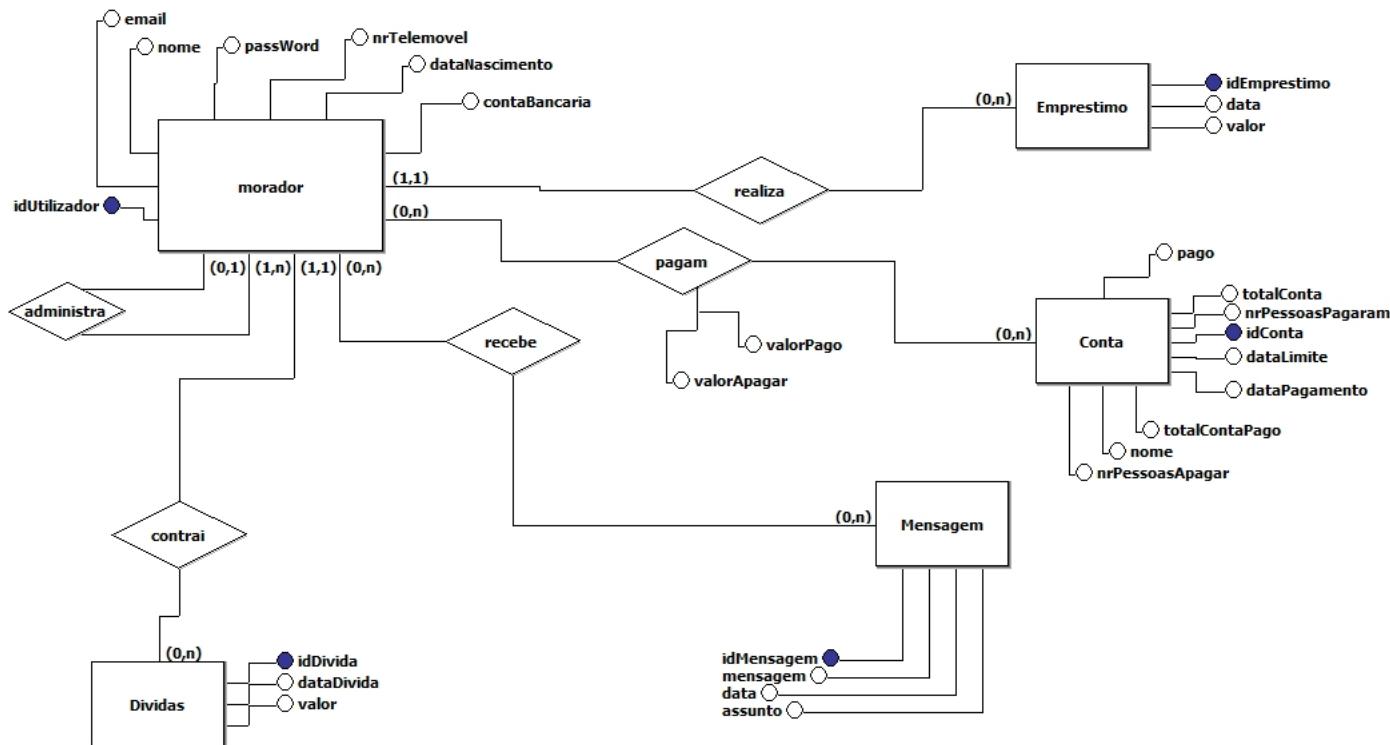


Figura 2.1: Modelo Conceitual

2.2.2 Modelo Lógico

Terminada a modelação conceitual, procedemos à fase de modelação lógica. Esta consiste na tradução do modelo conceitual para o lógico capaz de representar os requisitos definidos, assim como a validação do mesmo. Este modelo está representado na figura abaixo:

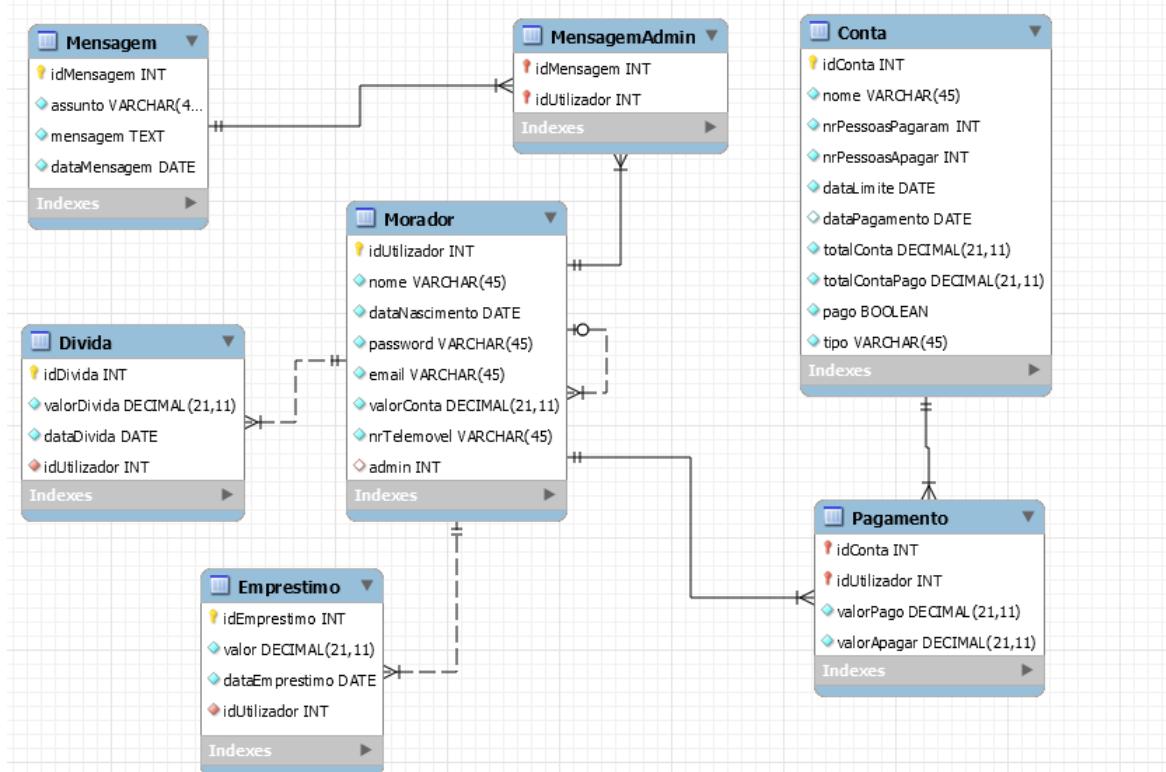


Figura 2.2: Modelo Lógico

2.2.3 Diagrama de Classes DAO

Para a realização deste projeto, construímos uma base de dados relacional em *MySQL* para armazenar toda a informação necessária para colocar o SGD¹ a correr. Para o programa aceder a essa informação, foram classes *DAO*², que fornecem métodos de acesso à base de dados. Cada método visível para o exterior desta classe solicita sempre uma conexão à base de Dados, este pedido é feito à classe *Connector*, podendo este conector estar configurado com Auto-Commit ou não dependendo do tipo de operações pretendidas. São também definidos métodos auxiliares alguns *Private* e outros *Protected* que executam as queries à base de dados, contêm *PreparedStatements* por questões de segurança contra *SQL injection*. A criação destes métodos deve-se à tentativa de minimizar as conexões simultâneas à Base de Dados.

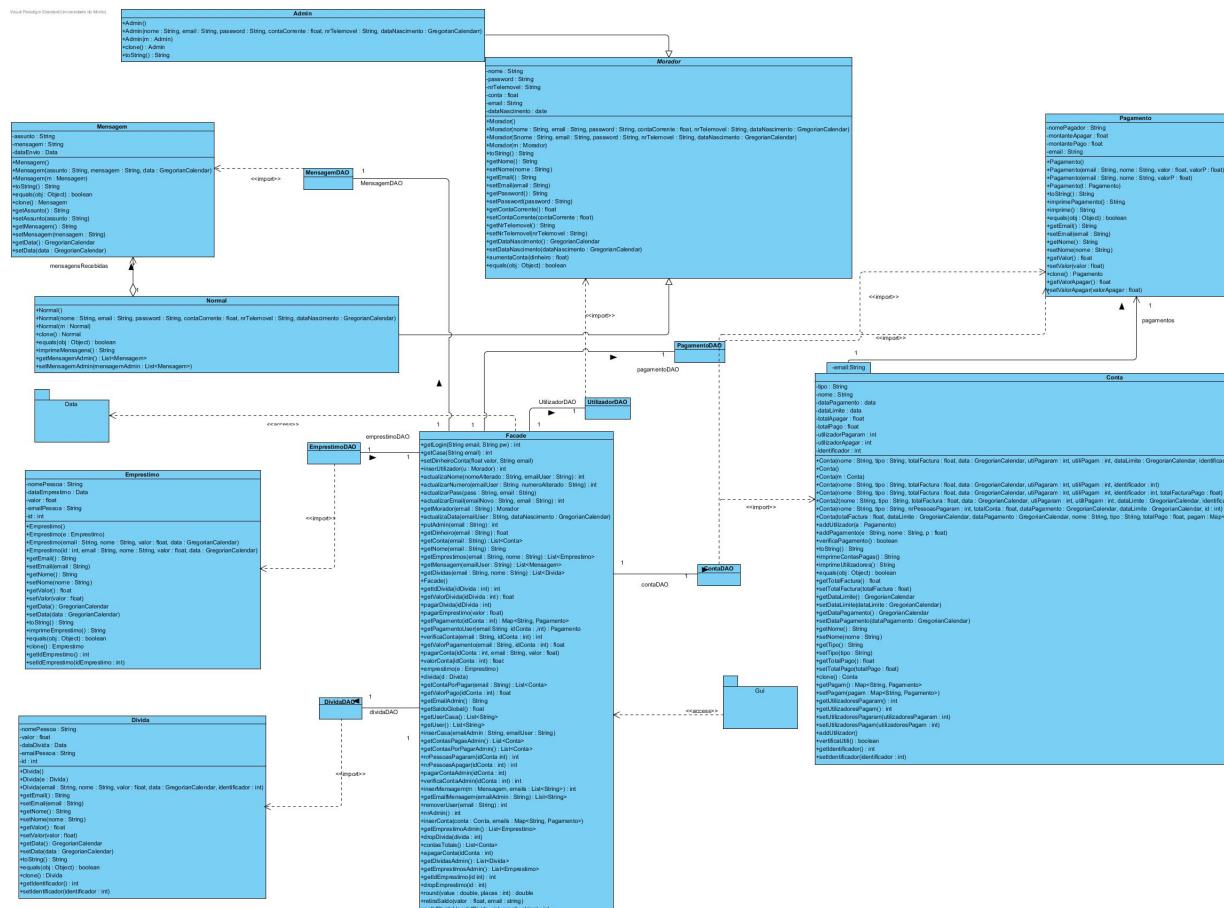


Figura 2.3: Diagrama de Classes DAO

¹Sistema de Gestão de Despesas

²Data Access Object

3. Arquitetura da Aplicação

3.1 Modelo de Domínio

Todo e qualquer projeto possui um domínio específico. O modelo de domínio deve capturar os seguintes pontos: as entidades, os relacionamentos entre as entidades e o vocabulário de domínio do problema. Para além disso também deve ser uma visão estática do problema onde é possível representar as regras de negócio invariantes no tempo. Ou seja, o modelo de domínio é a base para a análise de requisitos.

No que diz respeito à aplicação, como é dito na introdução, queremos desenvolver uma aplicação capaz de suportar o registo das despesas e a gestão do seu pagamento por parte dos moradores registados. Assim sendo com base nos requisitos iniciais construimos o modelo de domínio do Sistema de Gestão de Despesas (SGD).

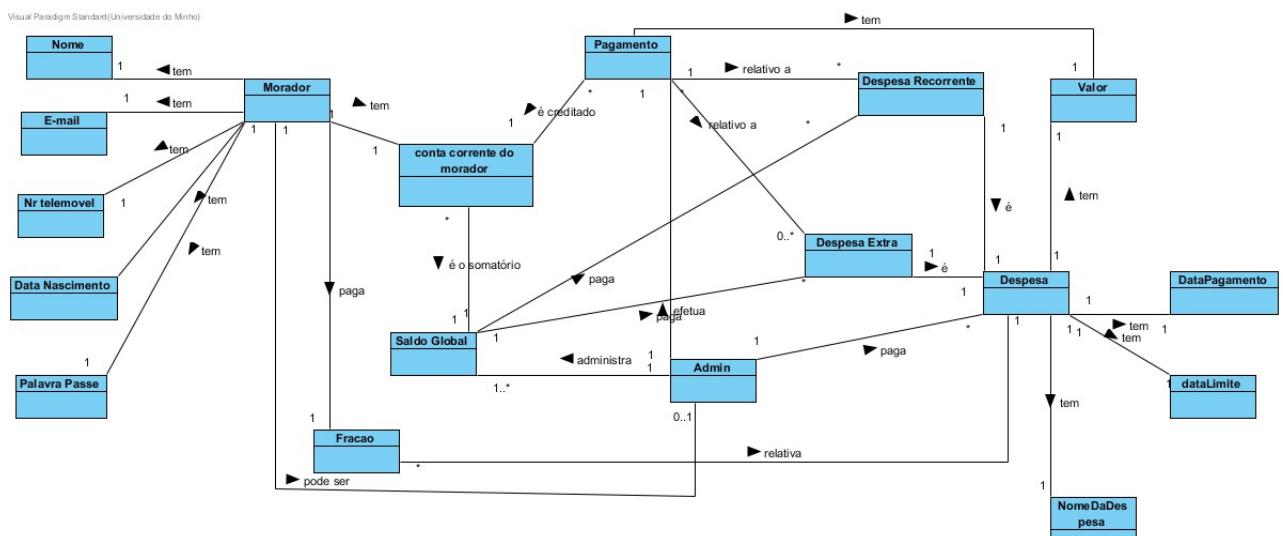


Figura 3.1: Modelo Dominio

O morador necessita de fornecer o nome, e-mail, número de telemovel e data de nascimento, para se registrar.

Para efetuar login será necessário inserir o e-mail e a palavra passe.

Como se pode observar na figura o morador efetua pagamento relativo a despesa recorrente ou extra, assim como paga uma fração da despesa, essa fração é relativa a um tipo de despesa.

O administrador administra o saldo global da casa/apartamento.

3.2 Diagrama de Classes

A metodologia de implementação das boas práticas do UML no desenvolvimento de qualquer projecto, especialmente em projetos de media e pequena escala, para possíveis implementações de ideias da camada de negócio e de esquemas concetuais, que podem ser validados e discutidos pelo grupo de desenvolvimento e analisados sem que seja necessário perder tempo com testes e *builds*. No caso específico do diagrama de classes aquilo que concluímos é que permite de uma forma simples e transparente partilhar e discutir como irá ser, ou pelos menos como se pretende que seja, os esquemas concetuais do modelo de domínio e esqueleto da implementação, ou seja, das estrutura básica das classes. O modelo abaixo apresentado é portanto uma tradução daquilo que se pretendia com o modelo de domínio, mas nesse caso, com maior detalhe técnico relativos à linguagem de programação escolhida, i.e. o Java. Temos portanto o seguinte diagrama de classes:

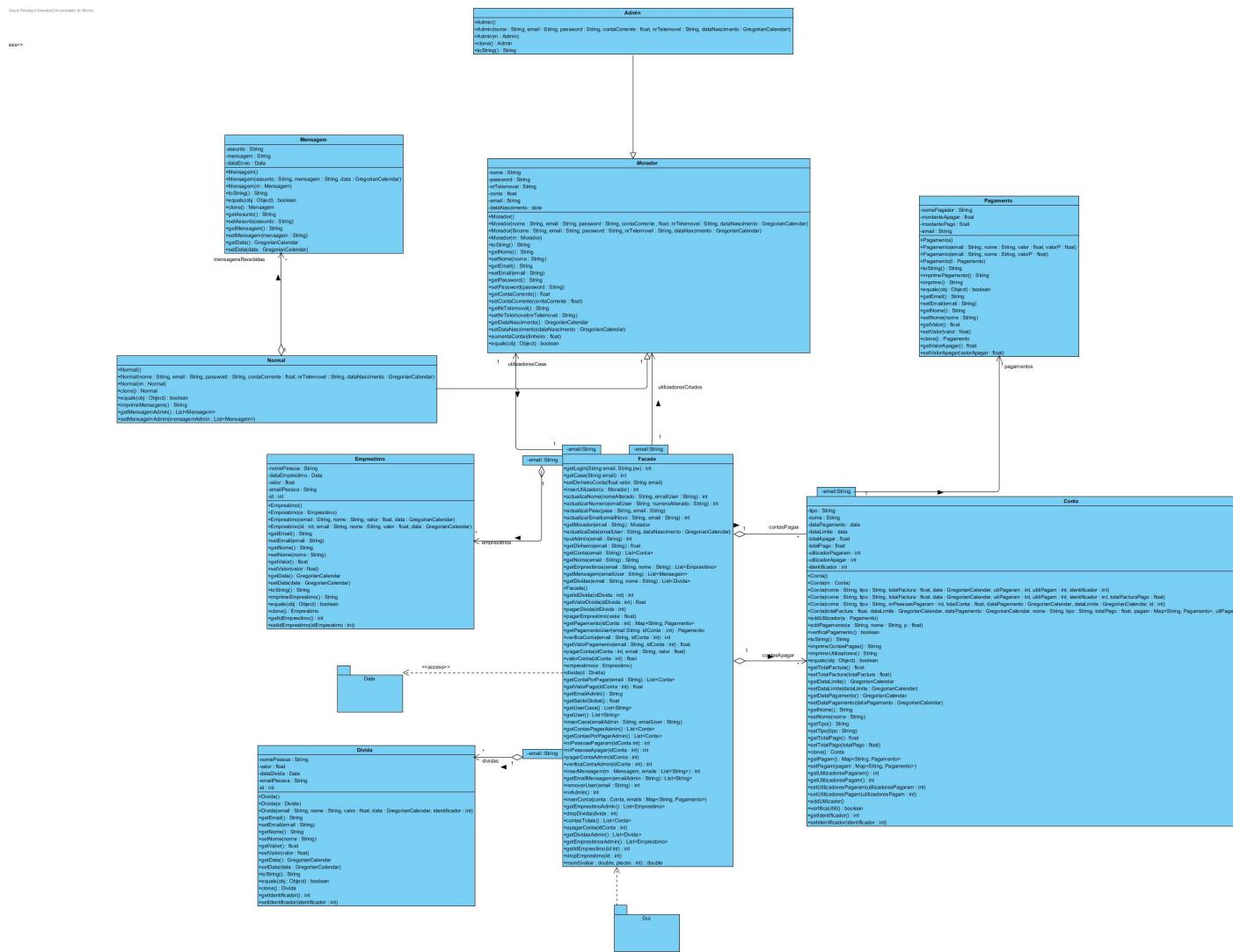


Figura 3.2: Diagrama de Classes

3.3 Modelo de Use Case

A segunda parte da análise de requisitos corresponde à definição dos *use cases*, com o objetivo de os aplicar numa primeira fase do projeto. Primeiramente é necessário identificar os atores, que serão quem interagirá com o sistema. Posterior à identificação dos atores, passamos então à identificação dos *use cases*, isto é, o que se pretende do sistema, que corresponde à especificação da funcionalidade a ser implementada. Neste sentido, quando definimos um *use case*, para além de ser uma espécie de documentação, temos de ter em conta que se trata de uma unidade coerente de funcionalidade, um serviço. Define também um comportamento do sistema, sem revelar a estrutura interna, divulgando desta forma, a comunicação entre os atores e o sistema. O conjunto de todos os *use cases* acaba por definir pela íntegra, toda a funcionalidade do sistema que decorre na sua essência, do diálogo entre o sistema e os atores, e a responsabilidade de resposta funcional do sistema.

3.3.1 Diagrama de Use Case

Neste modelo de *Use Case* o apelo visual permite literalmente desenhar o processo de execução do negócio e visualizar a responsabilidade de cada participante, quando entra em cena, qual a sua interação e a sequência em que o seu trabalho precisa ser realizado em relação às responsabilidades e tarefas dos restantes integrantes do processo. Neste projeto existem os atores "Admin" e "Morador" estes que desempenharam tarefas no sistema. Um caso de uso representa uma unidade discreta da interação entre um utilizador e o sistema. Um caso de uso é uma unidade de um trabalho significante. Por exemplo: o "Efetuar login", "Criar conta", "gerir despesas", "Administrar Contas", "Consultar MensagensDepositar dinheiro para a sua conta", entre outros, são todos casos de uso. Cada caso de uso tem uma descrição, esta que descreve a funcionalidade que irá ser construída no sistema proposto.

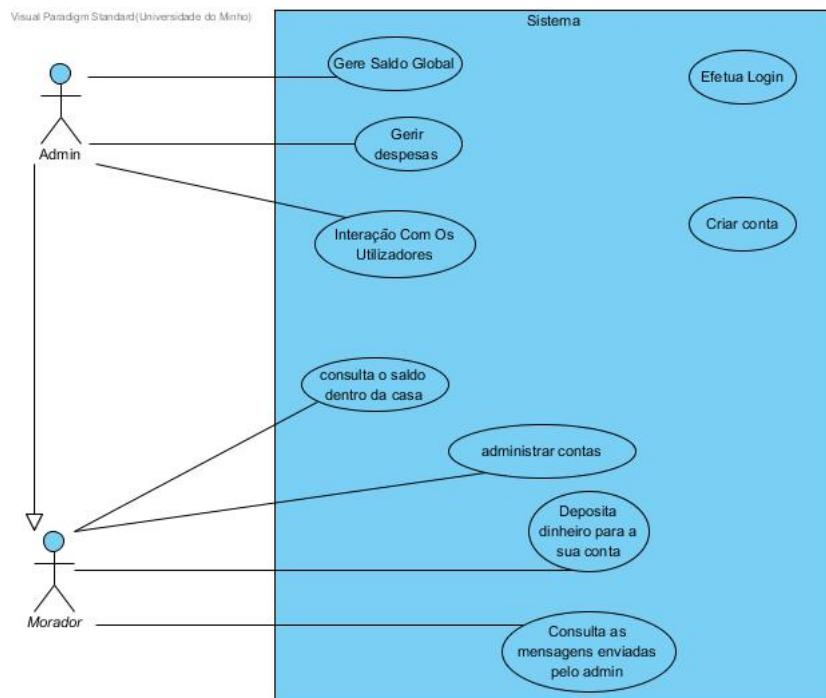


Figura 3.3: Modelo Use Case

Especificação: Gere Saldo Global

Esta interação pressupõe que o administrador tenha o login previamente efetuado e desta forma, a visualização do saldo global ao Admin será simples e clara. É então desta forma que o Admin tem acesso ao valor que consta no saldo global, onde esse valor é simplesmente dado a conhecer pelo sistema. O saldo global a que nos referimos é o resultado do somatório de todas as contas correntes de cada morador do apartamento em questão. Este Use Case apenas dá a conhecer ao Admin esse valor.

O use case em formato tabular do Gere saldo Global e o seu respectivo diagrama de sequência é o seguinte:

Date	2016	
Brief Description		
Preconditions	Admin tem login efetuado	
Post-conditions		
Flow of Events	Actor Input	System Response
1		o sistema apresenta o dinheiro do saldo
2	o admin vê o dinheiro	
3		

Figura 3.4: Especificação do Use Case: Gere Saldo Global

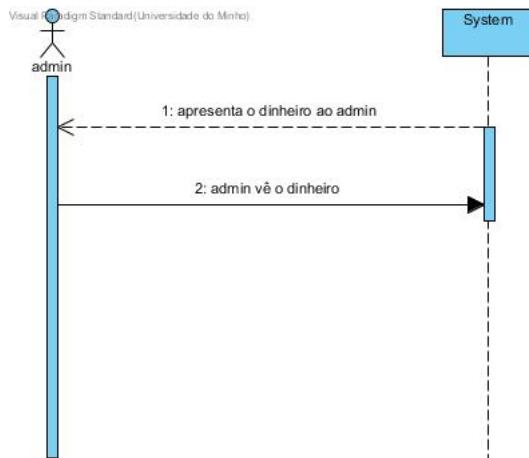


Figura 3.5: Diagrama de Sequência: Gere Saldo Global

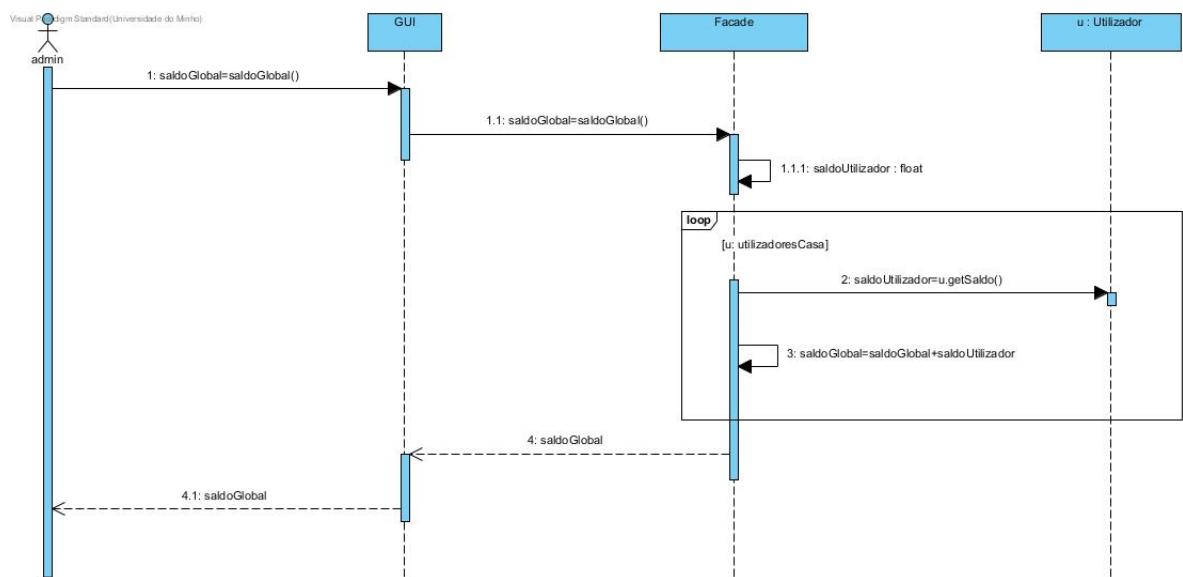


Figura 3.6: Diagrama de Iteração: Gere Saldo Global

Especificação: Efetua Login

Cada morador tem que efetuar Login na aplicação, para tal, é necessário que o indivíduo tenha conta previamente criada. Como em qualquer aplicação, este passo é fundamental e não poderia deixar de o ser no nosso sistema. Para se efetuar o registo do login é solicitado pelo sistema o endereço de email do utilizador em questão para sua validação. Após introduzir o email o sistema requer também o preenchimento da respetiva password. Optamos por colocar um endereço de email ao invés de um username ou um número de telemóvel porque assim poderá existir a possibilidade de futuramente serem enviados para o email dos utilizadores da aplicação, notificações relativas às despesas e ou pagamentos.

Preconditions	Ter criado conta	
Post-conditions	Ter efectuado o login	
Flow of Events		
	Actor Input	System Response
1		o sistema pede que introduza o email
2	o utilizador introduz o email	
3		o sistema valida o email
4		o sistema pede que introduza a palavra passe
5	o utilizador introduz a palavra passe	
6		o sistema valida a palavra passe
7		Login efectuado com sucesso
Exceção 1 [email invalido] (passo 3)		
1		o sistema informa que o email é invalido
Exceção 2 [palavra passe incorrecta] (passo 6)		
1		o sistema informa que a palavra passe é invalida

Figura 3.7: Especificação do Use Case: Efetua Login

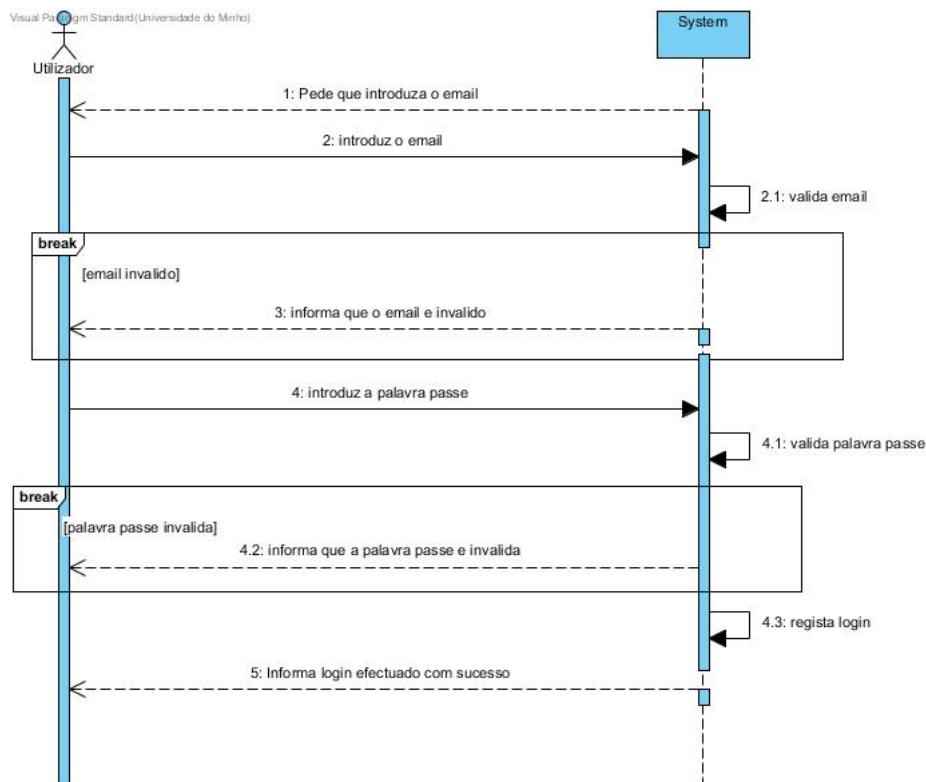


Figura 3.8: Diagrama de Sequência: Efetua Login

Especificação: Criar Conta

Esta interação é uma interação bastante vulgar no sentido em que praticamente todas as aplicações o ato de “criar conta” é quase obrigatório. No nosso sistema este passo é fundamental para os utilizadores poderem utilizar a nossa aplicação. Para um utilizador criar conta tem que mencionar determinados dados ao sistema e este tem aceitar estes mesmos dados. Esses dados são simples e optamos por apenas solicitar o nome, data de nascimento, e-mail, número de telemóvel e password.

Post-conditions	Conta Criada	
Flow of Events	Actor Input	System Response
	1	Inserir Nome Completo
	2 Utilizador insere nome	
	3	Inserir data de nascimento
	4 Utilizador insere data de nascimento	
	5	Inserir email
	6 insere mail	
	7	verifica e-mail
	8	pede para inserir password
	9 insere password	
	10	verifica que password
	11	pede para inserir nr de telemovel
	12 insere nr de telemovel	
	13	Regista dados
Excecao 1 (passo 7) [E-mail inválido]	Actor Input	System Response
	1	sistema informa que mail é inválido
Excecao 2 (passo 10) [password não cumpre requisitos]	Actor Input	System Response
	1	informa que password não cumpre requisitos mínimos

Figura 3.9: Especificação do Use Case: Criar Conta

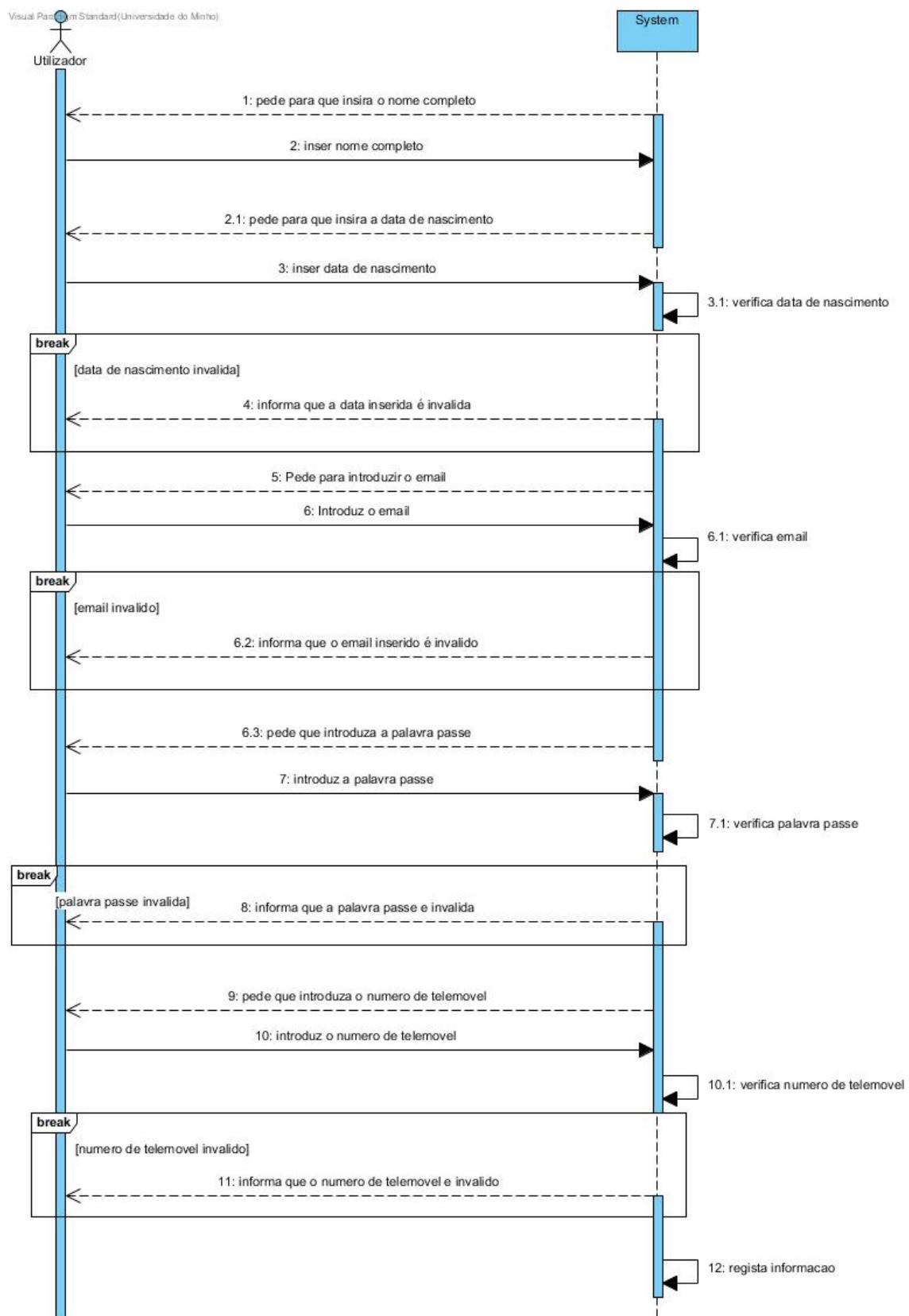


Figura 3.10: Diagrama de Sequência: Criar Conta

Especificação: Consulta Saldo dentro de casa

Este use case apenas é útil para a consulta do saldo dentro do apartamento, ou seja, existe a possibilidade de consultar o valor do saldo que consta no apartamento. Cada utilizador que tenta aceder a este saldo, tem que ter o login feito.

Date	23/out/2016 23:05:17									
Brief Description										
Preconditions	O utilizador ter feito o login									
Post-conditions										
Flow of Events	<table border="1"> <thead> <tr> <th></th> <th>Actor Input</th> <th>System Response</th> </tr> </thead> <tbody> <tr> <td>1</td> <td></td> <td>o sistema apresenta o saldo dentro da casa</td> </tr> <tr> <td>2</td> <td>o utilizador visualiza esse dinheiro</td> <td></td> </tr> </tbody> </table>		Actor Input	System Response	1		o sistema apresenta o saldo dentro da casa	2	o utilizador visualiza esse dinheiro	
	Actor Input	System Response								
1		o sistema apresenta o saldo dentro da casa								
2	o utilizador visualiza esse dinheiro									

Figura 3.11: Especificação do Use Case: Consulta Saldo dentro de Casa

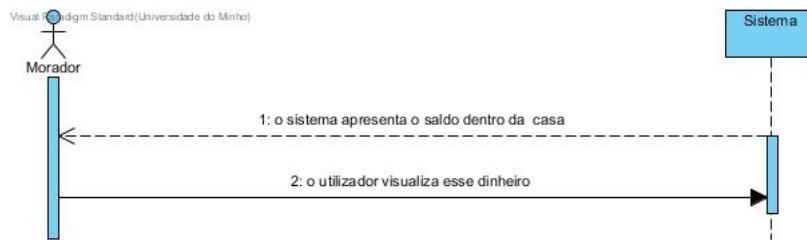


Figura 3.12: Diagrama de Sequência: Consulta saldo de casa

Especificação: Deposita Dinheiro para a sua Conta

Esta especificação foi feita a pensar no depósito da conta corrente de cada morador/utilizador da aplicação, assim como, a atualização da conta corrente. Basicamente, o sistema solicita que o utilizador adicione determinada quantia na conta corrente e após este procedimento ser efetuado com sucesso a conta corrente surge atualizada. Este processo requer que o utilizador tenha o login efetuado, pois só desta forma é possível aceder a estas especificações.

Date	17/Out/2016 16:16:28												
Brief Description													
Preconditions	login efetuado												
Post-conditions	conta corrente atualizada												
Flow of Events	<table border="1"> <thead> <tr> <th></th> <th>Actor Input</th> <th>System Response</th> </tr> </thead> <tbody> <tr> <td>1</td> <td></td> <td>Pede para inserir quantia a adicionar</td> </tr> <tr> <td>2</td> <td>insere quantia</td> <td></td> </tr> <tr> <td>3</td> <td></td> <td>Atualiza conta corrente</td> </tr> </tbody> </table>		Actor Input	System Response	1		Pede para inserir quantia a adicionar	2	insere quantia		3		Atualiza conta corrente
	Actor Input	System Response											
1		Pede para inserir quantia a adicionar											
2	insere quantia												
3		Atualiza conta corrente											

Figura 3.13: Especificação do Use Case: Deposita Dinheiro para a sua conta

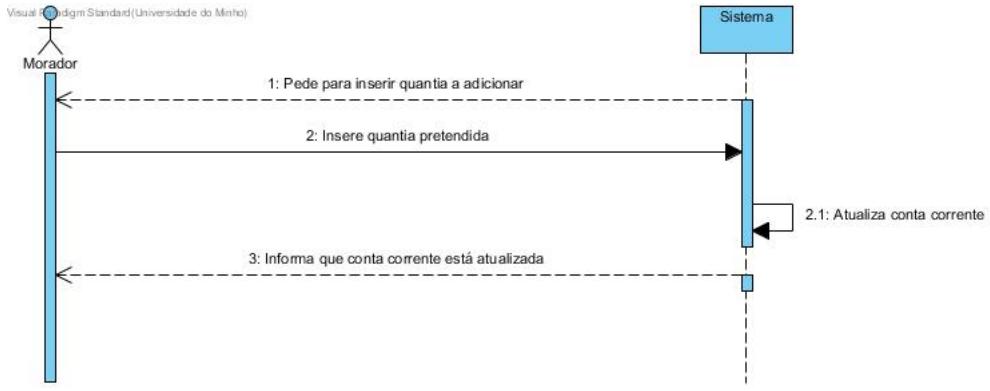


Figura 3.14: Diagrama de Sequência: Deposita Dinheiro na Conta

Especificação: Consulta Mensagens enviadas pelo Administrador

O use case “Consulta as Mensagens enviadas pelo Admin” foi criado a pensar nas ações do admin. Neste, em particular, o administrador é o único responsável por enviar mensagens aos restantes utilizadores da aplicação. Para uma consulta mais eficiente e simples o utilizador tem a facilidade de optar qual mensagem pretende ler. Estas mensagens são as mensagens que o admin envia com os valores em falta, por exemplo. Cada utilizador é notificado que tem uma mensagem por ler e apenas acede a ela e fica no imediato com a consulta efetuada.

Date	23/out/2016 21:54:07												
Brief Description													
Preconditions	Ter o login efectuado												
Post-conditions	Consegue consulta as mensagens recebidas												
Flow of Events	<table border="1"> <thead> <tr> <th></th> <th>Actor Input</th> <th>System Response</th> </tr> </thead> <tbody> <tr> <td>1</td> <td></td> <td>o sistema apresenta as mensagens enviadas pelo admin</td> </tr> <tr> <td>2</td> <td>o utilizador escolhe a mensagem que quer ler</td> <td></td> </tr> <tr> <td>3</td> <td></td> <td>o sistema apresenta a mensagem selecionada</td> </tr> </tbody> </table>		Actor Input	System Response	1		o sistema apresenta as mensagens enviadas pelo admin	2	o utilizador escolhe a mensagem que quer ler		3		o sistema apresenta a mensagem selecionada
	Actor Input	System Response											
1		o sistema apresenta as mensagens enviadas pelo admin											
2	o utilizador escolhe a mensagem que quer ler												
3		o sistema apresenta a mensagem selecionada											
Exceção(passo 1) [Não existem mensagens]	<table border="1"> <thead> <tr> <th></th> <th>Actor Input</th> <th>System Response</th> </tr> </thead> <tbody> <tr> <td>1</td> <td></td> <td>O sistema informa que não existem mensagens para apresentar</td> </tr> </tbody> </table>		Actor Input	System Response	1		O sistema informa que não existem mensagens para apresentar						
	Actor Input	System Response											
1		O sistema informa que não existem mensagens para apresentar											

Figura 3.15: Especificação do Use Case: Consulta Mensagens enviadas pelo Administrador

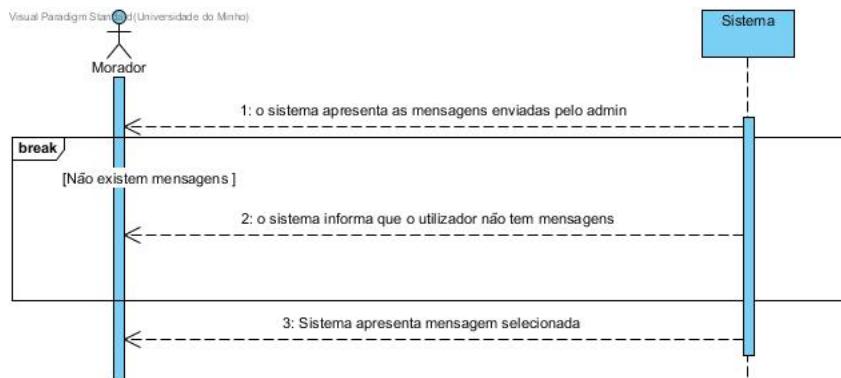


Figura 3.16: Diagrama de Sequência: Consulta Mensagens

3.3.2 Subdiagrama Gerir Despesas

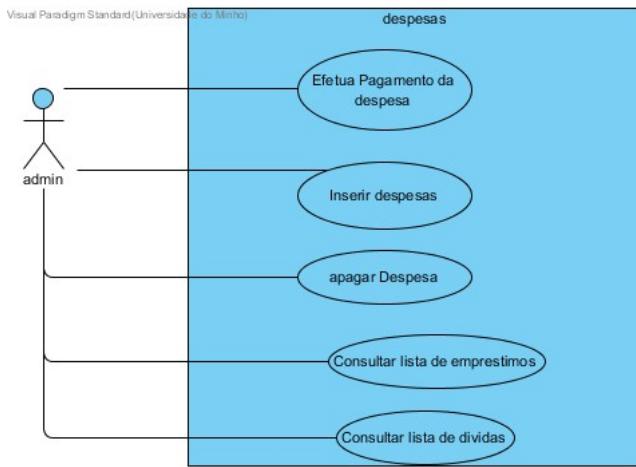


Figura 3.17: Sub-Diagrama Gerir Despesas

Especificação: Efetua pagamento da despesa

De forma a organizar a nossa aplicação apresentamos as despesas que estão por pagar por lista de onde são visualizadas as despesas que estão por pagar e assim fazendo um clique é possível aceder a uma determinada fatura seleccionada pelo utilizador. A despesa em falta é automaticamente descontada ao saldo global do apartamento em causa. Deste modo, sabemos sempre se existem faturas por pagar e o saldo é actualizado no imediato. Claro está , que é necessário que o valor do saldo seja sempre igual ou superior aos valores indicados nas faturas.

Author	Célia Figueiredo	
Date	17/Oct/2016 16:17:24	
Brief Description		
Preconditions	Ter dinheiro suficiente para pagar a despesa e ter efectuado o login	
Post-conditions	Pagamento efectuado	
Flow of Events	Actor Input	System Response
	1	O sistema apresenta as faturas que estão para ser pagas
	2 escolhe fatura para pagar	
	3	Apresenta valor a pagar
	4 Confirma que pagou (Ok)	
	5	Sistema Regista pagamento e desconta o valor da fatura no saldo global
Exceção (passo)	Actor Input	System Response
	1	O sistema informa que não existe faturas para serem pagas

Figura 3.18: Especificação do Use Case:Efetua Pagamento da despesa

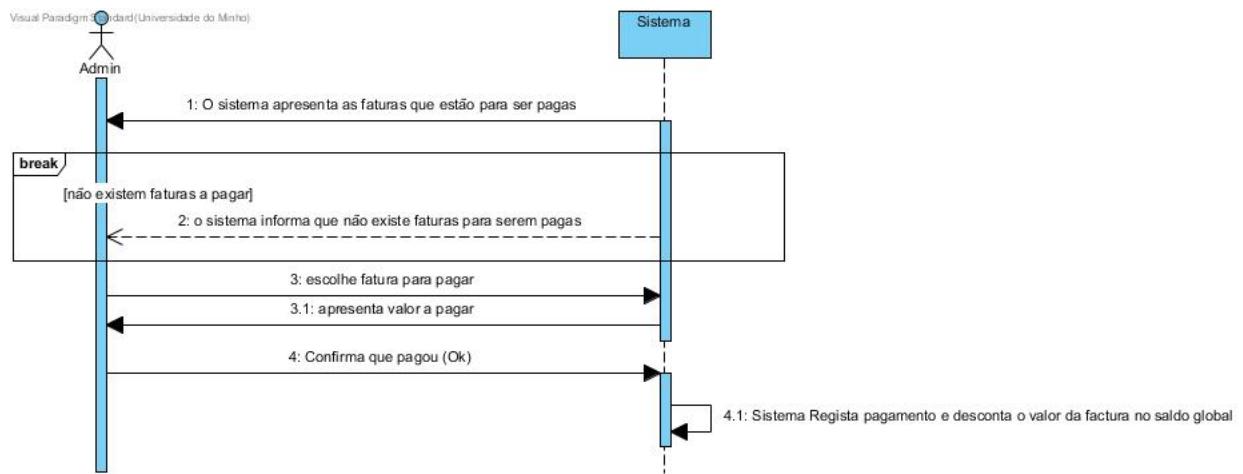


Figura 3.19: Diagrama de Sequência: Administrador efetua pagamento da despesa

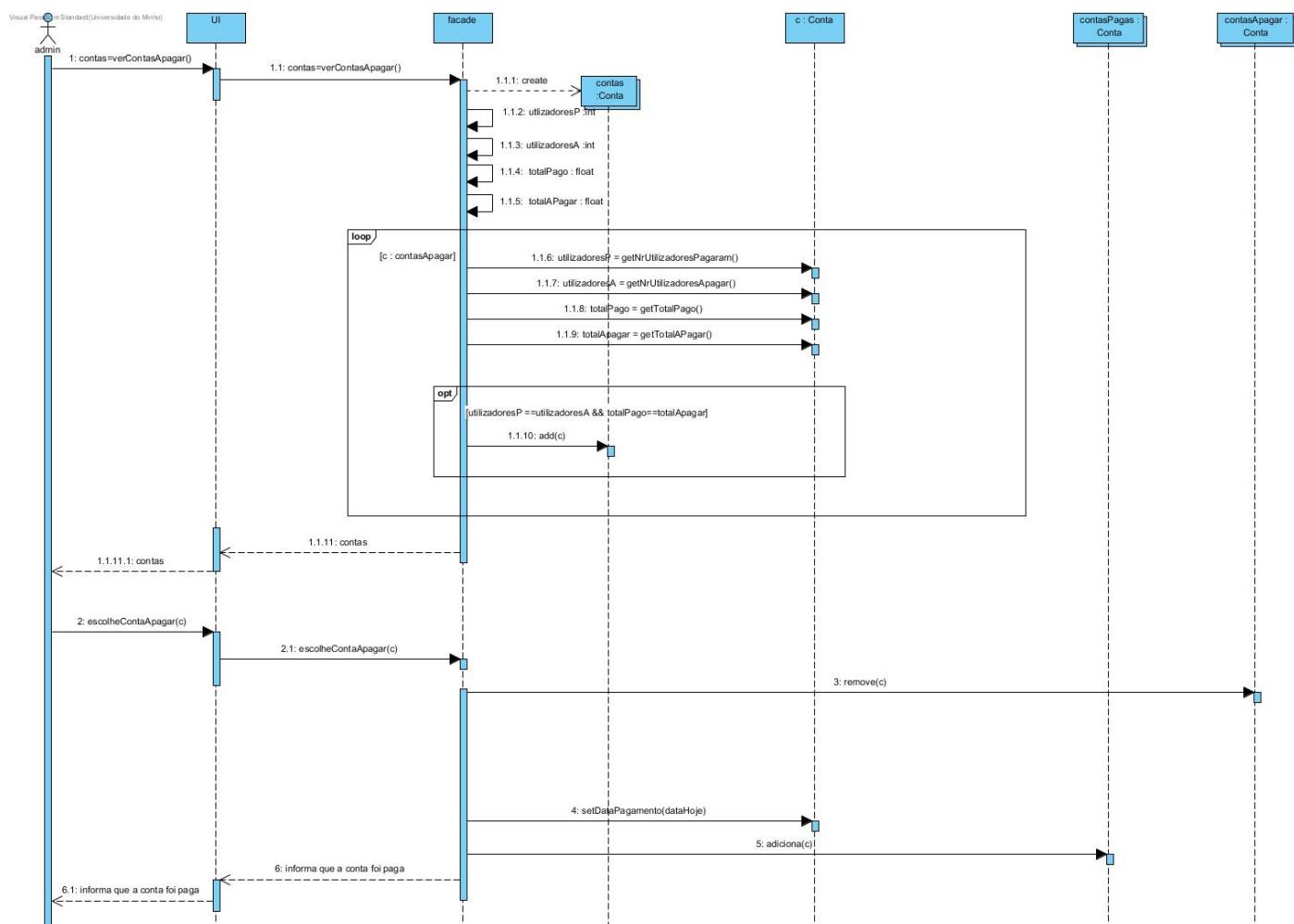


Figura 3.20: Diagrama de Iteração: Administrador efetua pagamento

Especificação: Inserir Despesa

Cabe ao administrador fazer as inserções das despesas no sistema. Para que cada inserção seja feita com sucesso têm que ser bem introduzidas e aceites pelo sistema, colocando a data, o nome e os valores de cada fatura válidos. É também tarefa do administrador decidir que percentagem cada morador registado tem que pagar de cada fatura/despesa.

Preconditions	o admin está autenticado	
Post-conditions	despesa inserida	
Flow of Events	Actor Input	System Response
	1	O sistema pede que introduza o nome da fatura
	2 Insere nome da fatura	
	3	O sistema pede que introduza a data da fatura
	4 o utilizador introduz a data	
	5	O sistema valida a data
	6	O sistema pede que introduza o valor da fatura
	7 Insere valor da fatura	
	8	O sistema valida o valor
	9	O sistema apresenta os tipos possíveis da fatura
	10 o utilizador escolhe o tipo normal	
	11	O sistema envia os valores para os utilizadores
	12	a fatura é introduzida com sucesso
	13	
Exceção (passo 4 [O sistema não valida a data])	Actor Input	System Response
	1	O sistema informa que não consegue validar a data introduzida
Exceção (passo 7) [o sistema não valida o valor introduzido]	Actor Input	System Response
	1	O sistema informa que o valor introduzido é inválido
Comp.Alternativo (passo 2) [Inser despesa extra]	Actor Input	System Response
	1	O sistema apresenta os utilizadores possíveis para pagarem
	2 o admin escolhe os utilizadores	
	3	O sistema pede que introduza a percentagem a pagar
	4 o admin introduz a percentagem que cada um tem de pagar	
	5	regressa 12

Figura 3.21: Especificação do Use Case: Inserir Despesas

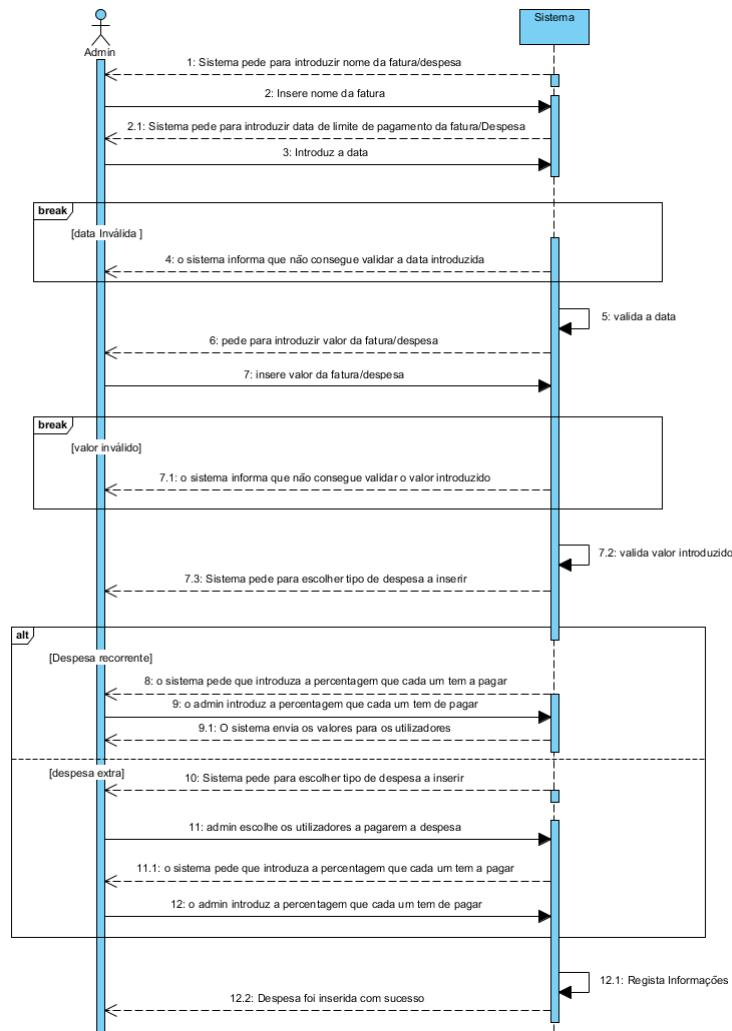


Figura 3.22: Diagrama de Sequência: Inserir Despesa

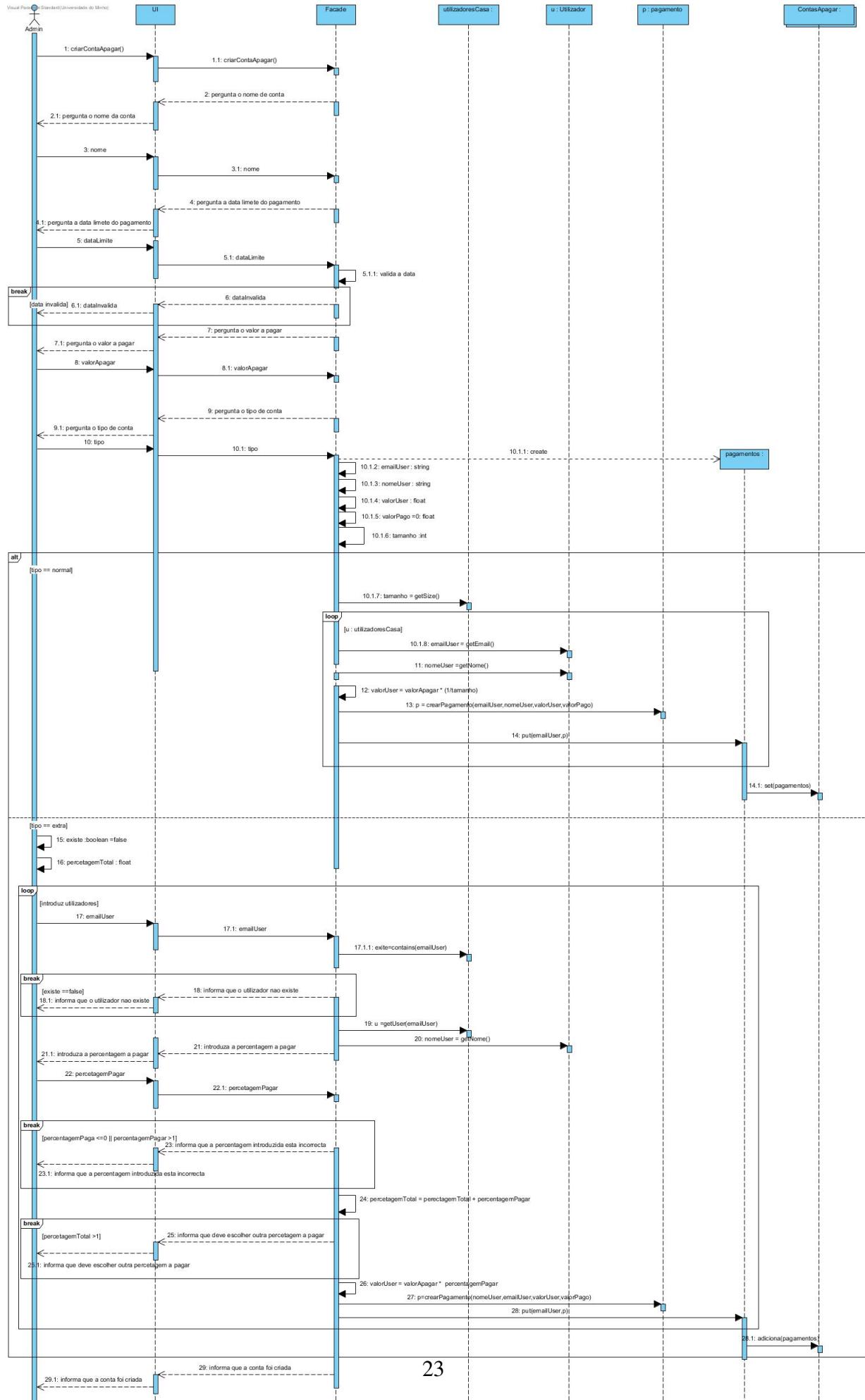


Figura 3.23: Diagrama de Iteração: Inserir Despesa

Especificação: Apagar Despesa

O administrador tanto insere as despesas no sistema como as elimina. Temos apenas que ter em consideração que é condição necessária que o admin tenha o login efetuado e a partir daqui o sistema apresenta a listas das despesas por pagar o administrador apenas selecciona aquela que efetivamente pretende que seja eliminada.

Date	8/jun/2016 1:00:26																		
Brief Description																			
Preconditions	o admin tem o login efectuado																		
Post-conditions																			
Flow of Events	<table border="1"> <thead> <tr> <th></th> <th>Actor Input</th> <th>System Response</th> </tr> </thead> <tbody> <tr> <td>1</td> <td></td> <td>o sistema apresenta as despesas por pagar</td> </tr> <tr> <td>2</td> <td>o utilizador escolhe a despesa que quer apagar</td> <td></td> </tr> <tr> <td>3</td> <td></td> <td>o sistema apaga a despesa</td> </tr> </tbody> </table> <table border="1"> <thead> <tr> <th></th> <th>Actor Input</th> <th>System Response</th> </tr> </thead> <tbody> <tr> <td>1</td> <td></td> <td>o sistema informa que não existe despesa por pagar</td> </tr> </tbody> </table>		Actor Input	System Response	1		o sistema apresenta as despesas por pagar	2	o utilizador escolhe a despesa que quer apagar		3		o sistema apaga a despesa		Actor Input	System Response	1		o sistema informa que não existe despesa por pagar
	Actor Input	System Response																	
1		o sistema apresenta as despesas por pagar																	
2	o utilizador escolhe a despesa que quer apagar																		
3		o sistema apaga a despesa																	
	Actor Input	System Response																	
1		o sistema informa que não existe despesa por pagar																	

Figura 3.24: Especificação do Use Case: Apagar Despesa

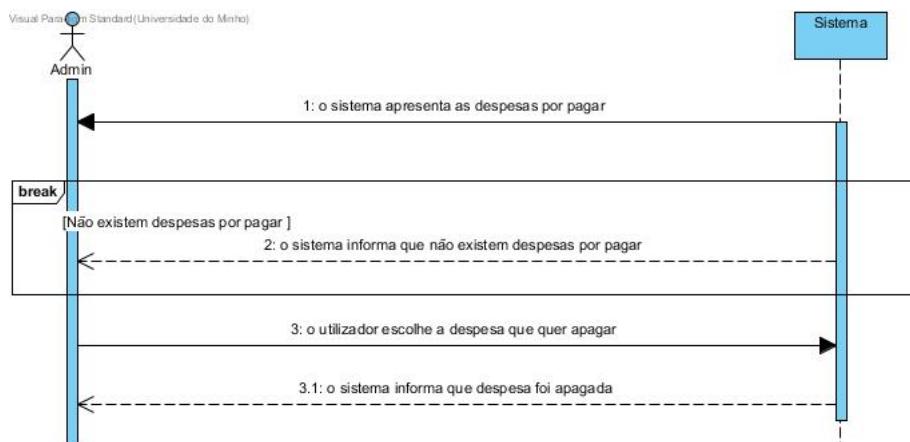


Figura 3.25: Diagrama de Sequência: Apagar Despesa

3.3.3 Subdiagrama Administrar Contas

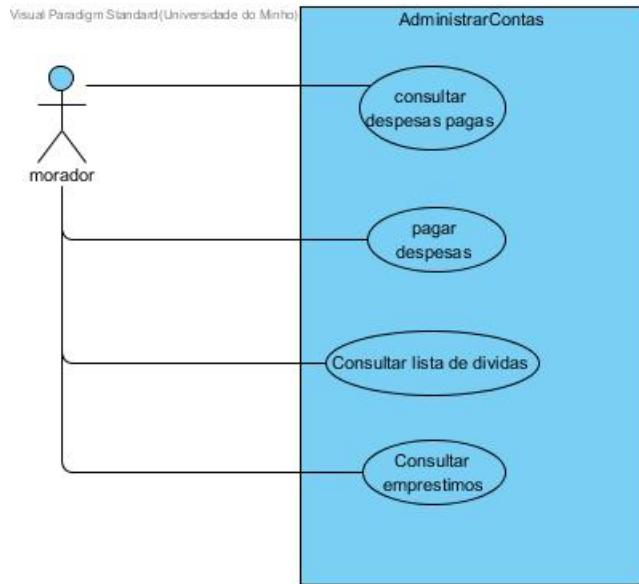


Figura 3.26: Subdiagrama: Administrar Contas

Especificação: Consultar despesas pagas

Esta é uma consulta feita pelo administrador. Fica possível a consulta de cada uma das despesas que já foram pagas. Esta interação apenas serve para consultar informação. O sistema apresenta todos os detalhes da fatura previamente seleccionada pelo admin. Como o próprio nome do subdiagrama indica, trata-se de administrar contas.

Date	23/out/2016 22:45:49												
Brief Description													
Preconditions	Ter efectuado o login												
Post-conditions	Consegue visualizar a fatura pretendida												
Flow of Events	<table border="1"> <thead> <tr> <th></th> <th>Actor Input</th> <th>System Response</th> </tr> </thead> <tbody> <tr> <td>1</td> <td></td> <td>Apresenta a lista das despesas pagas</td> </tr> <tr> <td>2</td> <td>o utilizador escolhe a fatura que quer ver</td> <td>o sistema apresenta detalhes da fatura</td> </tr> <tr> <td>3</td> <td></td> <td></td> </tr> </tbody> </table>		Actor Input	System Response	1		Apresenta a lista das despesas pagas	2	o utilizador escolhe a fatura que quer ver	o sistema apresenta detalhes da fatura	3		
	Actor Input	System Response											
1		Apresenta a lista das despesas pagas											
2	o utilizador escolhe a fatura que quer ver	o sistema apresenta detalhes da fatura											
3													
Exceção (passo1) [Não existem faturas pagas]	<table border="1"> <thead> <tr> <th></th> <th>Actor Input</th> <th>System Response</th> </tr> </thead> <tbody> <tr> <td>1</td> <td></td> <td>o sistema informa que não existem faturas pagas</td> </tr> <tr> <td>2</td> <td></td> <td></td> </tr> </tbody> </table>		Actor Input	System Response	1		o sistema informa que não existem faturas pagas	2					
	Actor Input	System Response											
1		o sistema informa que não existem faturas pagas											
2													

Figura 3.27: Especificação do Use Case: Consultar Despesas Pagas

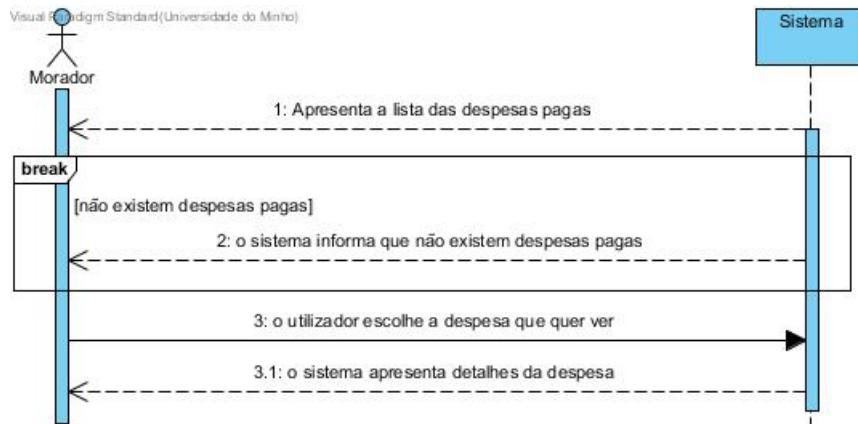


Figura 3.28: Diagrama de Sequência: Consultar Despesas pagas

Especificação: Pagar Despesas

É nesta especificação que está descrito como é que todos os pagamentos da nossa aplicação são efetivamente pagos. Cada utilizador, previamente registado e com login feito selecciona a conta que pretende pagar e procede ao pagamento em questão. O sistema após validar este pagamento atualiza a informação e coloca-a visível.

Flow of Events	Actor Input		System Response	
1			o sistema apresenta as conta para pagar	
2	o utilizador escolhe uma conta			
3			o sistema apresenta o valor da despesa a pagar	
4	o utilizador paga o valor			
5			o sistema valida o pagamento	
6			o sistema informa que registou a informação	

Comp.Alternativo(passo4)[utilizador paga a menos]	Actor Input		System Response	
1	o utilizador paga menos que o valor pretendido			
2			o sistema insere o utilizador na lista de dívidas e o valor que está a dever e desconta o valor pago na fatura	
3			regressa ao passo 5	

Excecao(passo 5)[Dinheiro Insuficiente na conta corrente]	Actor Input		System Response	
1			o sistema informa que não tem dinheiro suficiente na conta corrente para pagar o que pretende	

Comp.Alternativo(passo4)[utilizador paga a mais]	Actor Input		System Response	
1	o utilizador paga a mais do que o valor pretendido			
2			o sistema insere o utilizador na lista de empréstimos e o valor que ele emprestou	
3			regressa ao passo 5	

Excecao(passo 5)[Pagamento excede o valor da despesa]	Actor Input		System Response	
1			o sistema informa que o valor introduzido é maior que o valor da despesa	

Figura 3.29: Especificação do Use Case: Pagar Despesas

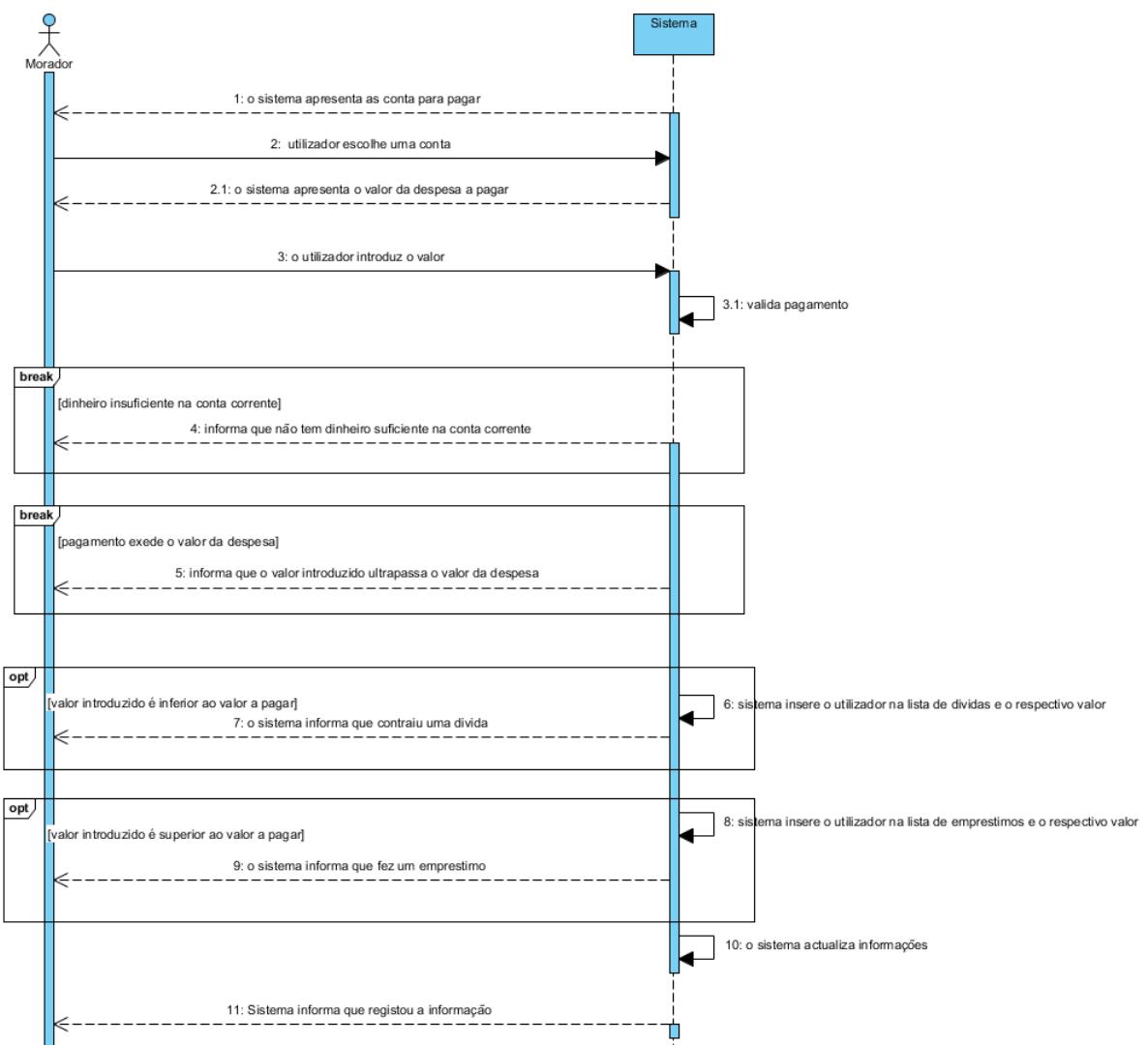


Figura 3.30: Diagrama de Sequência: Utilizador paga despesa

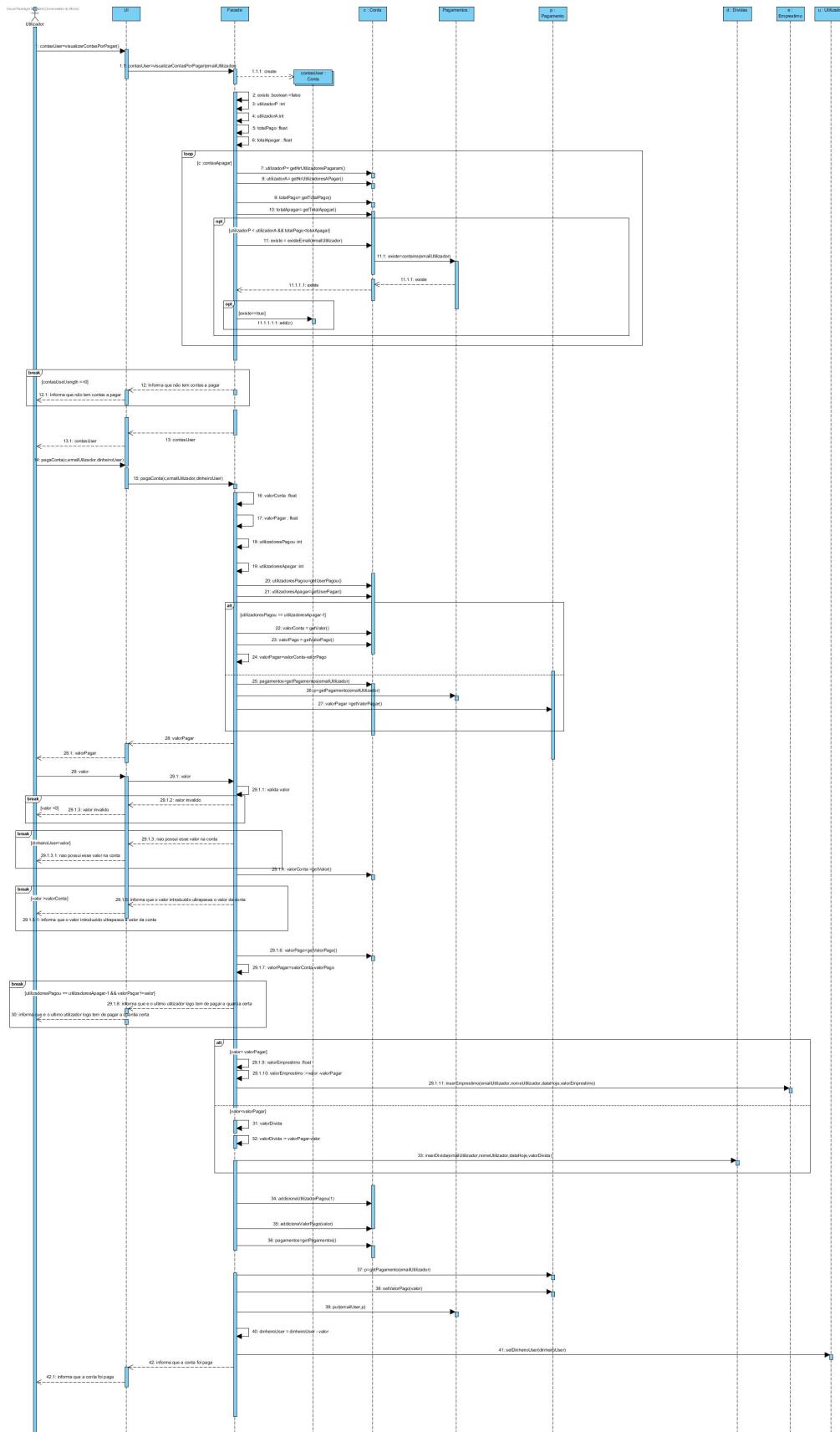


Figura 3.31: Diagrama de Iteração: Utilizador Paga Despesas

Especificação: Consultar Empréstimos

Nesta especificação o utilizador tem a vantagem de aceder à lista de empréstimos, ou seja, consegue através da aplicação visualizar os valores que emprestou à casa para efetuar o pagamento de conta/despesa de outros moradores do apartamento previamente registados na aplicação, embora o morador que emprestou não tenha conhecimento a quem emprestou.

Date	8/nov/2016 0:12:52									
Brief Description										
Preconditions										
Post-conditions										
Flow of Events	<table border="1"> <thead> <tr> <th></th> <th>Actor Input</th> <th>System Response</th> </tr> </thead> <tbody> <tr> <td>1</td> <td></td> <td>o sistema apresenta a lista de empréstimos</td> </tr> <tr> <td>2</td> <td>o utilizador escolhe um e vê o dinheiro que emprestou</td> <td></td> </tr> </tbody> </table>		Actor Input	System Response	1		o sistema apresenta a lista de empréstimos	2	o utilizador escolhe um e vê o dinheiro que emprestou	
	Actor Input	System Response								
1		o sistema apresenta a lista de empréstimos								
2	o utilizador escolhe um e vê o dinheiro que emprestou									
Exceção (passo 1) [não fez empréstimos]	<table border="1"> <thead> <tr> <th></th> <th>Actor Input</th> <th>System Response</th> </tr> </thead> <tbody> <tr> <td>1</td> <td></td> <td>o sistema informa que o utilizador não fez empréstimos</td> </tr> </tbody> </table>		Actor Input	System Response	1		o sistema informa que o utilizador não fez empréstimos			
	Actor Input	System Response								
1		o sistema informa que o utilizador não fez empréstimos								

Figura 3.32: Especificação do Use Case: Consultar Empréstimos

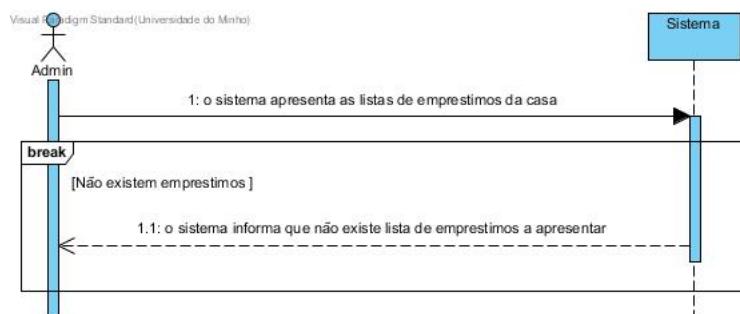


Figura 3.33: Diagrama de Sequência: Consulta lista de empréstimos

Especificação: Consultar Lista de Dívidas

Paralelamente ao que foi referido na especificação do use case de “consultar lista de empréstimos”, nesta secção é possível ter acesso à lista de dívidas. Por dívidas entende-se dinheiro que o morador deve à casa, neste caso a alguém que colocou o dinheiro por ele. A consulta de dívidas permite que cada utilizador tenha acesso à lista de dívidas e valores por pagar, desta forma tem sempre as contas com os valores exatos em dívida sem ter que fazer cálculos e arredondamentos.

Preconditions	o utilizador tem de ter efectuado o login	
Post-conditions		
Flow of Events	Actor Input	System Response
	1	o sistema apresenta a lista de dívidas
	2 o utilizador escolhe uma para pagar	
	3 o utilizador efectua o pagamento	
	4	o sistema valida o pagamento
Exceção (passo 4) [Dinheiro introduzido insuficiente]	Actor Input	System Response
	1	o sistema informa que o utilizador não tem dinheiro suficiente para pagar a dívida
Exceção (passo 4) [Dinheiro introduzido a menos]	Actor Input	System Response
	1	o sistema informa que o dinheiro introduzido não chega para pagar a dívida
Exceção (passo 4) [Dinheiro introduzido a mais]	Actor Input	System Response
	1	o sistema informa que o utilizador introduziu dinheiro a mais
Exceção (passo 1) [o utilizador não tem dívidas para pagar]	Actor Input	System Response
	1	o sistema informa que o utilizador não tem dívidas para pagar

Figura 3.34: Especificação do Use Case: Consultar lista de dívidas

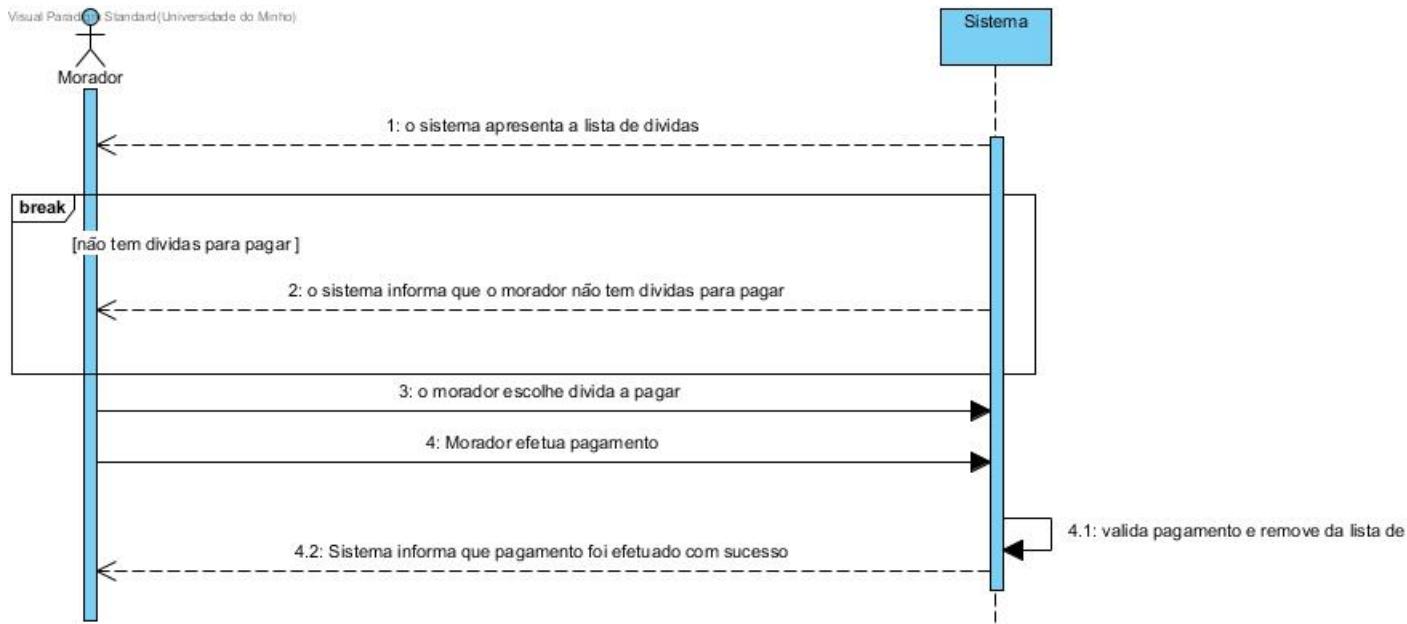


Figura 3.35: Diagrama de Sequência: Consulta lista de dívidas

3.3.4 Subdiagrama Interação com os Utilizadores

A interação com os utilizadores da aplicação torna a experiência mais interessante e motivadora.

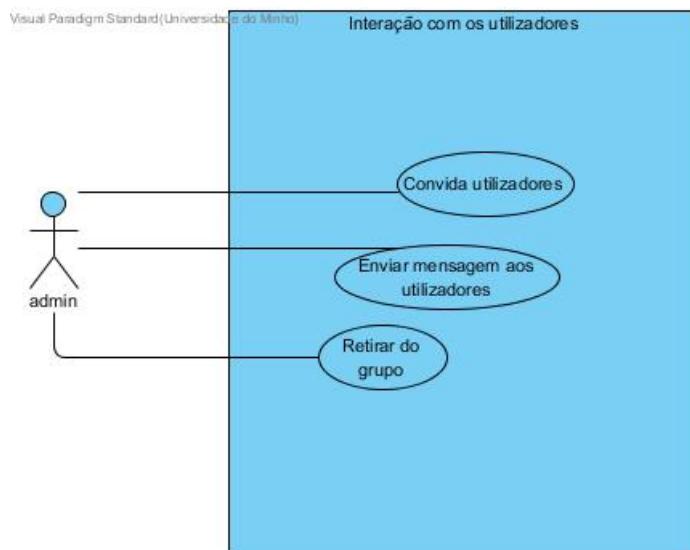


Figura 3.36: Subdiagrama: Interação com os utilizadores

Especificação: Convida Utilizadores

É através do email que um utilizador previamente registado convida outros a pertencerem ao “grupo” e a começarem a participar nas despesas do apartamento. O novo membro é notificado e de imediato pode proceder ao seu registo e efetuar login.

Date	17/Oct/2016 16:44:57															
Brief Description																
Preconditions	login efetuado															
Post-conditions	convite efetuado															
Flow of Events	<table border="1"> <thead> <tr> <th></th> <th>Actor Input</th> <th>System Response</th> </tr> </thead> <tbody> <tr> <td>1</td> <td></td> <td>pede para inserir e-mail</td> </tr> <tr> <td>2</td> <td>Insera e-mail</td> <td></td> </tr> <tr> <td>3</td> <td></td> <td>envia e-mail para os utilizadores</td> </tr> <tr> <td>4</td> <td></td> <td></td> </tr> </tbody> </table>		Actor Input	System Response	1		pede para inserir e-mail	2	Insera e-mail		3		envia e-mail para os utilizadores	4		
	Actor Input	System Response														
1		pede para inserir e-mail														
2	Insera e-mail															
3		envia e-mail para os utilizadores														
4																
Exceção 1 (passo 3) [e-mail inválido]	<table border="1"> <thead> <tr> <th></th> <th>Actor Input</th> <th>System Response</th> </tr> </thead> <tbody> <tr> <td>1</td> <td></td> <td>informa que mail não é válido</td> </tr> </tbody> </table>		Actor Input	System Response	1		informa que mail não é válido									
	Actor Input	System Response														
1		informa que mail não é válido														

Figura 3.37: Especificação do Use Case: Convida Utilizadores

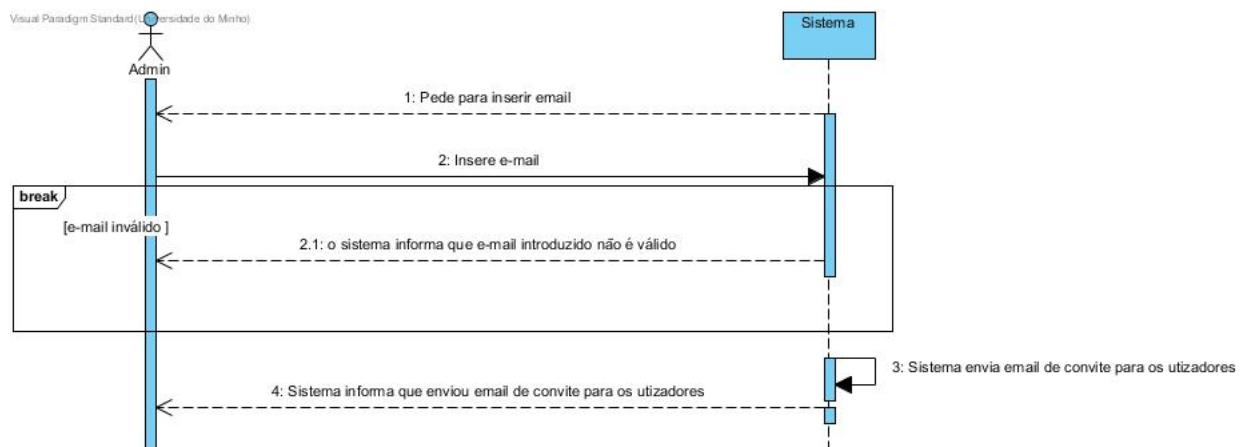


Figura 3.38: Diagrama de Sequência: Convida Utilizadores

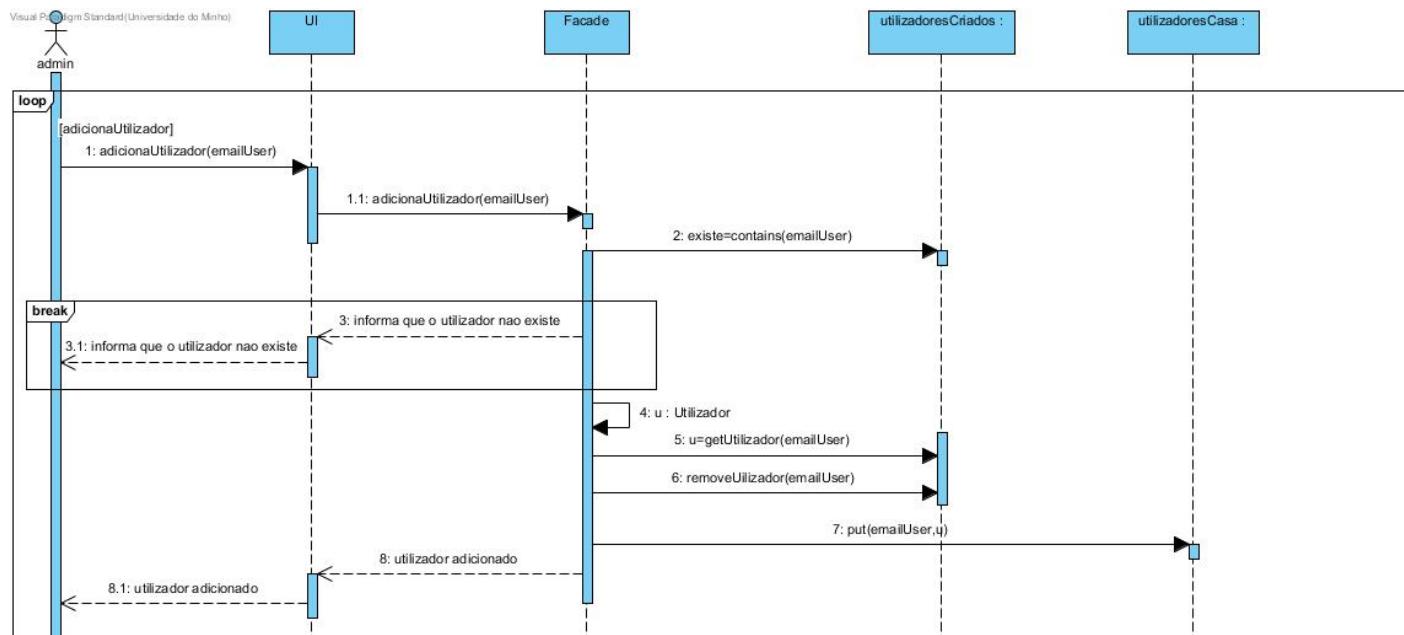


Figura 3.39: Diagrama de Iteração: Convida Utilizadores

Especificação: Envia Mensagens aos Utilizadores

O administrador da aplicação tem acesso à lista de utilizadores que estão aptos a receber mensagens. O mesmo escolhe a quem pretende enviar uma mensagem e envia. O sistema apenas confirma o envio. Este processo pode ser enviado aos utilizadores que o administrador quiser.

Date	1/nov/2016 23:28:15	
Brief Description		
Preconditions	Ter o login efectuado	
Post-conditions		
	Actor Input	System Response
Flow of Events	1	O sistema apresenta os utilizadores aos quais pode enviar mensagem
	2 O admin escolhe um utilizador	
	3 O admin escreve a mensagem	
	4	Mensagem é enviada

Figura 3.40: Especificação do Use Case: Envia Mensagens aos Utilizadores

Especificação: Retirar do grupo

O administrador consegue remover da aplicação os membros que de uma forma ou de outra já não pertencem ao grupo. O sistema apresenta a lista dos moradores registados e apenas é seleccionado o membro a excluir.

Date	23/out/2016 19:45:13	
Brief Description		
Preconditions	o admin ter o login efectuado	
Post-conditions	o admin consegue remover o utilizador	
	Actor Input	System Response
Flow of Events	1	O sistema apresenta a lista de utilizadores que estão a viver no apartamento
	2 O admin escolhe qual o utilizador que quer remover do grupo	
	3	o sistema consegue remover o utilizador
Excecao [Não consegue remover o utilizador] (passo 3)	Actor Input	System Response
	1	o sistema informa que o utilizador não pode ser removido

Figura 3.41: Especificação do Use Case: Retirar do grupo

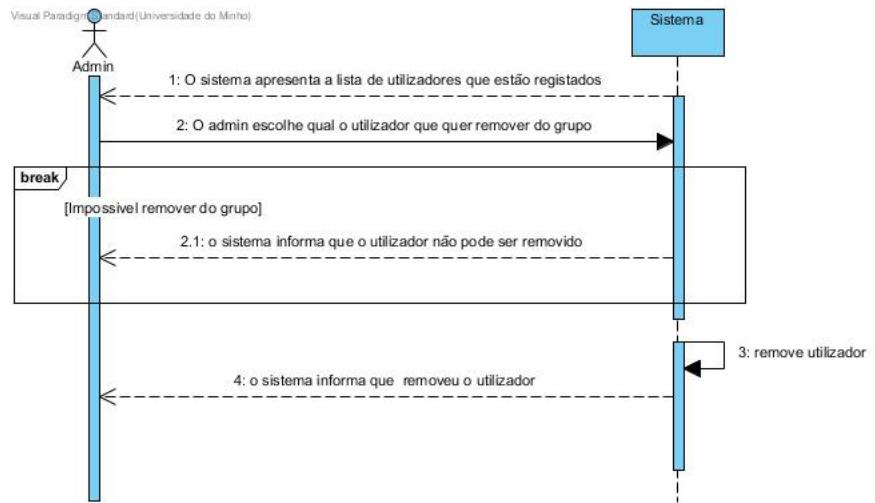


Figura 3.42: Diagrama de Sequência: Administrador Remove do Grupo

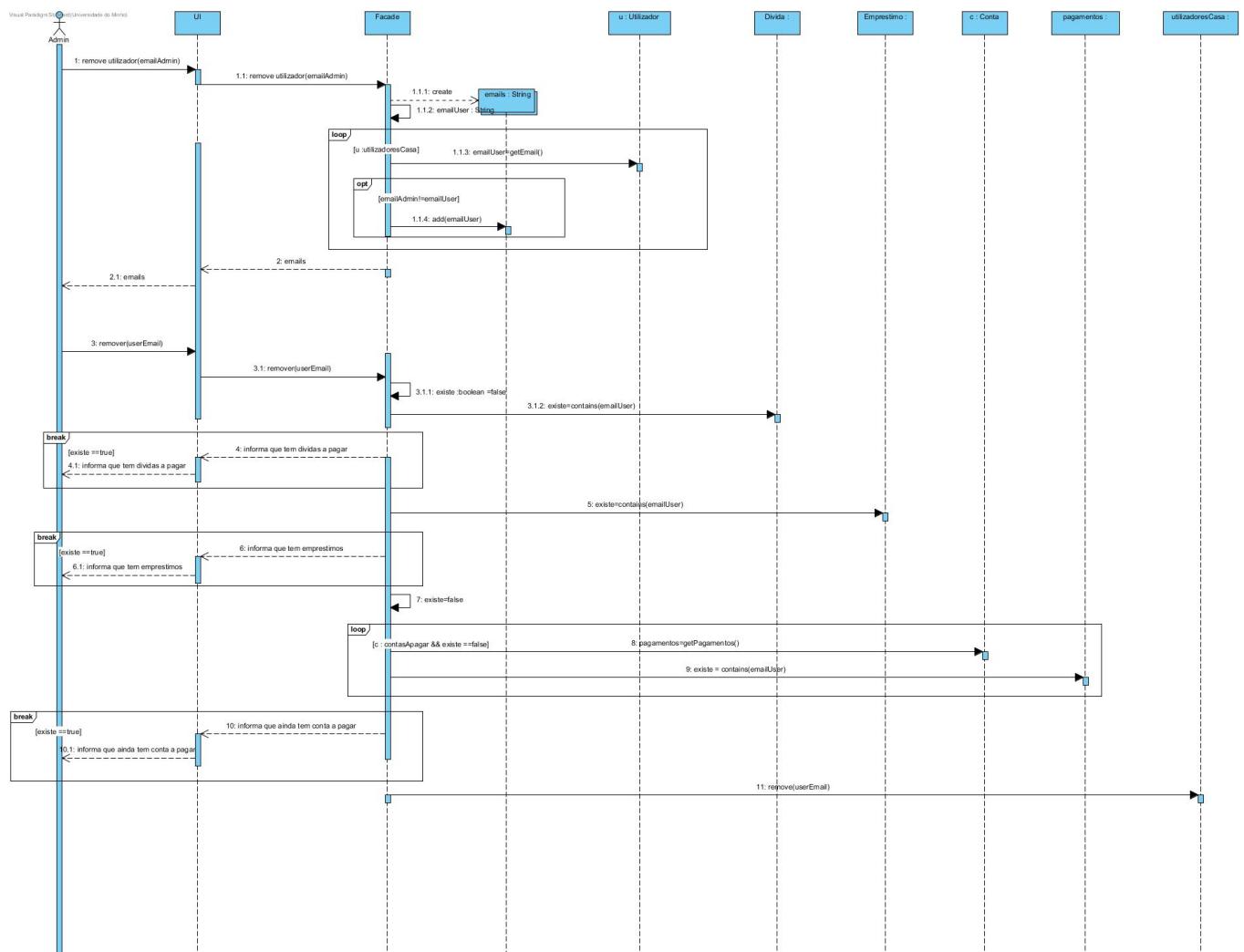


Figura 3.43: Diagrama de Iteração: Administrador Remove do Grupo

4. Implementação e Instalação do Sistema

4.1 Diagrama de Instalação

Os dois grandes subsistemas da aplicação desenvolvida são o computador do utilizador e o servidor da base de dados. O programa é uma aplicação Java, e para a apresentação gráfica ao utilizador, é usado JavaSwing. Para ser possível a comunicação entre a aplicação em Java e a base de dados *MySQL*, recorre-se à interface *JDBC*¹.

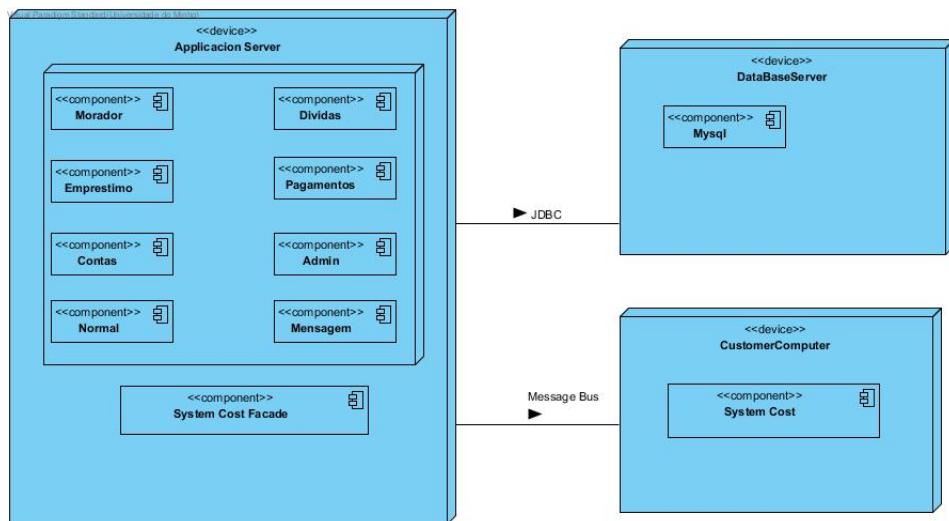


Figura 4.1: Diagrama de Instalação ou Deployment Diagram

¹Java Database Connectivity

4.2 Diagrama de Package

Os três principais pacotes definidos são *Presentation*, que trata da apresentação gráfica, em *Java Swing*; *Business*, que contém toda a lógica de negócio; e *Data*, responsável pela ligação à base de dados. A camada de apresentação comunica com a camada de lógica de negócio a partir da classe SGD (Sistema de Gestão de Despesas), que desempenha a função de *Facade*. Não existe qualquer tipo de comunicação feita diretamente entre a camada de apresentação e a camada de dados. A comunicação entre o package *Business* e a persistência de Dados é feita recorrendo às várias classes *DAO* implementadas no *Package Data*, que comunicam com a base de dados *MySQL*.

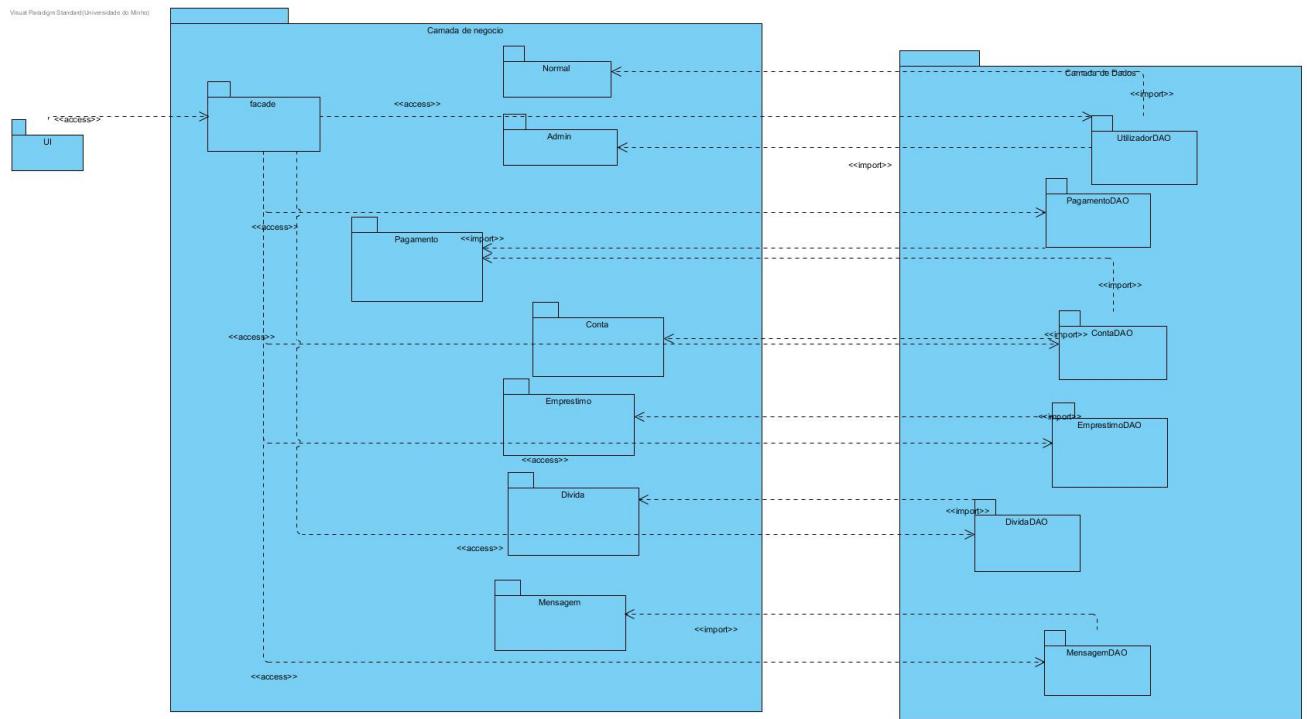


Figura 4.2: Diagrama de Pacotes

4.3 Diagrama de Actividade

4.3.1 Diagrama de Atividade: Funcionamento da Aplicação

O objetivo do diagrama de atividades é mostrar o fluxo de atividades num único processo. O diagrama mostra como uma atividade depende de outras e torna a leitura do projeto mais simples e apelativa. Temos de seguida o diagrama de atividade que explica o funcionamento da nossa aplicação. Conseguimos claramente identificar com detalhe as funcionalidades do administrador da aplicação, assim como, do morador e do sistema. Fica evidente a ligação que todas estas componentes têm entre si.

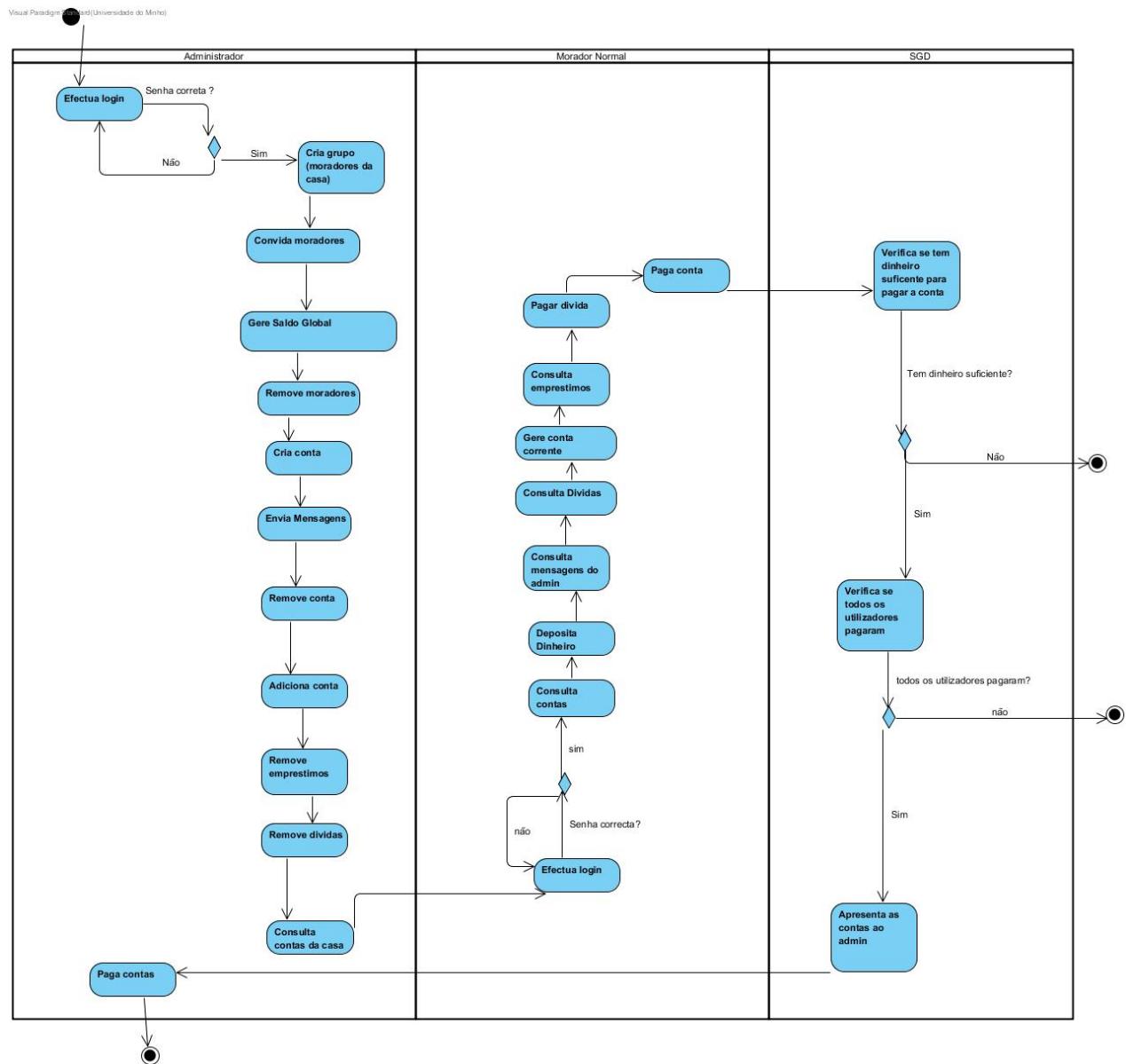


Figura 4.3: Diagrama de Atividade

4.3.2 Adicionar Despesa/Fatura

O diagrama que se segue diz respeito ao caso de uso “Inserir Despesa”. É precisamente neste diagrama que está explicado de que forma o administrador da aplicação introduz uma nova fatura no sistema. É da responsabilidade do administrador introduzir em sistema as novas faturas a pagar, definir que tipo de fatura é (recorrente ou extraordinária), e ainda, distribuir as percentagens que cada morador tem que pagar. Portanto, de forma bastante simples segue-se o diagrama que explicita cada um destes aspectos referidos.

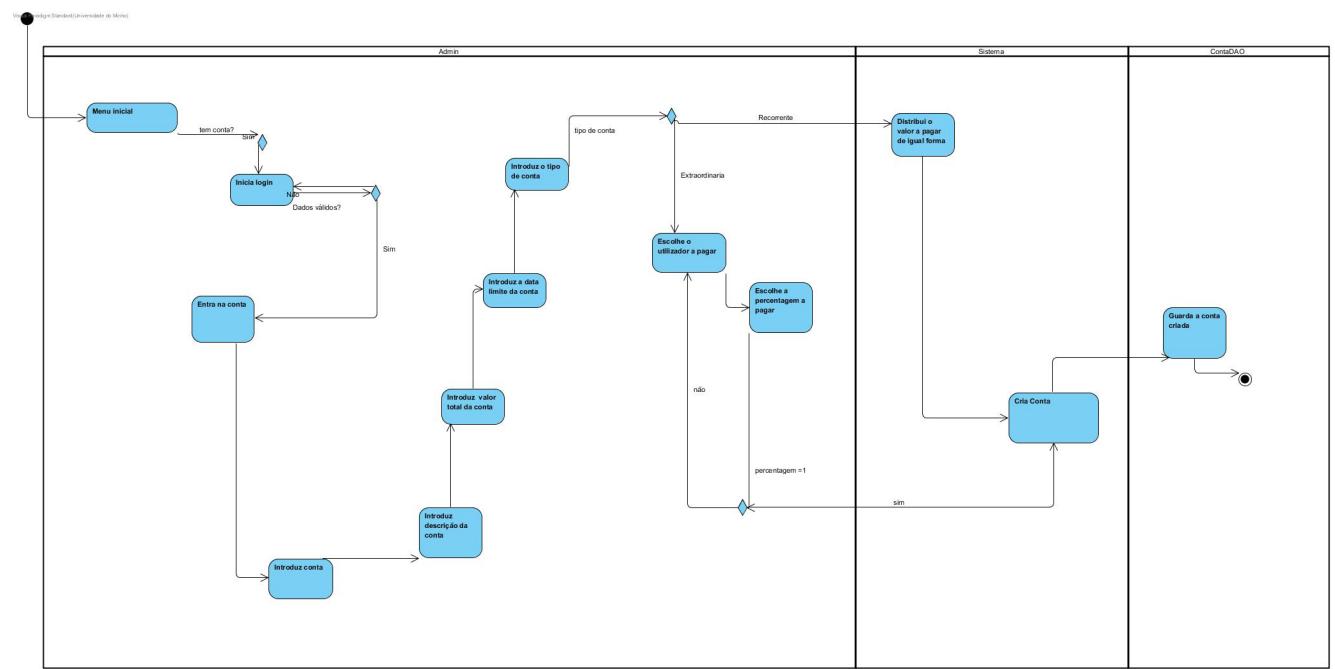


Figura 4.4: Diagrama Atividade: Criar Conta

4.3.3 Pagar Conta

Perante um pagamento de um user considerado normal, o procedimento é bastante simples e está descrito na imagem que se segue. O sistema apresenta ao utilizador as contas que têm que ser pagas e este, no imediato, selecciona uma conta para efetuar o devido pagamento. Todo este processo fica atualizado assim que há movimentação de valores.

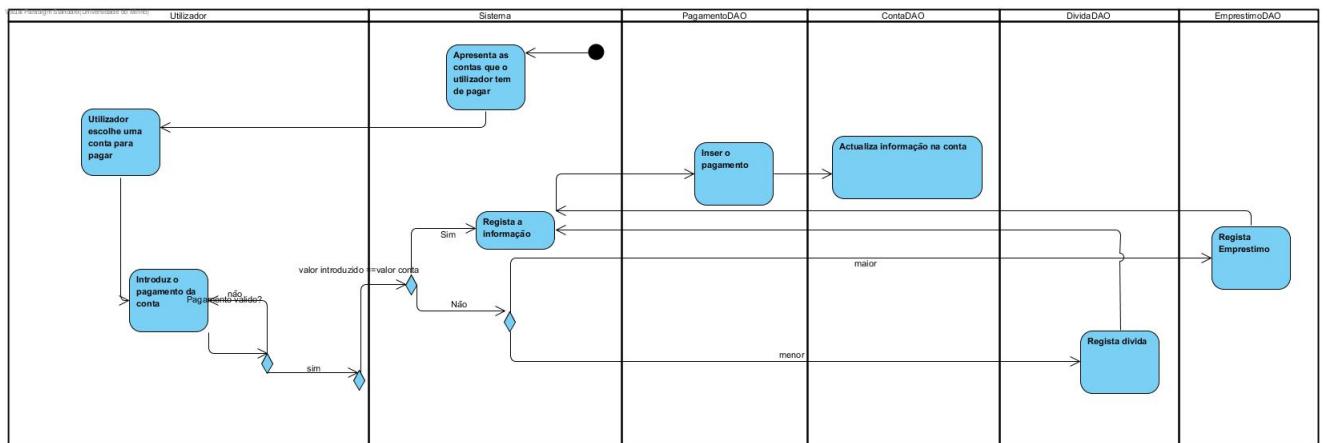


Figura 4.5: Diagrama de Atividade: Pagar Conta

4.3.4 Pagar Divida

Como já referimos anteriormente neste relatório, cada morador fica encarregue de pagar as suas dívidas. É o sistema que apresenta ao utilizador a lista de dívidas e este apenas selecciona a dívida a pagar. Esta leitura está bastante simples e coerente no diagrama que apresentamos de seguida.

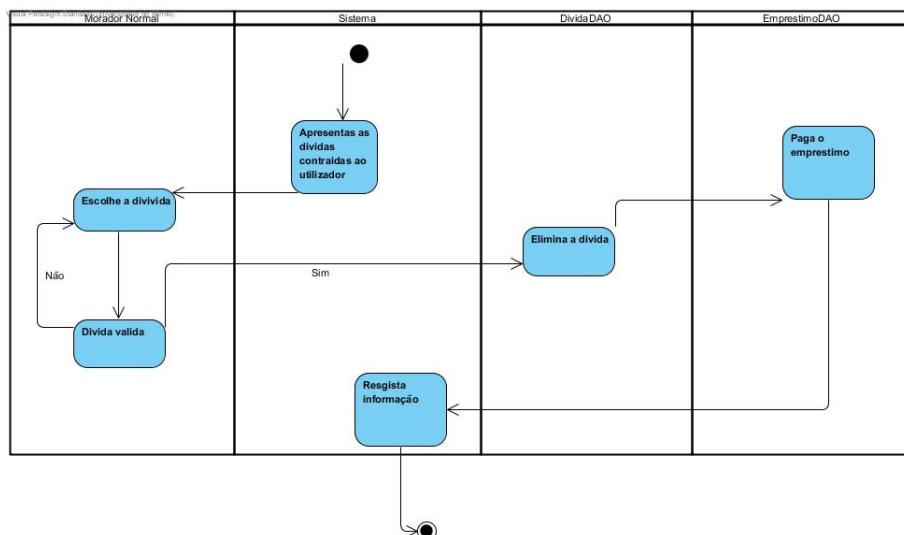


Figura 4.6: Diagrama de Atividade: Pagar divida

5. Interface do Sistema Gestão de Despesas - SGD

5.1 Máquinas de Estado

5.1.1 Efetuar Login

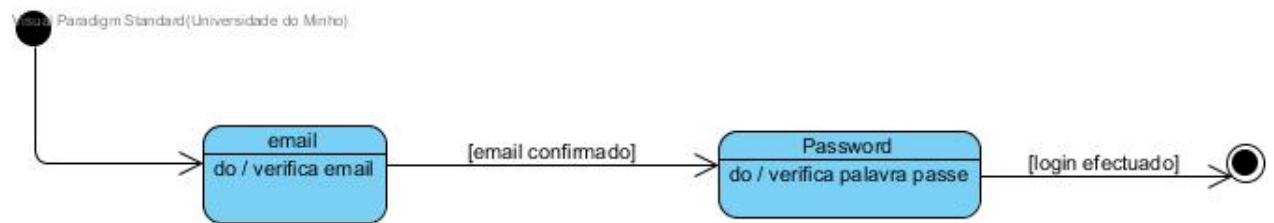


Figura 5.1: Máquina de Estados: Efetuar Login

5.1.2 Criar Conta

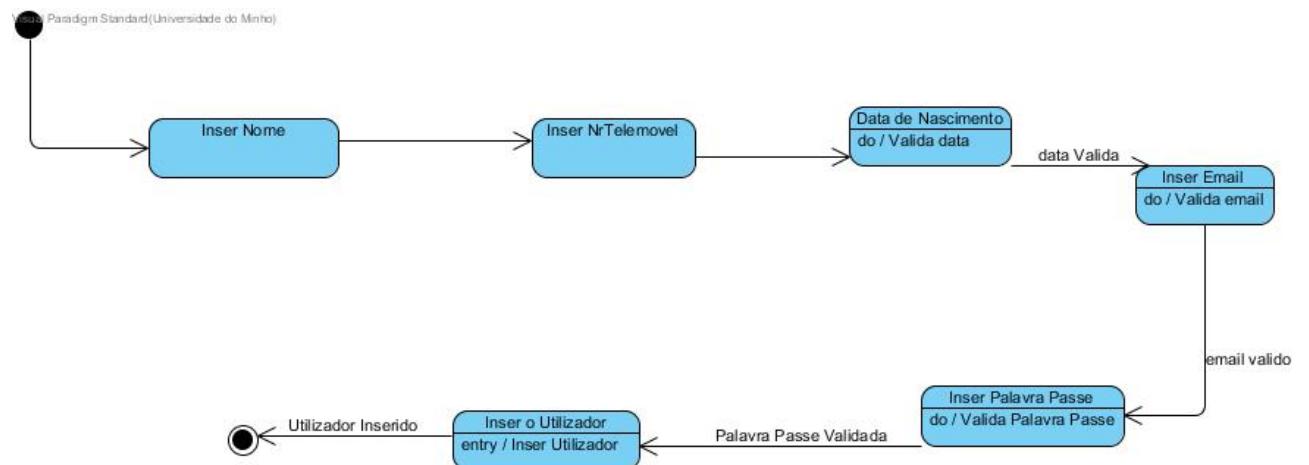


Figura 5.2: Máquina de Estados: Criar Conta

5.1.3 Convidar Utilizadores

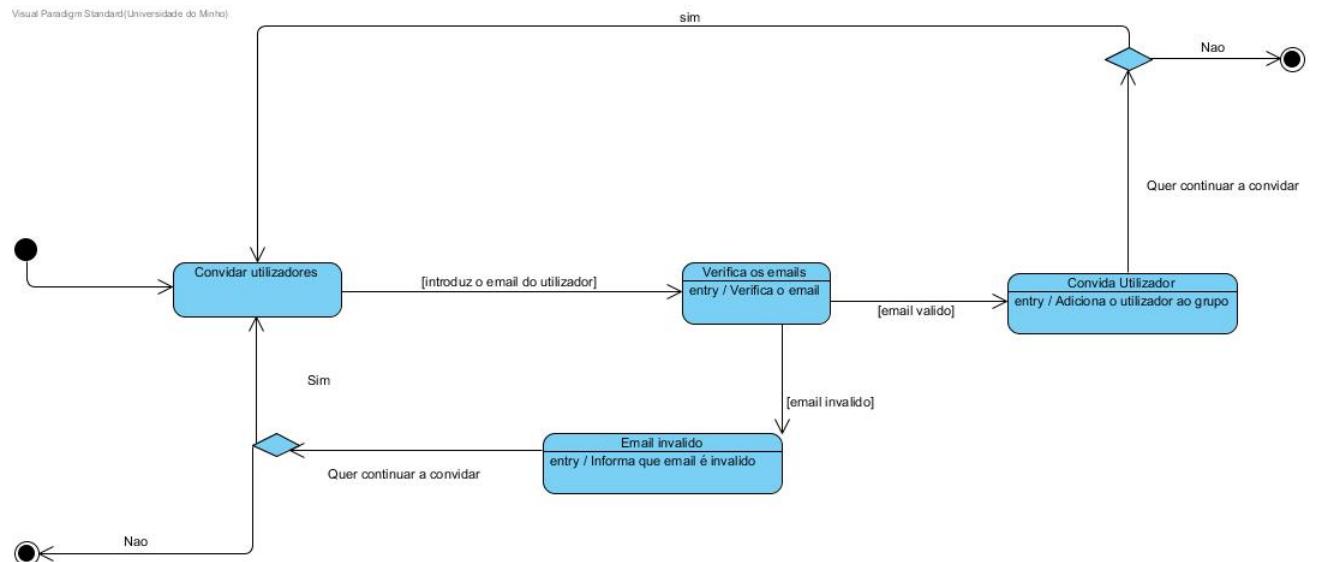


Figura 5.3: Máquina de Estados: Convidar Utilizadores

5.1.4 Remover Utilizadores

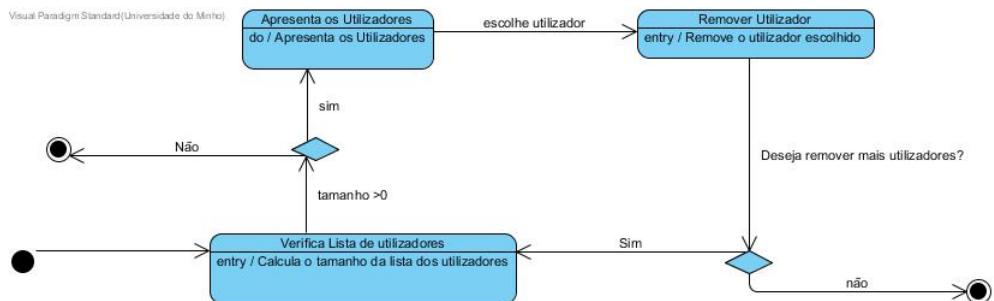


Figura 5.4: Máquina de Estados: Remover Utilizadores

5.1.5 Inserir Fatura

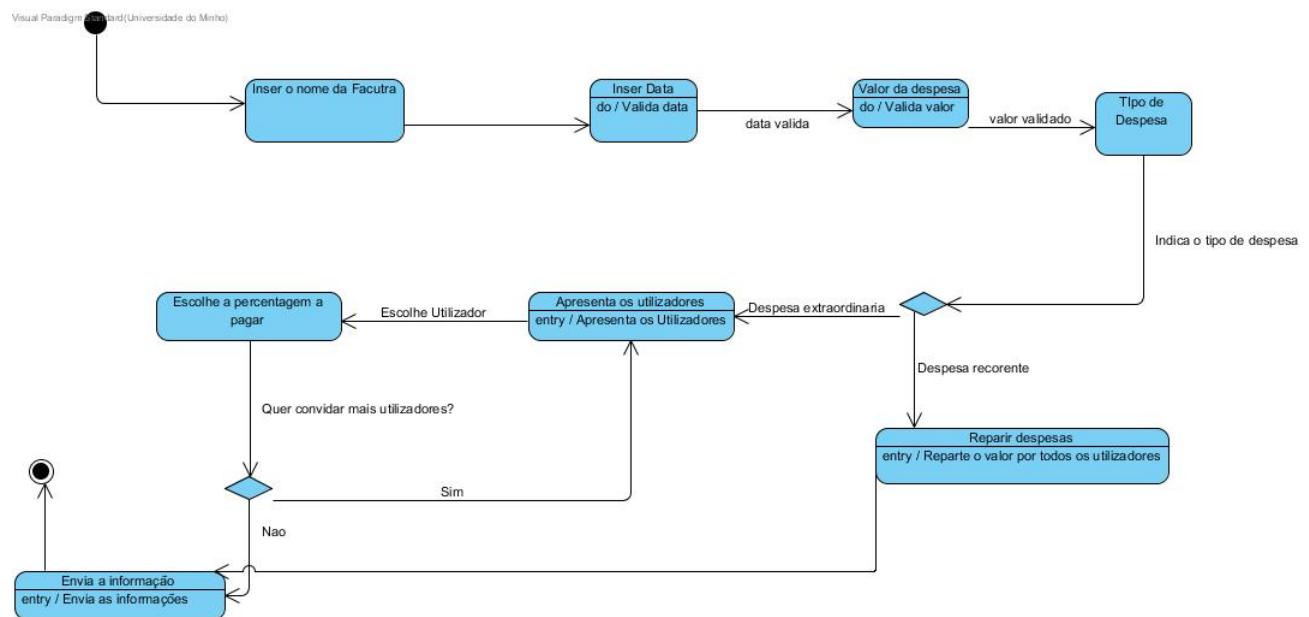


Figura 5.5: Máquina de Estados: Inserir Fatura

5.1.6 Deposita Dinheiro

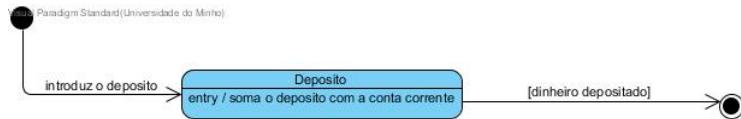


Figura 5.6: Máquina de Estados: Deposita Dinheiro

5.2 Mockups

Apresentamos de seguida uma proposta de interface com o utilizador. Utilizámos o programa 'Pencil' para nos auxiliar na construção de uma possível interface com o utilizador.

Como já refirmos, para o utilizador efetuar o login necessita de se registar previamente, fornecendo alguns dados que o identifiquem.

The window title is 'System Cost'. It contains the following fields:

- Introduza Nome Completo: Celia
- Introduza Data de Nascimento: 24/12/1992
- Introduza Email: celia@mail.com
- Introduza Password: *****
- Repetir Password: *****
- Introduza o número de telemóvel: 9312456

At the bottom are two buttons: 'Criar Conta' (Create Account) and 'Cancelar' (Cancel).

Figura 5.7: Criar nova Conta

Esta será a janela para os moradores e administrador efetuarem login na aplicação.

The window title is 'System Cost'. It contains the following fields:

- Email: celia@email.com
- Password: *****

At the bottom is a single 'Login' button.

Figura 5.8: Login

O Administrador efetua login, mas ainda não existem grupo criado. Pode escolher a opção "Convidar Pessoas", para iniciar a formação de um grupo.

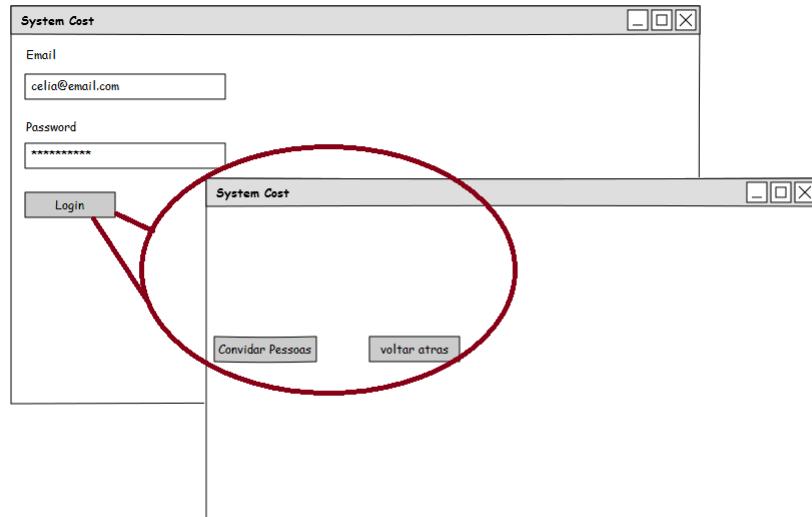


Figura 5.9: Login quando não existem grupos criados

Haverá sempre a possibilidade de ver/alterar os campos preenchidos inicialmente. Carregando no botão grupo abre uma janela onde se pode entrar para grupo constituído pelos elementos da casa/apartamento. Após clicar no botão "Entrar no grupo" é apresentada uma lista com as pessoas já existentes e a possibilidade de convidar mais membros.

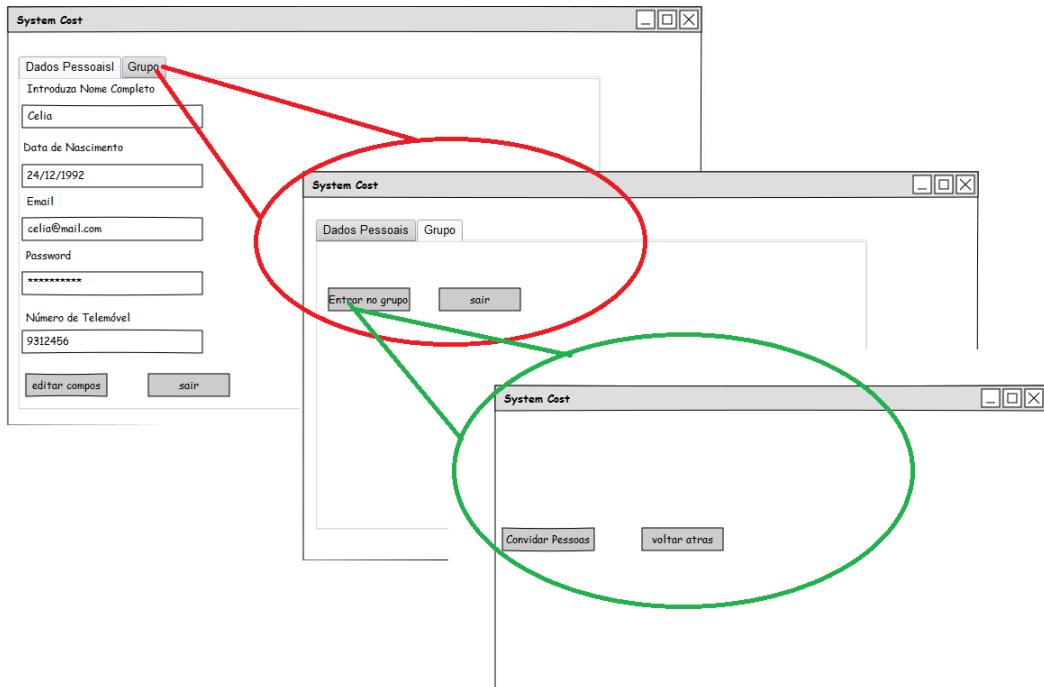


Figura 5.10: Visualização/Alteração dos dados

Após o utilizador efetuar o login é-lhe apresentada uma janela com as funcionalidades que a aplicação lhe oferece, como por exemplo pagar contas e acesso à lista de dívidas, assim como

o valor da conta corrente.

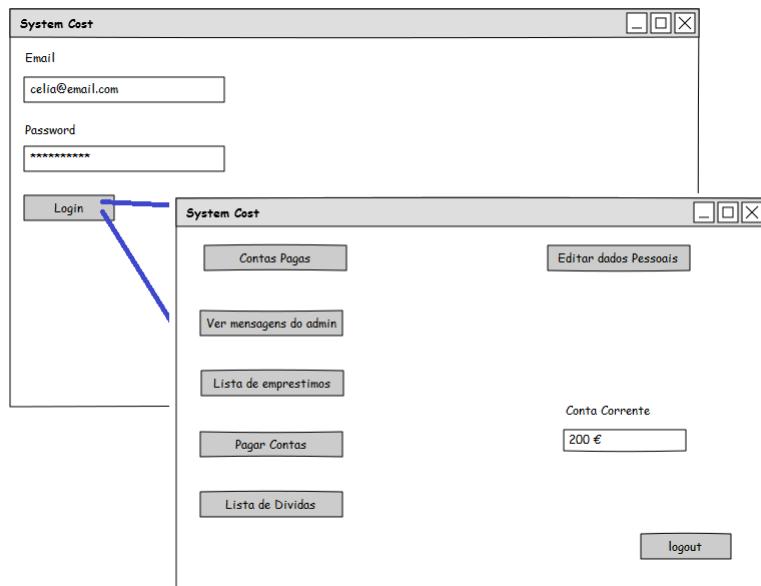


Figura 5.11: Login /página inicial morador

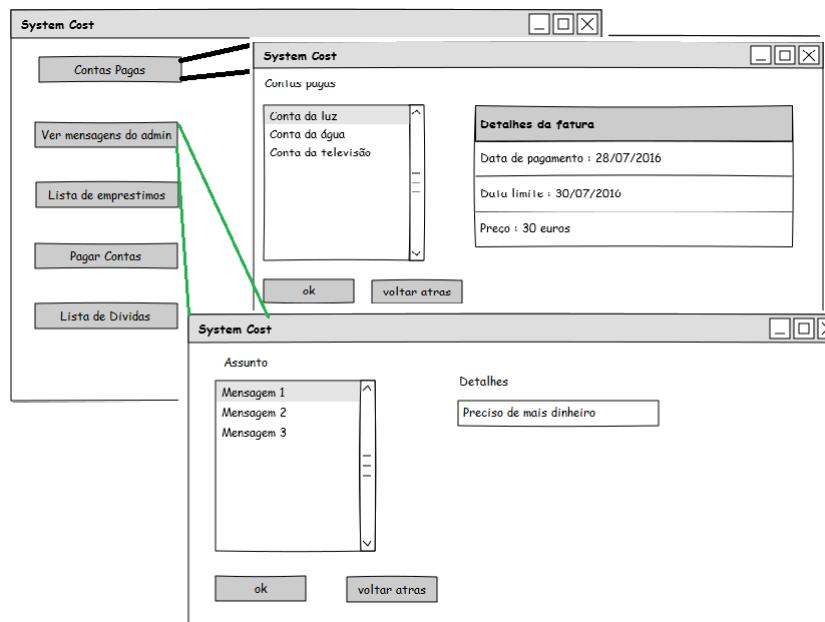


Figura 5.12: Opções do utilizador

Após o administrador efetuar o login é-lhe apresentada uma janela com as funcionalidades que a aplicação lhe oferece, como por exemplo pagar contas e adicionar/remover utilizador, enviar mensagem e verificar o saldo global

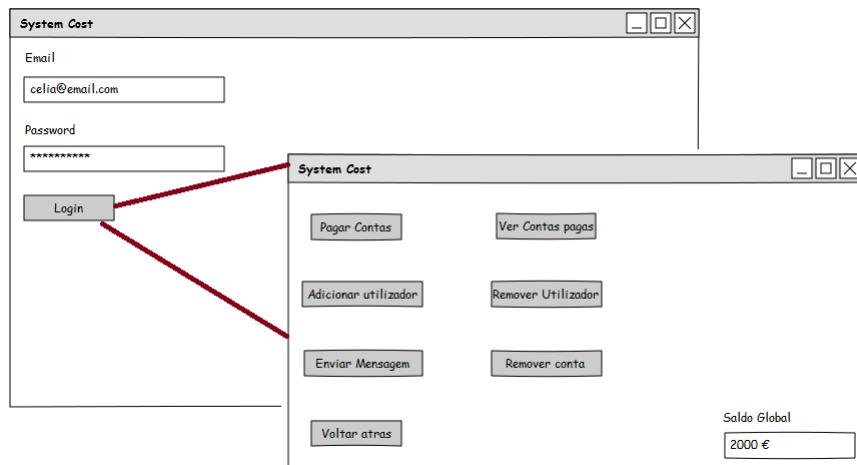


Figura 5.13: Login/Privilépios de administrador

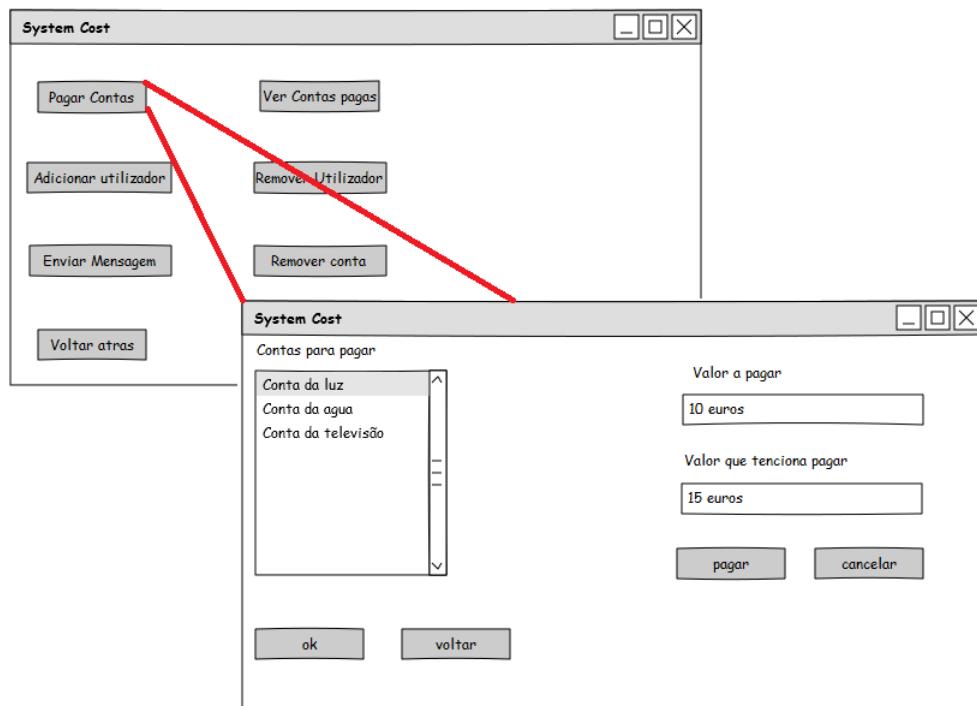


Figura 5.14: Interface do administrador, janela pagar contas

5.3 Interação com utilizador

Como implementação dos mockups, apresentamos de seguida cada uma das janelas por nós criadas de interação com o utilizador. Em primeiro lugar, apresentamos a janela inicial, onde o utilizador pode fazer login e/ou criar conta.



Figura 5.15: Página inicial do SGD

Acedendo ao botão “criar conta” surge a seguinte janela. Aqui o utilizador cria a sua conta, após especificar cada campo corretamente.

A screenshot of a registration form titled "Nome Completo" with the value "Márcia Maria Fernandes da Costa". Below it is a "Data de Nascimento" section with dropdown menus for "Dia" (21), "Mes" (1), and "Ano" (1994). To the right of the form is a decorative graphic consisting of two nodes connected by a line, with a red circle above the first node and a grey circle below the second. The next section is "Introduza o endereço do correio electrónico" with the value "marciacosta". Below that is "Introduza a palavra passe" with a masked input field showing "*****". To the right is a graphic of a grey square with three horizontal bars. The next section is "Repetir palavra passe" with a masked input field showing "*****". To the right is a graphic of a grey square with three horizontal bars. The final section is "Número de telemóvel" with the value "986532012". To the right is a graphic of a blue location pin. At the bottom left is a blue "criar conta" button, and at the bottom right is a blue "cancelar" button.

Figura 5.16: Janela de criação de conta

Quando o administrador da aplicação acede a “Administrar Casa”, surge de imediato a janela com as funcionalidades que apenas o administrador tem acesso, como por exemplo, “convidar utilizadores para a casa”, “ver contas por pagar” e “Remover do grupo”. Nesta janela existe ainda uma “caixa” onde consta o saldo global da casa. Este saldo, como já foi referido anteriormente, é o resultado do somatório de todas as contas correntes.

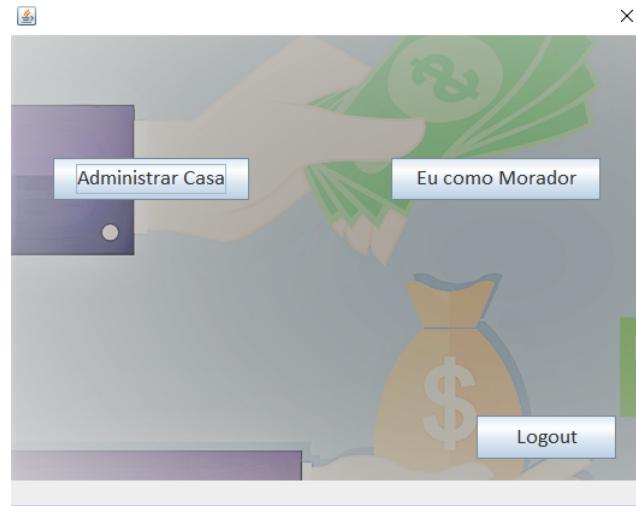


Figura 5.17: Página inicial do administrador

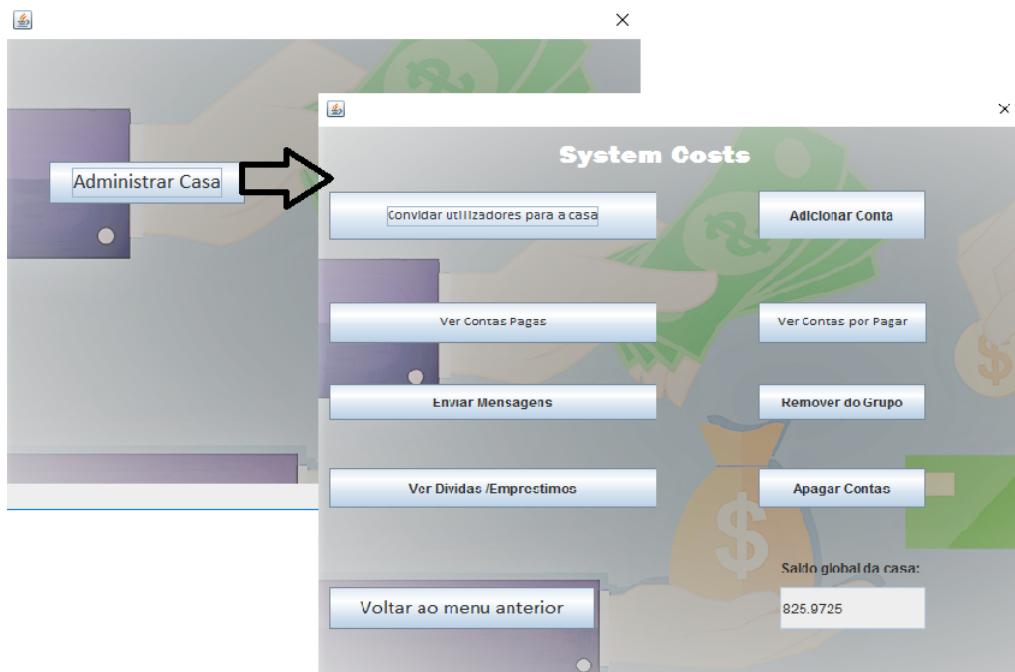


Figura 5.18: Página inicial do administrador com todas as funcionalidades

O administrador pode convidar utilizadores com conta previamente criada para “aceder ao grupo” do apartamento. A ideia é selecionar os membros com conta criada e adicioná-los ao mesmo grupo para partilhar despesas.

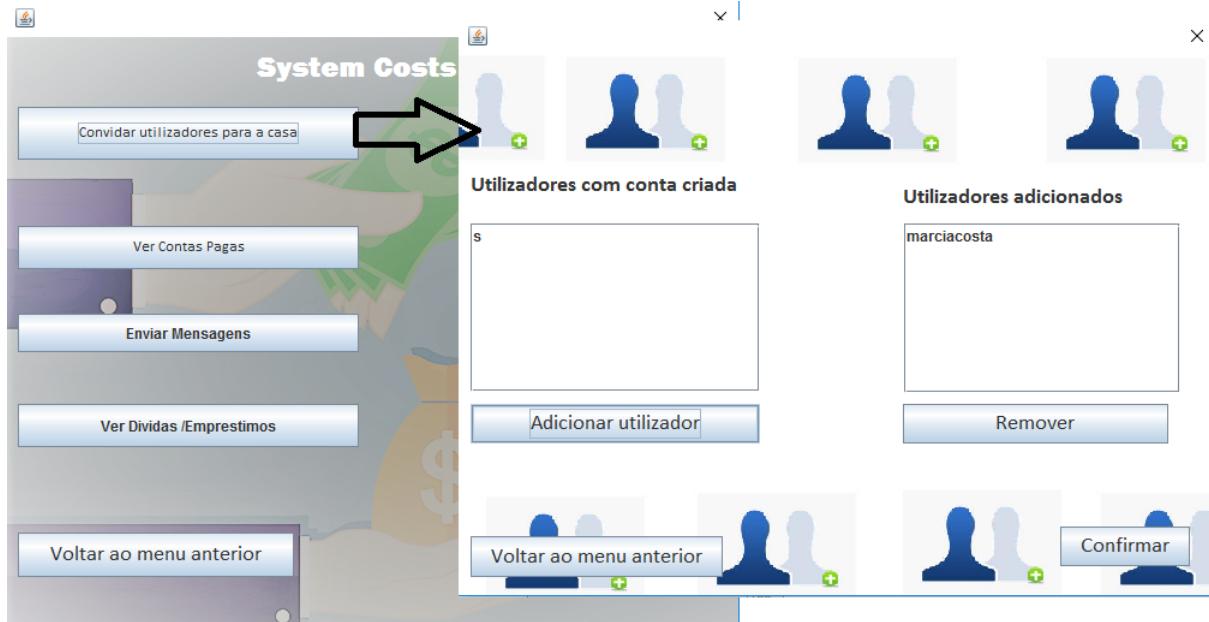


Figura 5.19: Janela para adicionar novos membros

O administrador tem acesso à lista de contas que já foram pagas anteriormente pelos restantes moradores do apartamento. Aqui consegue visualizar os dados de cada fatura assim como os detalhes de cada pagamento feito por cada morador.

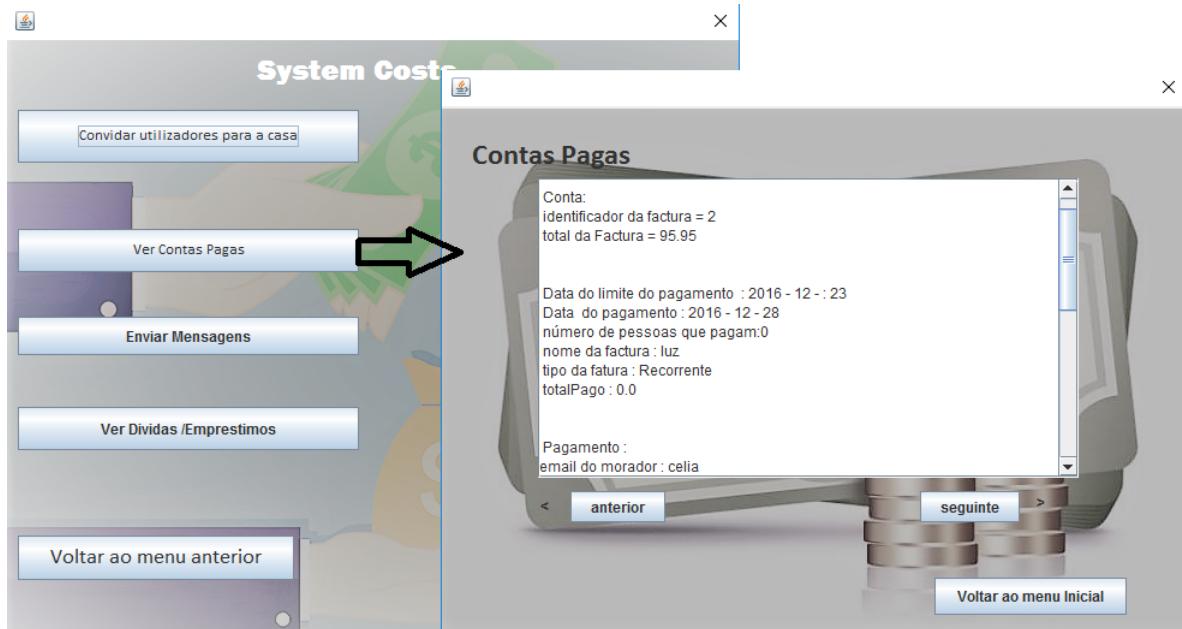


Figura 5.20: Janela para visualizar as contas pagas

O administrador tem a funcionalidade de enviar mensagens aos outros utilizadores. Desta forma torna a comunicação mais interativa e consegue fazer uma espécie de lembrete para que não se esqueça de efetuar os seus pagamentos, liquidar dívidas, entre diversos tipos de mensagens. Para tal, basta seleccionar os membros a quem pretende enviar determinada mensagem e depois envia. Está claramente descrito na imagem que se segue.

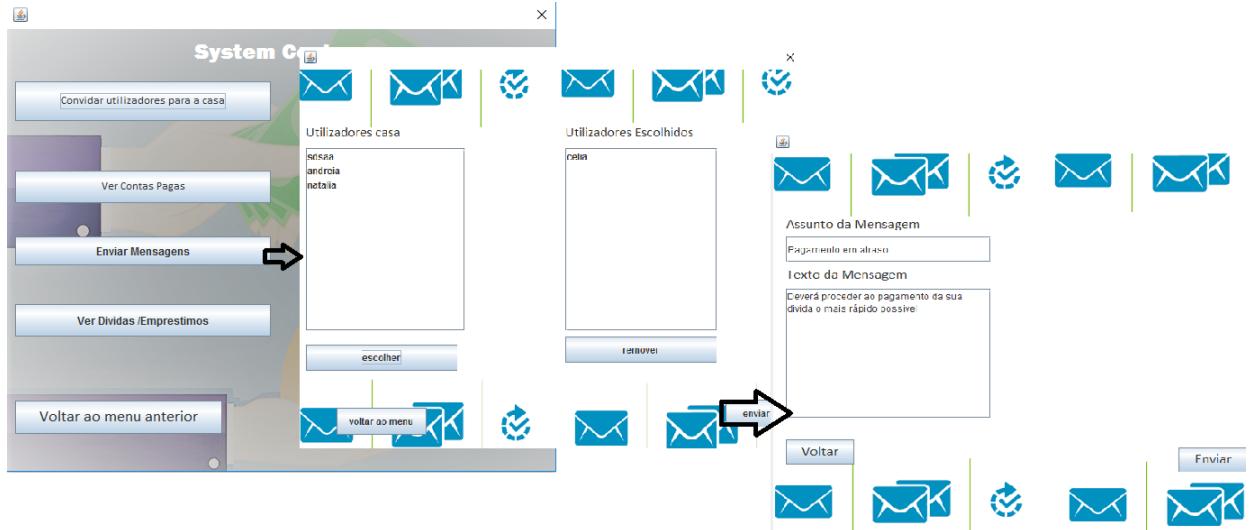


Figura 5.21: Janela para enviar mensagens aos moradores

A interface que se segue diz respeito às dívidas e empréstimos da casa. O administrador tem acesso a consultar todas as dívidas que eventuais moradores possam ter, assim como cada empréstimo feito à casa para pagar montantes em falta de outros inquilinos. Para além de consultar, o administrador da aplicação tem a capacidade para apagar determinada dívida e/ou empréstimo.

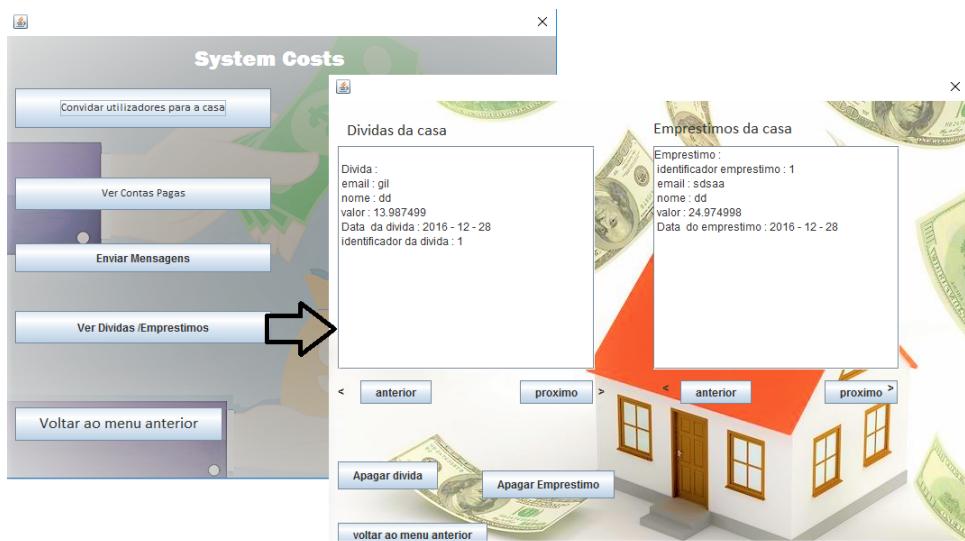


Figura 5.22: Janela para visualizar as dívidas e empréstimos

Ainda dentro das funcionalidades possíveis do administrador, existe o botão “Adicionar Conta”. É da responsabilidade do administrador da aplicação colocar em sistema todas as contas que recepciona. Para colocar uma nova fatura em sistema, visível para todos os moradores é necessário colocar o nome da conta, o valor da mesma e a data limite de pagamento. Assim, surge a seguinte janela.

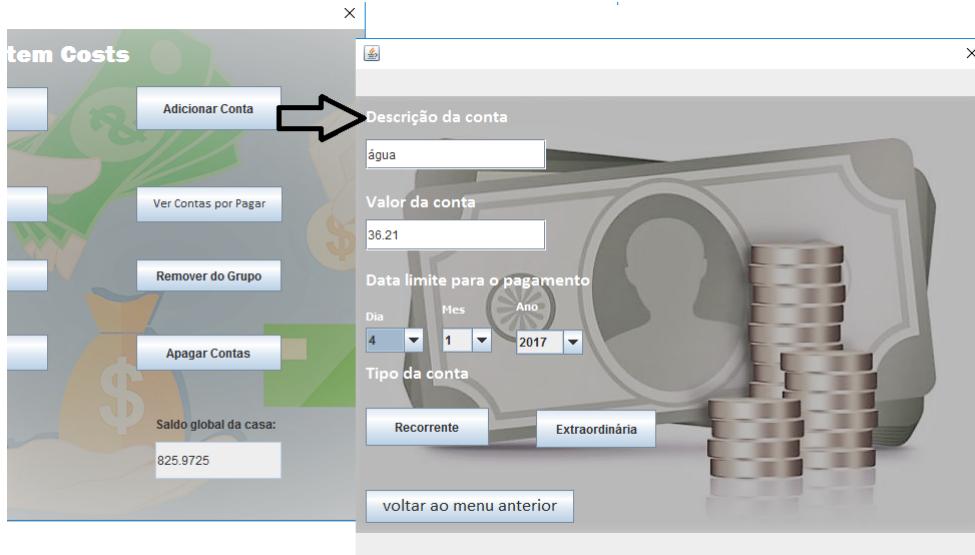


Figura 5.23: Janela para adicionar fatura/despesa

O administrador ao aceder a “Ver contas por pagar”, visualiza a lista de contas que estão por pagar e ainda consegue ter acesso aos detalhes de cada morador, isto é, consegue saber quem já pagou, quem falta pagar e que quantias tem cada um que pagar.

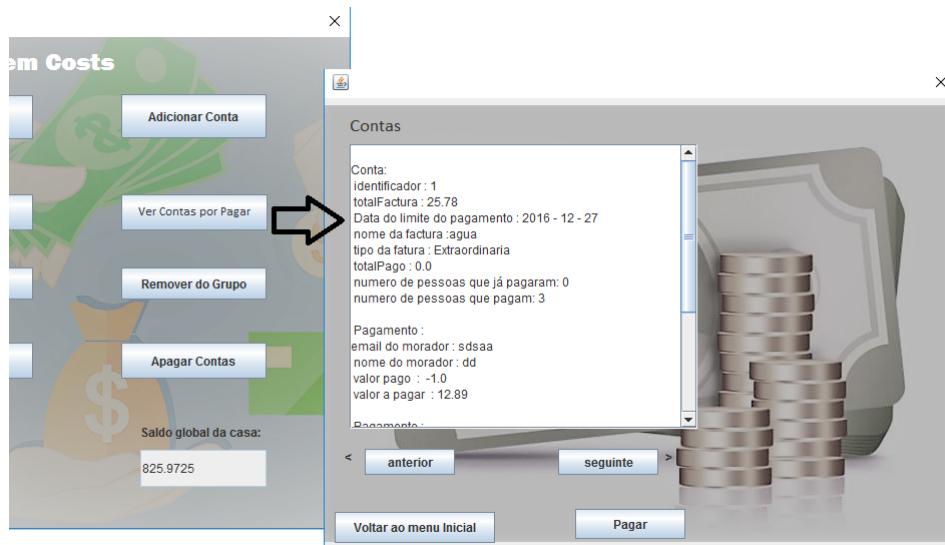


Figura 5.24: Janela para visualizar as contas por pagar

O administrador pode remover utilizadores com conta previamente criada para “sair do grupo” do apartamento. A ideia é selecionar os membros com conta criada e de seguida removê-los. Desta forma esse membro ou membros deixa de ter acesso a aplicação e deixa automaticamente de poder partilhar despesas.



Figura 5.25: Janela para remover utilizadores do sistema

É da competência do administrador apagar as contas/faturas do sistema. O administrador escolhe o identificador do Pagamento que efetivamente pretende eliminar e apaga. É bastante simples, pois , após visualizar a lista de contas, basta saber o id a eliminar.

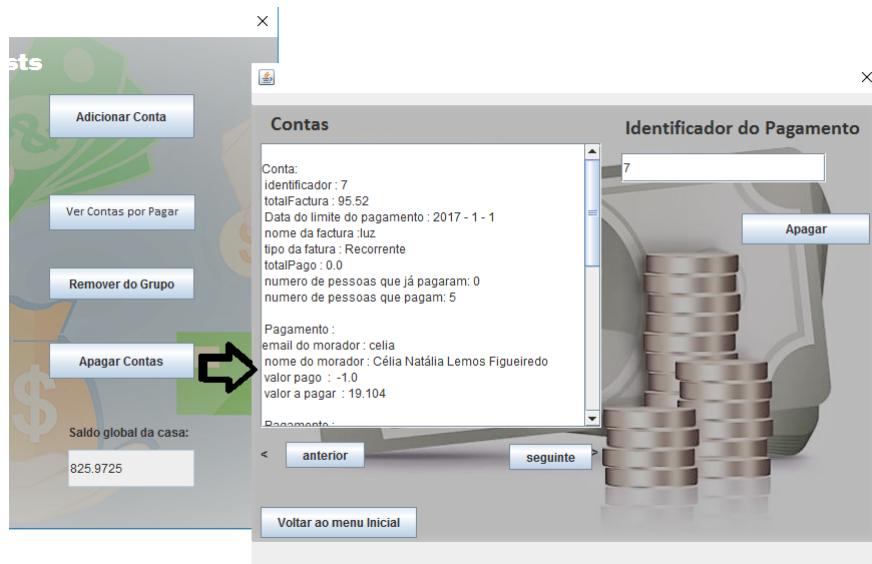


Figura 5.26: Janela para remover utilizadores do sistema

Cada utilizador, sendo um morador do apartamento após fazer login surge-lhe a janela com todas as suas funcionalidades. Aqui tem acesso a todas as suas funções, como por exemplo, “ver mensagens do admin”, “ver contas pagas”, “consultar dívidas”, entre outras. Ainda nesta janela consta no canto inferior esquerdo o valor da conta corrente do morador em questão. Estas funcionalidades são comuns tanto ao utilizador normal como ao administrador, quando este entra como “Eu como Morador”.



Figura 5.27: Página inicial do morador

Cada utilizador pode a qualquer momento receber uma mensagem de texto enviada do administrador. Selecione este botão surgem as mensagens recebidas. A imagem que se segue mostra exatamente uma mensagem que o administrador enviou a um morador.

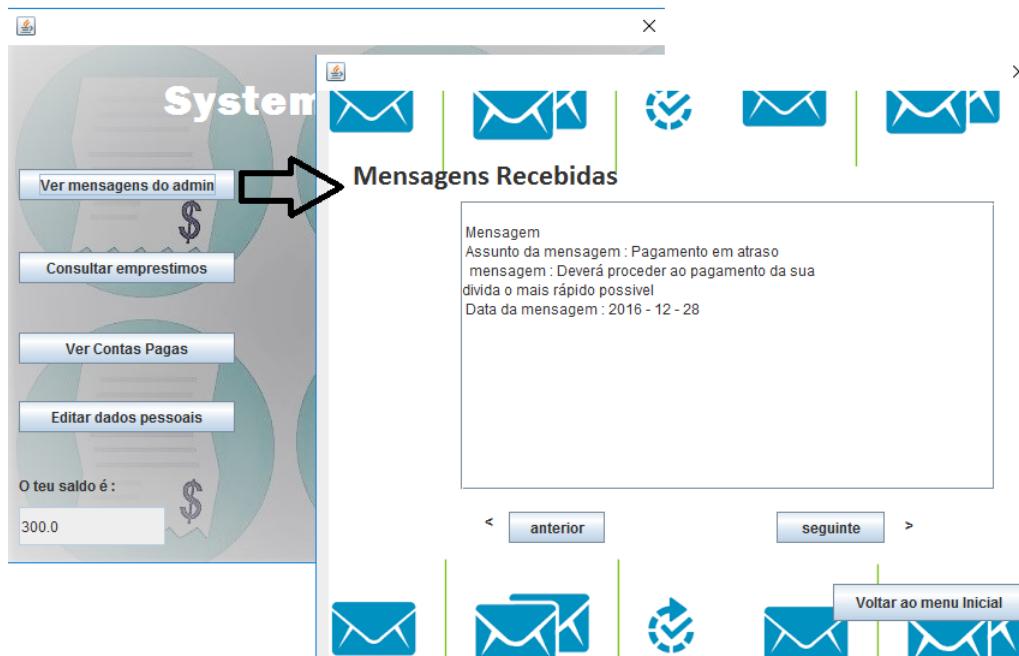


Figura 5.28: Janela ver mensagens do administrador

Nesta janela cada morador tem acesso à lista de valores que emprestou ao apartamento, ou seja, à lista de empréstimos que foi por ele efetuada. Cada empréstimo é identificado por um e-mail, nome, valor e uma data de empréstimo.

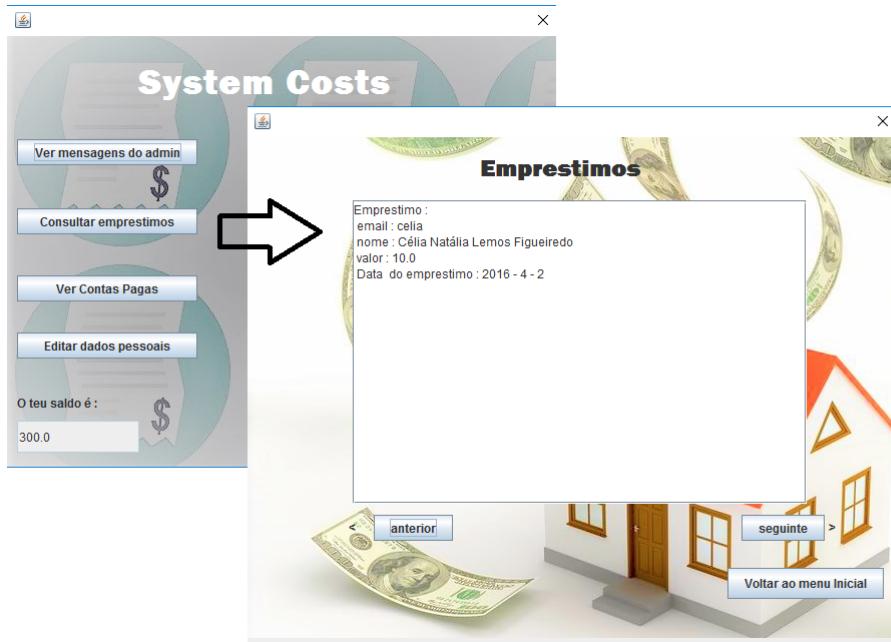


Figura 5.29: Janela para consultar os empréstimos que fez à casa

Nesta janela é apresentada a lista de contas por cada morador já pagas. Nesta janela surge detalhadamente o valor total da fatura, a percentagem que cada morador tem efetivamente que pagar e o valor que foi pago até ao momento.

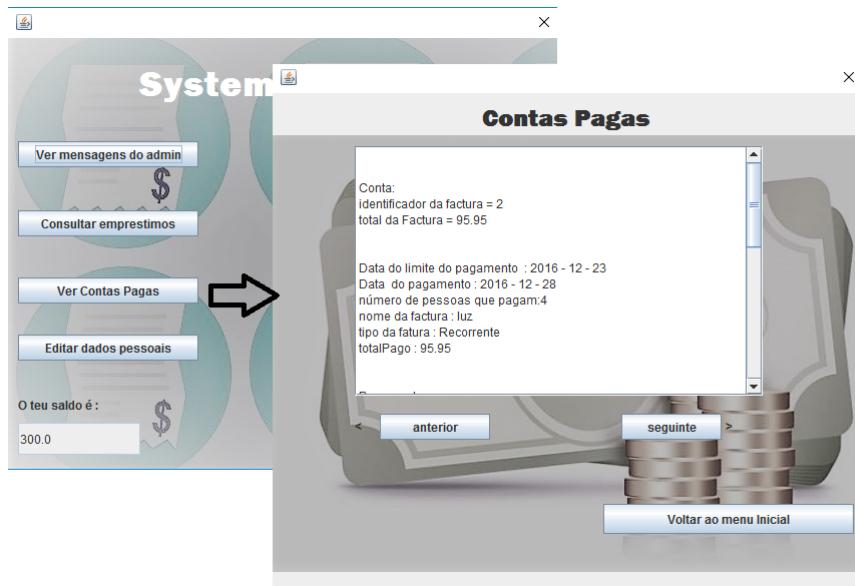


Figura 5.30: Janela para consultar a lista das contas já pagas

Cada utilizador tem a facilidade de poder editar os seus dados. Pode alterar o seu e mail, número de telemóvel, entre outros dados que possam estar mal introduzidos.

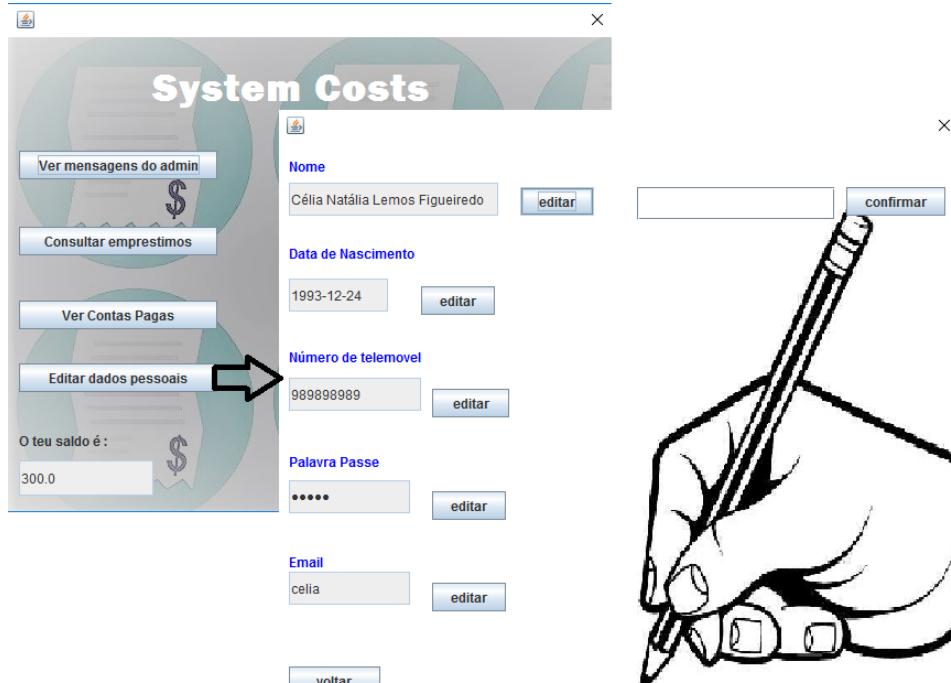


Figura 5.31: Janela para editar dados pessoais

Nesta janela cada morador tem acesso à lista de valores das suas dívidas relativas ao apartamento, isto é, o valor que cada morador fica em dívida para com o apartamento. Cada dívida é identificada por um e mail, um nome, um valor e uma data da dívida.

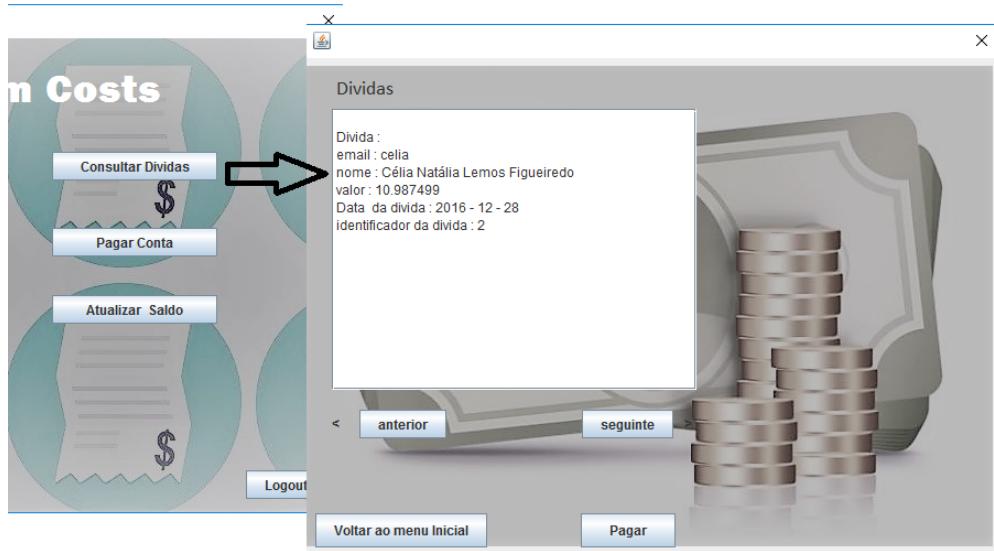


Figura 5.32: Janela para consultar dívidas

Para fazer o pagamento de determinada fatura, o morador acede a “pagar conta”, onde tem acesso às listas de todas as contas que o administrador previamente colocou em sistema, assim como, os detalhes de cada uma delas. Desta forma, apenas seleciona através do identificador da fatura a conta a pagar e o valor que pretende pagar.

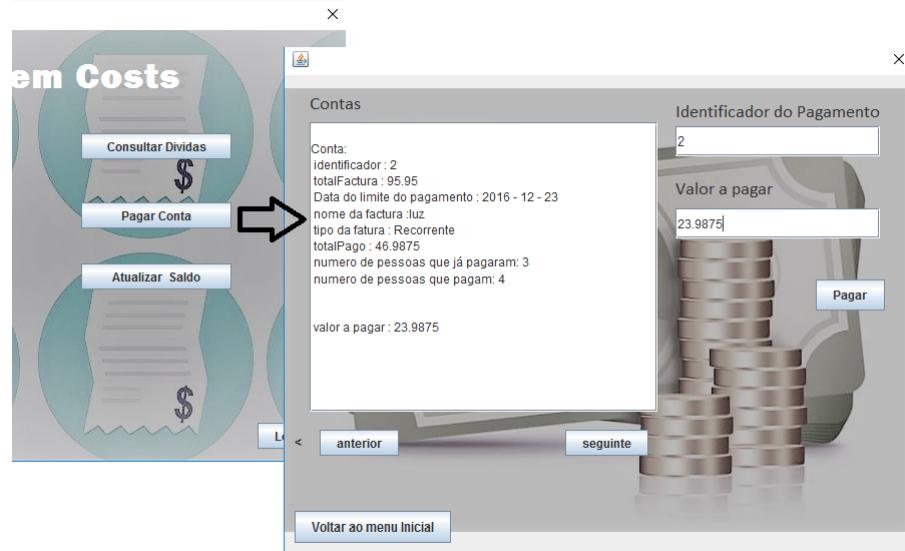


Figura 5.33: Janela para efetuar pagamento de despesa

Esta janela é útil para cada morador efetuar uma atualização do valor da sua conta corrente. Sempre que o morador apresentar um valor insuficiente para o pagamento de determinada fatura é aqui que deve fazer o seu “carregamento” de saldo.

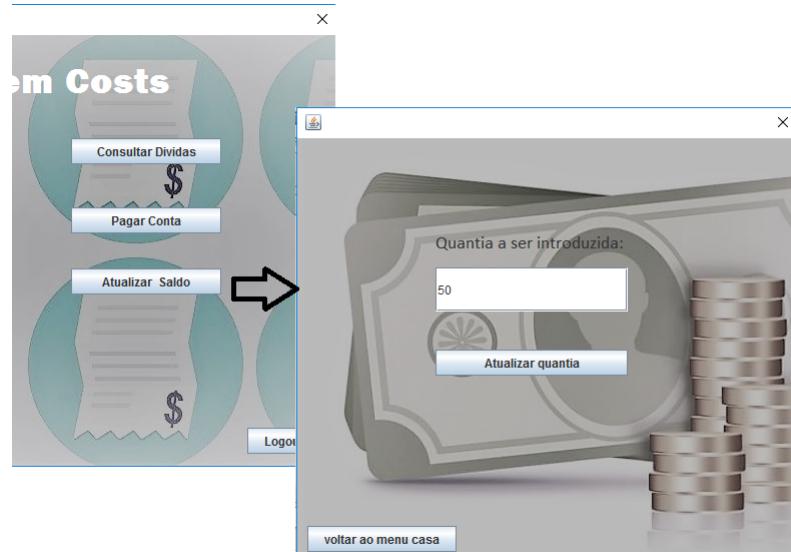


Figura 5.34: Janela para atualizar conta corrente

6. Conclusão

Para concluir a primeira fase de modelação do projeto que consistiu em desenvolver um sistema de suporte à partilha de despesas num apartamento, foi proposto enquadrar e descrever da forma mais detalhada possível o sistema a ser desenvolvido. Procedemos a uma primeira fase de investigação sobre como funciona a partilha de despesas dentro de uma casa /apartamento, investigação essa que foi relativamente simples dado que alguns dos elementos do grupo estão familiarizados com o problema em questão. A partir daí fizemos uma descrição do processo de análise de requisitos construindo assim o modelo de domínio onde foi utilizada a linguagem UML. Do modelo de domínio e requisitos do sistema foi possível desenvolver os diagramas de 'Use Case' e a posterior especificação de cada um deles.

Depois de todos estes elementos procedemos à fase de pensar de como seria a nossa aplicação fisicamente falando e então para tal utilizamos o programa 'Pencil' e desta forma desenvolvemos a nossa primeira proposta de interface com o utilizador. Construímos também os diagramas de máquinas de estado de acordo com a interface pensada.

Um dos principais problemas que encontramos foi a modelação do Modelo de Domínio, pois estavam sempre a surgir novos requisitos e a serem eliminados outros assim como nas especificações dos 'Use Case'. Podemos concluir que esta é uma etapa que requer uma análise muito cuidada, pois estão sempre a surgir novas maneiras de abordar o projeto.

Para a fase final do projeto desenvolvemos os "Diagramas de Sequência" com o objetivo de descrever o funcionamento da aplicação e percebemos que alguns "use cases" precisavam de ser alterados, na parte das especificações pois a maneira pensada inicialmente não estava bem conseguida. Depois de muita discussão e partilha de ideias concebemos a aplicação com o objetivo de ser eficiente e simples de usar, com uma interface apelativa, até porque se idealizarmos algo demasiado complexo, muito provavelmente a escassez de tempo juntamente com a possível falta de conhecimentos, poderiam levar a uma má implementação ou até mesmo à impossibilidade de concluirmos o trabalho. A solução usada para o problema pareceu-nos a melhor e fácil de implementar, contudo, esta solução está dependente do fator seriedade dos moradores da casa, pois existem certas funcionalidades que podem causar certa controvérsia. Por exemplo: quando um morador empresta dinheiro à casa poderá nunca o vir a receber, visto que aplicação só gere despesas.

Em relação a trabalho futuro poderá ser implementado um sistema que permita a troca de mensagens entre os utilizadores e não só o administrador a enviar mensagens, pois a comunicação deverá ser feita entre todos os intervenientes do grupo. Além disso, convém ao administrador saber se uma mensagem foi vista por todos, ou quantos/quem ainda não viu essa mensagem, para que assim o administrador possa posteriormente apagar a mensagem pois já não tem utilidade.

Outro aspecto a melhorar é o caso de o transferir dinheiro de uma conta para outra, ficando assim uma dívida direta. Também na apresentação das contas/mensagens o utilizador visualiza a mensagem/conta e só quando clicar é que consegue ver o detalhe das mesmas.

Aplicação foi submetida a testes, e corrigida quando os testes falharam no entanto há sempre testes que não conseguimos até ao momento prever.