

# Relatório

## DESENVOLVIMENTO SISTEMAS E SOFTWARE 2016



## Índice

Introdução .....	1
Modelo de Domínio.....	2
Diagrama de Use Case .....	4
Sistema de Partilha de Despesas.....	4
Gestão de Despesas.....	5
Gestão de Pagamentos .....	6
Gestão de Consultas.....	7
Especificações dos Use Cases .....	8
Registar .....	8
Login .....	9
Adicionar Despesa - Equitativo.....	10
Adicionar Despesa - Percentual .....	11
Editar Despesa.....	12
Eliminar Despesa .....	13
Efetuar Pagamento .....	14
Eliminar Pagamento.....	15
Consultar Despesas (Individuais) em Atraso.....	16
Consultar Pagamentos Efetuados .....	17
Consultar Pagamentos Recebidos .....	18
Consultar Estado das Despesas Adicionadas.....	19
Mokups da Interface .....	20
Menu de Login e Registo .....	20
Menu de Despesas .....	21
Janela de Adicionar Despesa .....	22
Janela de Editar Despesa .....	23
Menu de Despesas Individuais .....	24
Janela de Efetuar Pagamento .....	25
Pagamentos .....	26
Máquinas de Estado .....	27
Comportamento da interface .....	27

Diagramas de Sequencia .....	29
Registrar Morador: .....	29
Adicionar Despesa – Equitativo .....	30
Efetuar Pagamento .....	31
Consultar Despesas .....	32
Registrar Morador .....	33
Adicionar Despesa – Equitativo .....	34
Efetuar Pagamento .....	35
Consultar Despesas .....	36
Diagrama de Package .....	37
Diagramas de Classe e de Sequência .....	38
Registrar Morador .....	39
Adicionar Despesa - Equitativo .....	40
Efetuar Pagamento .....	41
Consultar Despesas .....	42
Diagrama de Instalação .....	45
Implementação .....	46
Conclusão .....	47
Anexos .....	48
Adicionar despesa percentual .....	48
Consultar despesas individuais em atraso .....	49
Consultar estado das despesas adicionadas .....	49
Consultar pagamentos .....	49
Editar Despesa .....	50
Eliminar despesa .....	51
Eliminar pagamento .....	52
Login .....	53
Adicionar despesa – Percentual (Subsistemas) .....	54
Consultar despesas individuais (Subsistemas) .....	55
Consultar estado das despesas adicionadas (Subsistemas) .....	56
Consultar pagamentos (Subsistemas) .....	57
Editar Despesa (Subsistemas) .....	58

Eliminar despesa (Subsistemas) .....	59
Eliminar pagamento (Subsistemas) .....	60
Login (Subsistemas) .....	61
Adicionar despesa – Percentual (Classes) .....	62
Consultar despesas individuais (Classes) .....	63
Editar Despesa (Classes) .....	64
Eliminar Despesa (Classes).....	65
Eliminar Pagamento (Classe) .....	66
Login (Classes) .....	67

## Introdução

---

O trabalho prático da disciplina de DSS propõe o desenvolvimento de uma aplicação com base num sistema de suporte à partilha de despesas de um apartamento. Esta aplicação deverá suportar o registo de despesas e gestão de pagamentos. É pedido, também, que seja possível seleccionar quais os utilizadores a partilhar uma despesa.

A nossa aplicação dá ao utilizador a oportunidade de gerir as despesas distribuídas por um apartamento partilhado, isto é, despesas recorrentes, como a conta da luz e de água, mas também despesas extraordinárias, como a reparação de um eletrodoméstico ou cuidados a ter com um animal de estimação.

Temos também diferentes formas de distribuir as despesas. Sendo a primeira equitativamente, isto é, dividir as despesas de forma igual por todos os moradores do apartamento. E a segunda percentualmente que implica dividir os custos de maneira diferente dando uma maior percentagem a uns moradores do que outros.

---

## Modelo de Domínio

---

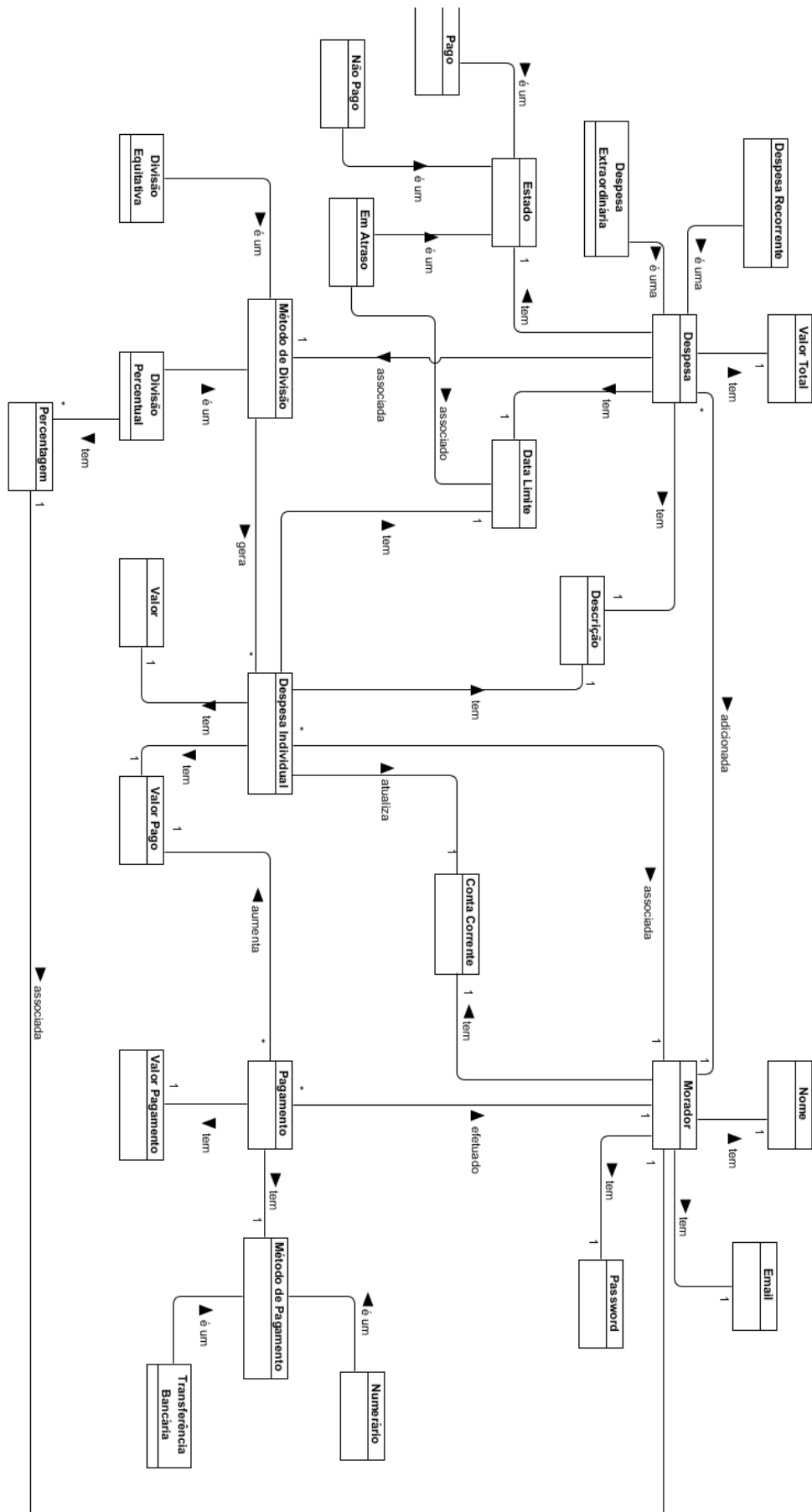
Para o modelo de domínio decidimos que seria oportuno, a existência de uma entidade morador. A cada morador está associado um nome, um email e uma password.

A entidade despesa tem um valor total e uma descrição própria, e a possibilidade de ser ou uma despesa recorrente ou extraordinária. Existem dois tipos de divisões possíveis de despesas: a equitativa, onde a despesa é dividida em partes iguais por todos, e a percentual, onde a cada morador está associada a percentagem do valor total a pagar.

A despesa individual e o seu valor são gerados pelo método de divisão. Numa divisão equitativa, o valor corresponde à divisão do valor total pelo número de moradores associados à despesa. Numa divisão percentual, o valor é calculado a partir do valor total e da percentagem correspondente.

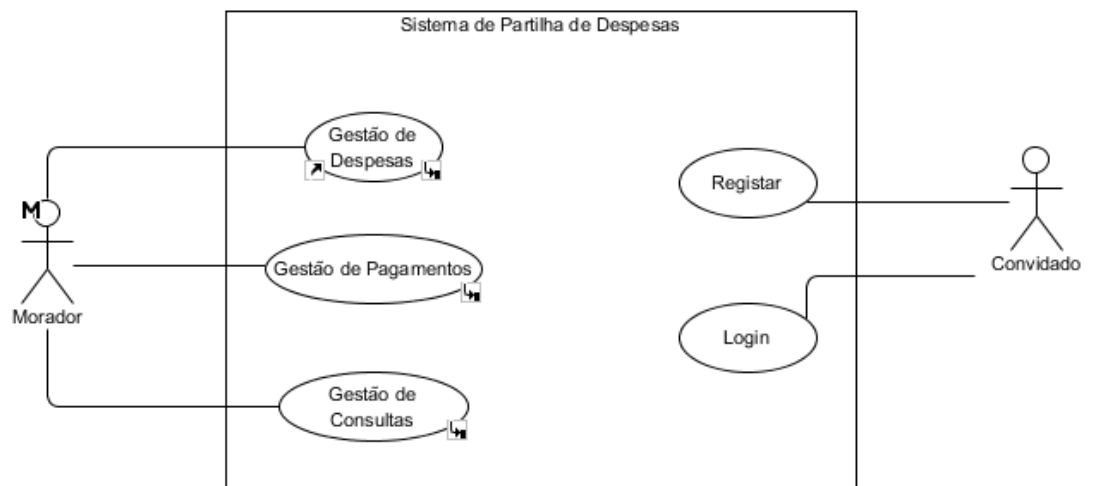
Um morador ao fazer um pagamento aumenta o valor pago associado à sua despesa individual. O morador pode fazer vários pagamentos, com valores diferentes, da mesma despesa individual. Os pagamentos podem ser feitos de duas formas diferentes: numerário ou transferência bancária.

Uma implementação alternativa para esta aplicação seriam despesas partilhadas pelos moradores onde cada morador iria pagar a sua parte, até que a despesa fosse completamente liquidada. Esta implementação apesar de simples, não seria eficiente, flexível nem segura uma vez que sendo a despesa partilhada, cada morador poderia apagar a despesa sem que ela fosse completamente paga, causando diversos problemas.



## Diagrama de Use Case

### Sistema de Partilha de Despesas



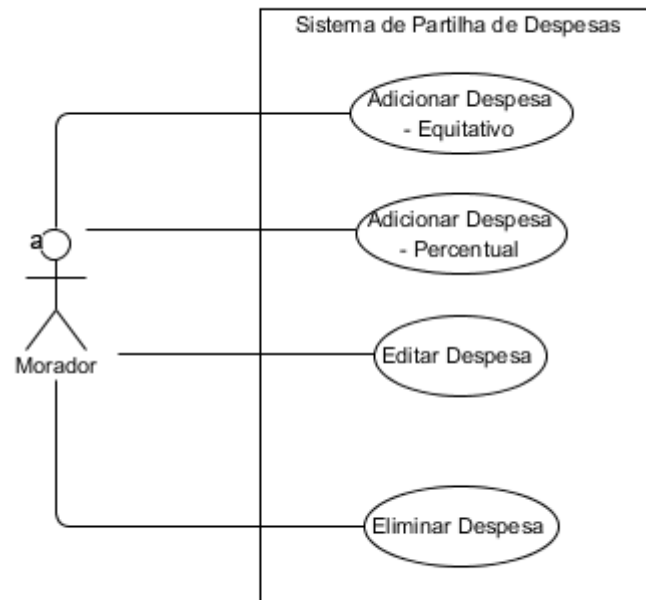
Neste Sistema de partilha de despesas, o ator, neste caso o Morador, tem de ser capaz de fazer a gestão das suas despesas, de gerir os seus pagamentos e ainda fazer consultas sobre as mesmas.

O ator convidado simboliza um morador que ainda não se registou no sistema e por isso tem de ter a opção de se registar. Depois disto este torna-se capaz de fazer login.

Para não subcarregar este diagrama decidimos criar três subdiagramas de use case para explicar todo o sistema de uma forma mais clara.



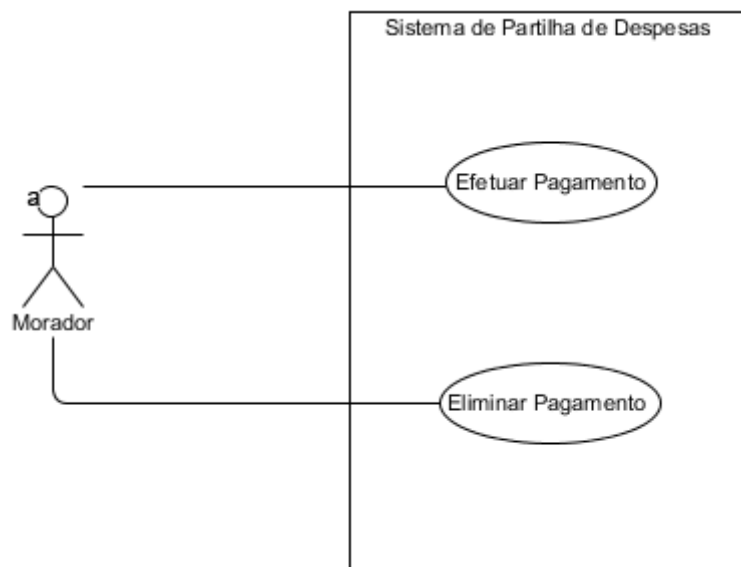
## Gestão de Despesas



Este Subdiagrama foi criado com o intuito de simplificar o entendimento do use case “Gestão de Despesas”.

O ator Morador, para gerir as suas despesas, necessita de ser capaz de adicionar, editar e eliminar despesas. Isto permite-lhe ter todas as suas despesas atualizadas e ainda corrigir qualquer erro (como por exemplo: o valor da despesa, a descrição ou até o valor da despesa) que tenha sido cometido na introdução de uma despesa.

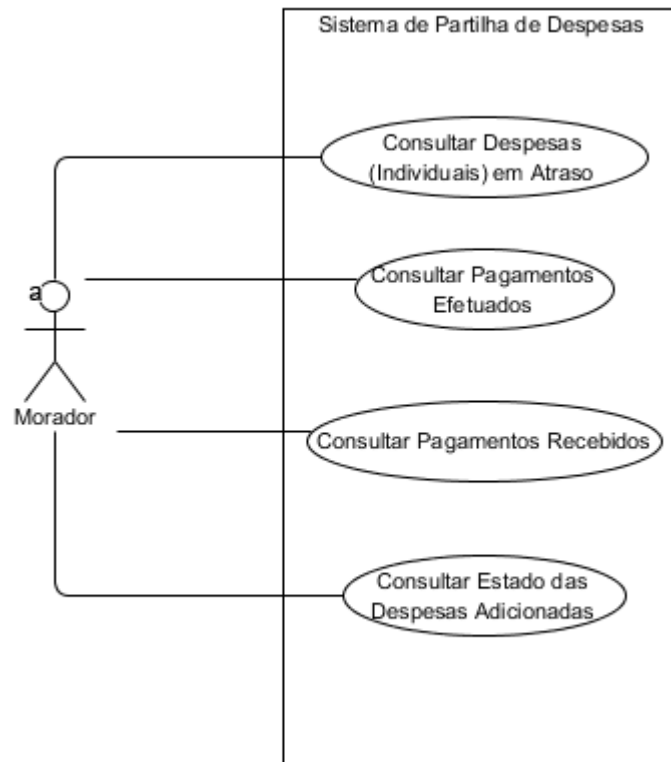
### Gestão de Pagamentos



Este Subdiagrama simplifica o use case “Gestão de Pagamentos”.

Esta gestão de pagamentos tem como objetivo atualizar os pagamentos efetuando-os e eliminando-os.

### Gestão de Consultas



Este Subdiagrama descreve o use case “Gestão de Consultas”.

Ao consultar as suas despesas, o ator Morador tem a opção de consultar as suas despesas individuais em atraso, os pagamentos que já efetuou, os pagamentos recebidos e ainda o estado das suas despesas adicionadas.

Com isto o Morador é capaz de analisar todas as despesas que estejam relacionadas com o mesmo e assim simplificar toda a gestão necessária para este sistema.

Especificações dos Use Cases

Registrar

Preconditions			
Post-conditions	Passa a estar registado		
Flow of Events		Actor Input	System Response
	1	Inserir dados de registo (nome, email password)	
	2		Valida dados inseridos
	3		Regista novo utilizador
Excecao 2 [Email já existe] (Passo 2)		Actor Input	System Response
	1		Notifica que email já existe

**Registrar morador:** Para o Utilizador poder fazer LOGIN tem de estar registado na aplicação. Os dados que insere são os seguintes: Nome, Email, Password. O registo só será concretizado se o Email ainda não tiver sido registado.

## Login

Preconditions		
Post-conditions	Utilizador fica autenticado	
Flow of Events		Actor Input
	1	Introduz os dados de login (E-mail e Password)
	2	Valida dados de login
	3	Autentica utilizador
Excecao 1 [Dados Inválidos] (Passo 2)		Actor Input
	1	Notifica que dados introduzidos são inválidos

**Efetuar Login:** Utilizador ao iniciar aplicação tem de fazer o login. O login só será efetuado com sucesso se o utilizador já estiver registado e os dados introduzidos (Email e Password) estiverem corretos.

### Adicionar Despesa - Equitativo

Preconditions	Morador autenticado		
Post-conditions	Despesa registada		
Flow of Events		Actor Input	System Response
	1	Introduz dados de despesa (Descrição, Data Limite, Valor, Tipo e Moradores)	
	2		Valida dados (Valor e Moradores)
	3		Regista despesa
	4		Notifica despesa registada
Excecao 1 [Dados Invalidos] (Passo 2)		Actor Input	System Response
	1		Notifica Dados inválidos

**Adicionar despesa:** Para um Morador poder adicionar uma despesa tem de estar autenticado (fazer LOGIN).

O morador insere a Descrição da despesa, a Data Limite que tem de ser paga pelos outros moradores, o Valor da despesa, o Tipo da despesa (Recorrente ou Extraordinária) e a lista de moradores com quem vai ser dividida a despesa. De seguida o Valor e a lista de Moradores são validados (Valor tem de ser positivo e todos os moradores inseridos tem de estar registados).

Caso os dados sejam validos, é criada a despesa e as despesas individuais para cada morador, com um valor igual para cada um.

### Adicionar Despesa - Percentual

Preconditions	Morador esta autenticado		
Post-conditions	Despesa adicionada		
Flow of Events		Actor Input	System Response
	1	Introduz dados de despesa (Descrição, Data Limite, Valor, Tipo, Moradores e Percentagens)	
	2		Valida dados (Valor, Moradores e Percentagens)
	3		Regista despesa
	4		Notifica despesa registada
Excecao 1 [Dados Invalidos] (Passo 2)		Actor Input	System Response
	1		Notifica dados invalidos

**Adicionar Despesa:** O procedimento é identico ao do use case “Adicionar Despesa – Equitativo” com a diferença da lista de percentagens. A validação desta lista ocorre da seguinte forma: verificar se todas as percentagens são positivas e se a soma delas não resulta num valor superior a 100. É também verificado se o tamanho da lista de moradores é igual ao tamanho da lista de percentagens.

## Editar Despesa

Preconditions	Morador autenticado e tem despesas adicionadas		
Post-conditions	Despesa é editada		
Flow of Events		Actor Input	System Response
	1	Seleciona despesa a editar e insere data limite, descrição e valor	
	2		Valida valor
	3		Regista alterações
	4		Notifica despesa alterada
Excecao 1 [Valor não pode ser alterado] (Passo 2)		Actor Input	System Response
	1		Notifica valor não pode ser alterado
	2		Regressa a 2

**Editar despesa:** Para um Morador poder editar uma despesa tem de, anteriormente, ter adicionado uma ou mais despesas e estar autenticado.

O morador seleciona uma das suas despesas e indica a nova Data Limite, a nova Descrição e o novo Valor.

O Valor da despesa apenas pode ser alterado quando ainda não foi feito nenhum pagamento dessa despesa. É necessário também verificar se o novo valor é um valor positivo

Por fim, a despesa é alterada.



## Eliminar Despesa

Super Use Case		
Author	Fábio Baião	
Date	8/Nov/2016 19:33:48	
Brief Description		
Preconditions	Morador autenticado e tem despesas que podem ser eliminadas.	
Post-conditions	Despesa eliminada	
Flow of Events		<b>Actor Input</b>
		<b>System Response</b>
	1	Seleciona despesa a eliminar
	2	Elimina despesa
	3	Notifica despesa eliminada

**Eliminar despesa:** Para um Morador poder eliminar uma despesa tem de ter adicionado uma ou mais despesas que já estejam pagas ou que ainda não receberam nenhum pagamento e estar autenticado.

O processo de eliminação é muito simples: o Morador seleciona a despesa que pretende eliminar e a despesa é eliminada.

## Efetuar Pagamento

Preconditions	Morador Autenticado e com despesas por pagar		
Post-conditions	Pagamento efetuado		
Flow of Events		Actor Input	System Response
	1	Seleciona despesa e introduz valor a pagar e método de pagamento e data do pagamento	
	2		Valida valor de pagamento
	3		Regista pagamento efetuado
	4		Notifica pagamento efetuado
Excecao 1 [Valor de pagamento inválido] (Passo 2)		Actor Input	System Response
	1		Notifica valor inválido

**Efetuar Pagamento:** um morador para poder fazer um pagamento tem de ter despesas por pagar e estar autenticado.

O Morador seleciona a despesa que quer pagar, insere o valor a pagar, a data do pagamento e o método de pagamento utilizado. É feita a validação do valor do pagamento, tendo em conta que este tem de ser positivo e não pode ser superior ao valor que está a dever dessa despesa.

O pagamento é registado.

## Eliminar Pagamento

Super Use Case			
Author	joaor		
Date	13/nov/2016 13:48:06		
Brief Description			
Preconditions	Morador autenticado e com pagamentos que pode eliminar		
Post-conditions	Pagamento eliminado		
Flow of Events		Actor Input	System Response
	1	Seleciona pagamento a eliminar	
	2		Elimina pagamento escolhido
	3		Notifica pagamento eliminado

**Eliminar pagamento:** Um morador para poder eliminar um pagamento tem de ser o morador que fez o pagamento e tem, também, de estar autenticado.

Mais uma vez o funcionamento deste use case é muito simples: o morador seleciona o pagamento que pretende eliminar e o sistema elimina o pagamento.

### Consultar Despesas (Individuais) em Atraso

Super Use Case		
Author	joao	
Date	12/nov/2016 21:31:27	
Brief Description		
Preconditions	Está autenticado e tem despesas em atraso	
Post-conditions		
Flow of Events		Actor Input
	1	Pede a lista de despesas em atraso
	2	Calcula a lista de despesa em atraso
	3	Envia lista de despesas em atraso

**Consultar despesa (individuais) em atraso:** Um morador para poder fazer esta consulta tem de ter despesas por pagar e estar autenticado.

Se a pré-condição for cumprida o sistema simplesmente apresenta a lista das despesas que o Morador ainda não pagou com os seguintes detalhes: o morador a quem está a dever, o valor que falta pagar, a Data Limite de pagamento e a descrição da despesa.

É ainda apresentado o valor total que falta pagar por esse morador.

### Consultar Pagamentos Efetuados

Super Use Case		
Author	joaor	
Date	12/nov/2016 21:40:21	
Brief Description		
Preconditions	Está autenticado e realizou pagamentos	
Post-conditions		
Flow of Events		Actor Input
	1	Pede lista de pagamentos efetuados
	2	
	3	
		System Response
		Calcula lista de pagamentos efetuados
		Envia lista de pagamentos efetuados

**Consultar pagamentos efetuados:** Para um morador fazer esta consulta tem de ter efetuado pagamentos e estar autenticado.

O sistema apresenta, então, a lista de pagamentos efetuados com os seguintes detalhes: o morador a quem fez o pagamento, o valor do pagamento e o tipo de pagamento que foi utilizado.

É ainda apresentado o valor total de todos os pagamentos efetuados por esse morador.

**Consultar Pagamentos Recebidos**

Super Use Case		
Author	joaor	
Date	12/nov/2016 21:44:52	
Brief Description		
Preconditions	Está autenticado e recebeu pagamentos	
Post-conditions		
Flow of Events		<b>Actor Input</b>
		<b>System Response</b>
	1	Pede lista de pagamentos recebidos
	2	Calcula lista de pagamentos recebidos
	3	Envia lista de pagamentos recebidos

**Consultar pagamentos recebidos:** para um morador fazer esta consulta tem de ter recebido pagamentos e estar autenticado.

É então apresentada uma lista de todos os pagamentos que já recebeu com os seguintes detalhes: o morador que fez o pagamento, o valor do pagamento e o tipo de pagamento que foi utilizado.

É ainda apresentado o valor total de todos os pagamentos recebidos por esse morador.

### Consultar Estado das Despesas Adicionadas

Super Use Case			
Author	joaor		
Date	12/nov/2016 21:49:03		
Brief Description			
Preconditions	Está autenticado e adicionou despesas		
Post-conditions			
Flow of Events		Actor Input	System Response
	1	Pede lista de estados de despesas adicionadas	
	2		Calcula lista de estados de despesas adicionadas
	3		Envia lista de estados de despesas adicionadas

**Consultar estado das despesas adicionadas:** Para um Morador poder editar uma despesa tem de, anteriormente, ter adicionado uma ou mais despesas e estar autenticado.

São apresentadas todas as despesas que adicionou, cada uma com os seguintes detalhes: descrição da despesa, estado da despesa (Paga, Não Paga, Em Atraso), valor total a ser pago, data limite de pagamento, moradores com quem foi dividida a despesa e o valor individual a pagar.

## Mokups da Interface

### Menu de Login e Registo

The image shows a hand-drawn mockup of a login and registration interface on lined paper. The interface is divided into two main sections. The top section is for login, featuring a title bar with three buttons (minimize, maximize, close) in the top right corner. Below the title bar, there are two input fields: 'Email:' and 'Password:'. To the right of these fields is a button labeled 'Login'. The bottom section is for registration, featuring four input fields: 'Nome:', 'Email:', 'Password:', and 'Repetir Password:'. To the right of these fields is a button labeled 'Registrar'.

[-] [X]

Email:

Password:

Login

Nome:

Email:

Password:

Repetir Password:

Registrar



### Menu de Despesas

Hand-drawn sketch of a 'Menu de Despesas' interface. The interface includes a table with the following columns: 'Descrição', 'Data Limite', 'Valor Total', and 'Estado'. The table has 6 rows. To the right of the table is a vertical scrollbar. Below the table are three buttons: 'Adicionar', 'Editar', and 'Eliminar'. In the top right corner, there are three small boxes containing the symbols 'L', 'Q', and 'X'.

Descrição	Data Limite	Valor Total	Estado

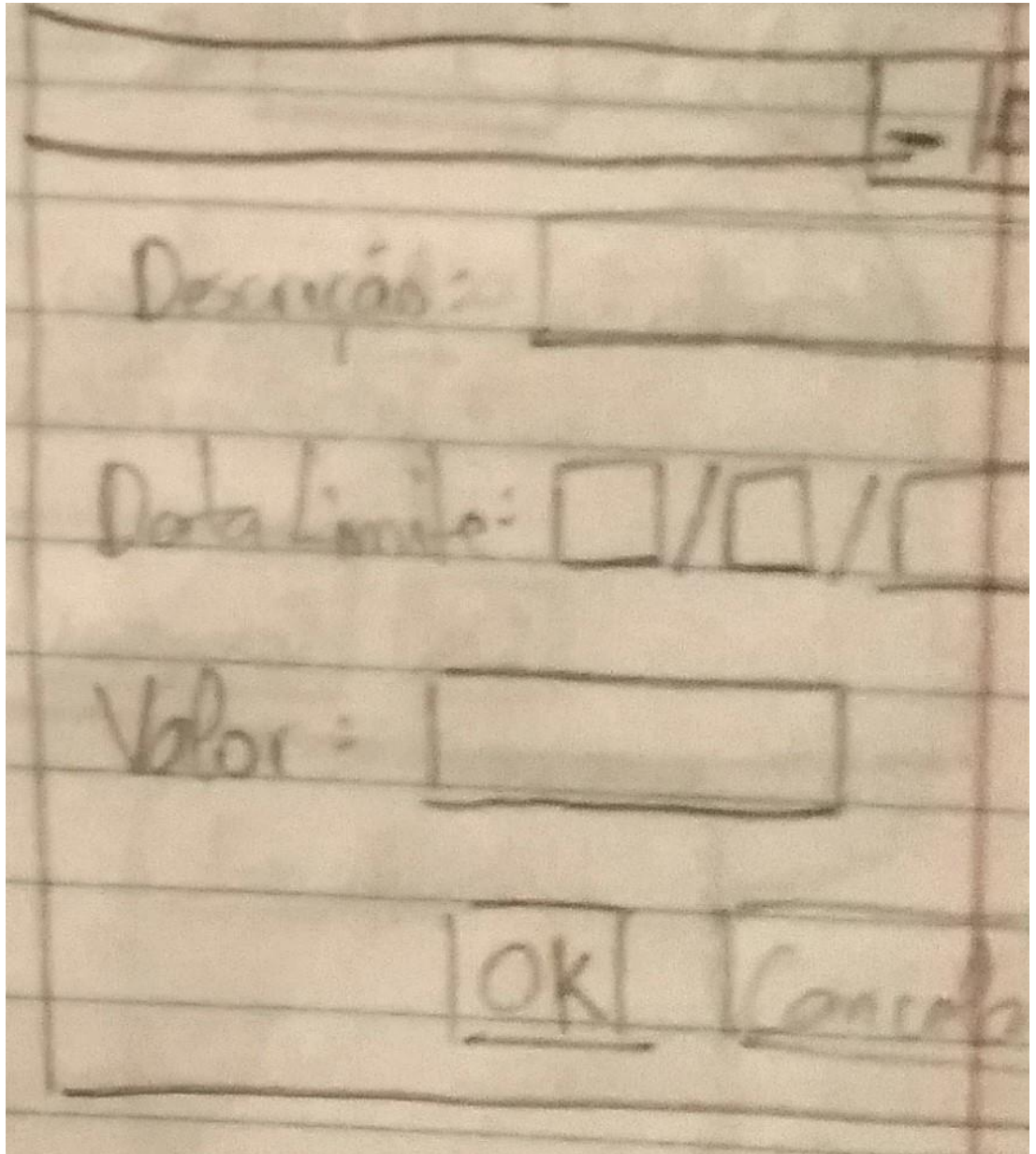
Adicionar    Editar    Eliminar

### Janela de Adicionar Despesa

The mockup shows a window titled 'Janela de Adicionar Despesa' with standard window controls (minimize, maximize, close) in the top right corner. The form contains the following fields and options:

- Descrição:** A text input field for the expense description.
- Data Limite:** Three input fields for day, month, and year, separated by slashes.
- Valor:** A text input field for the amount, followed by a Euro symbol (€).
- Tipo:** Two radio button options: ☐ Recorrente and ☐ Extraordinária.
- Divisão:** Two radio button options: ☐ Equitativa and ☐ Percentual.
- Modalidades:** A section with two columns. The left column is labeled 'Modalidades' and the right column is labeled 'Percentagens'. Each column contains a large, empty rectangular box for input.
- Buttons:** Two buttons at the bottom right: 'OK' and 'Cancelar'.

Janela de Editar Despesa



Hand-drawn sketch of a "Janela de Editar Despesa" (Expense Edit Window) on lined paper. The sketch includes a title bar at the top with a close button (X). Below the title bar are three input fields: "Descrição:" (Description), "Data Limite:" (Due Date) with a date format of [ ]/[ ]/[ ], and "Valor:" (Value). At the bottom, there are two buttons labeled "OK" and "Cancelar" (Cancel).

### Menu de Despesas Individuais

The mockup shows a mobile application interface for managing individual expenses. It consists of a table with four columns: 'Descrição' (Description), 'Data Limite' (Due Date), 'Valor por pagar' (Amount to Pay), and 'Nome morador' (Landlord Name). The table has several rows, with the first row containing some faint, illegible text. A vertical scrollbar is located on the right side of the table. Below the table, there is a button labeled 'Efetuar Pagamento' (Make Payment).

Descrição	Data Limite	Valor por pagar	Nome morador

At the bottom of the screen, there is a button labeled "Efetuar Pagamento".

Janela de Efetuar Pagamento

A hand-drawn sketch of a payment window interface on lined paper. The sketch is enclosed in a rectangular border. At the top right, there is a small box containing a minus sign, a square, and an 'X'. Below this, the text 'Valor:' is followed by a rectangular input field and a Euro symbol (€). Underneath, the text 'Método:' is followed by two radio button options: 'Numeração' and 'Transferência'. At the bottom, there are two rectangular buttons labeled 'OK' and 'Cancelar'.



## Pagamentos

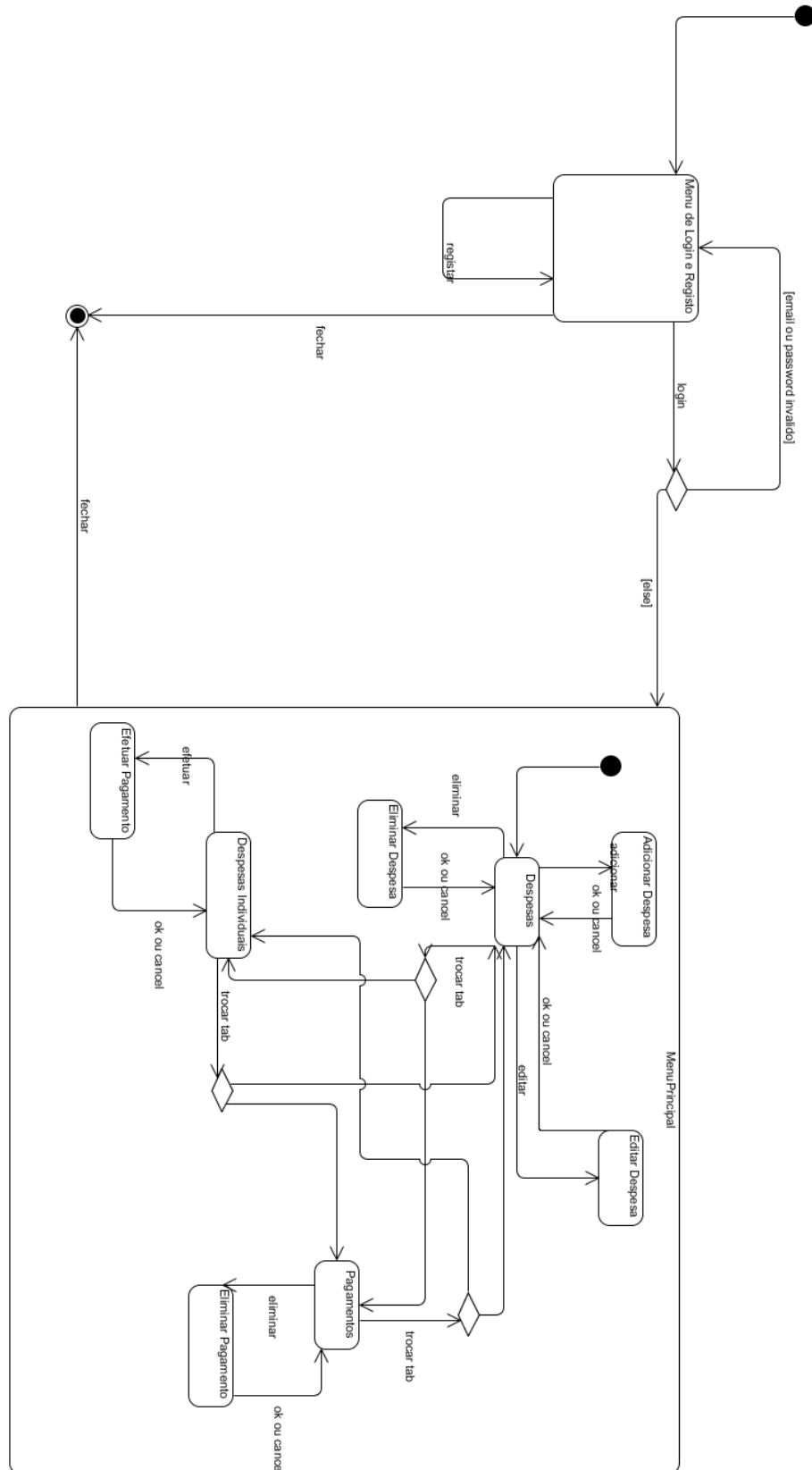
\_ | Q | X

Descrição	Data pagamento	Valor pago	Morador
			<div style="display: flex; justify-content: space-between; align-items: center;"> <span>Recibido</span> <span>▼</span> </div>
			<div style="display: flex; justify-content: space-between; align-items: center;"> <span>Eliminado</span> <span>▼</span> </div>

Eliminar pagamento

# Máquinas de Estado

## Comportamento da interface



Ao abrir a aplicação o primeiro menu que aparece é o menu de login e de registo. O Utilizador pode, assim, registar-se e/ou autenticar-se. Apenas quando fizer o login e os dados estiverem corretos aparece o menu principal.

Neste menu, é possível consultar as despesas adicionadas, as despesas individuais que ainda não foram pagas e todos os pagamentos efetuados.

No submenu das despesas é possível adicionar despesas, editar despesas e eliminar despesas.

No submenu das despesas individuais é possível efetuar pagamentos das despesas individuais que são apresentadas.

Por fim, no submenu dos pagamentos é possível eliminar pagamentos recebidos e efetuados.

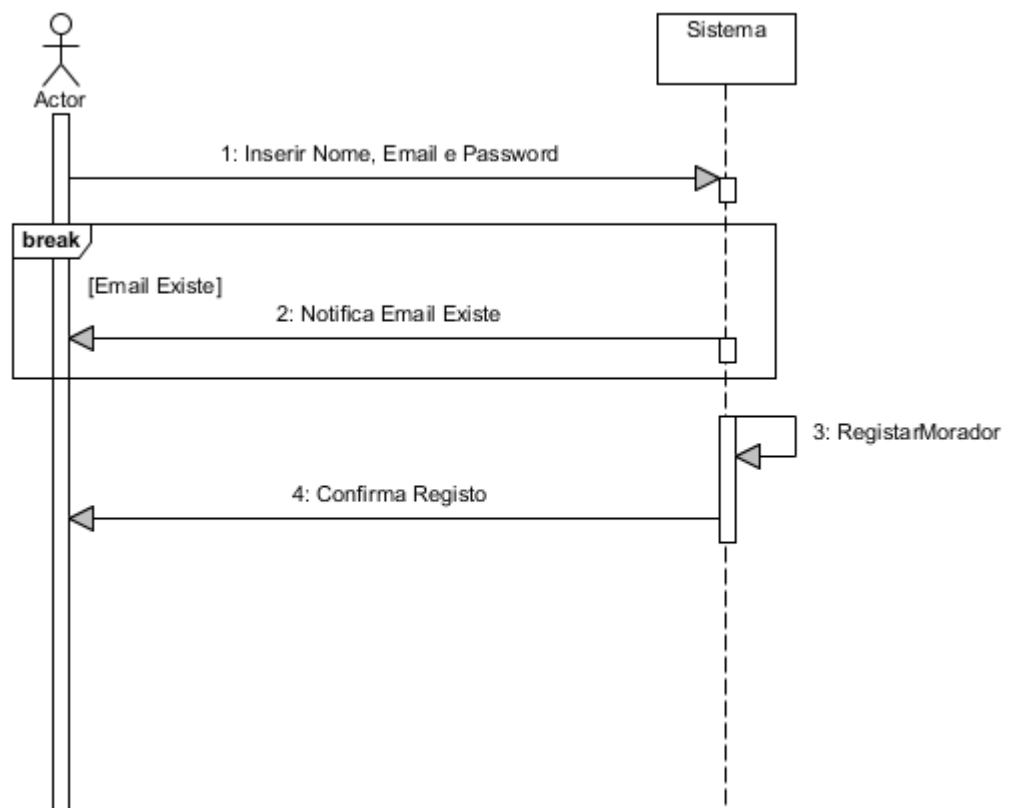


## Diagramas de Sequencia

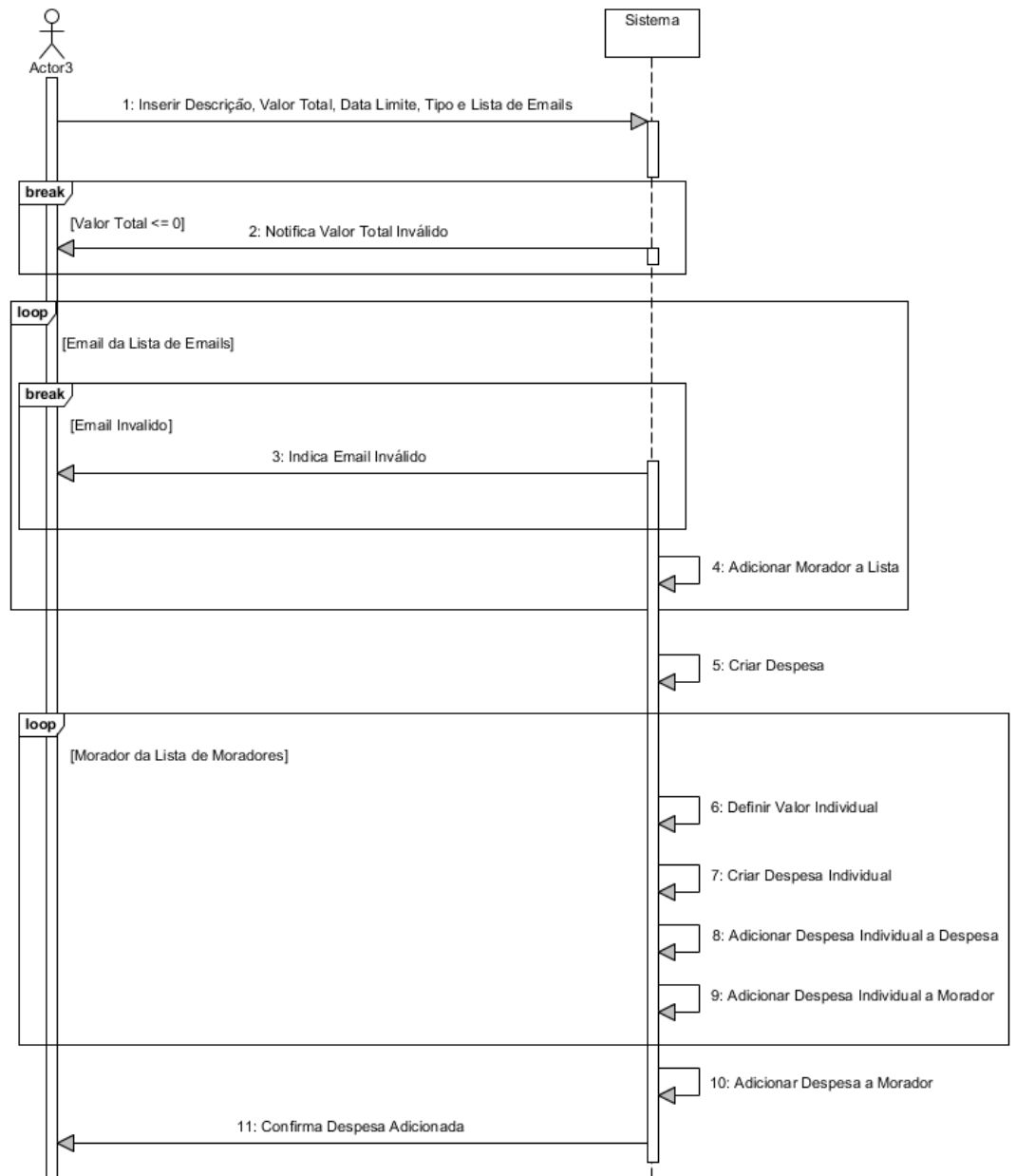
Nesta fase, o primeiro passo foi passar das especificações dos use cases para diagramas de sequência de sistema.

Apresentamos alguns exemplos de seguida:

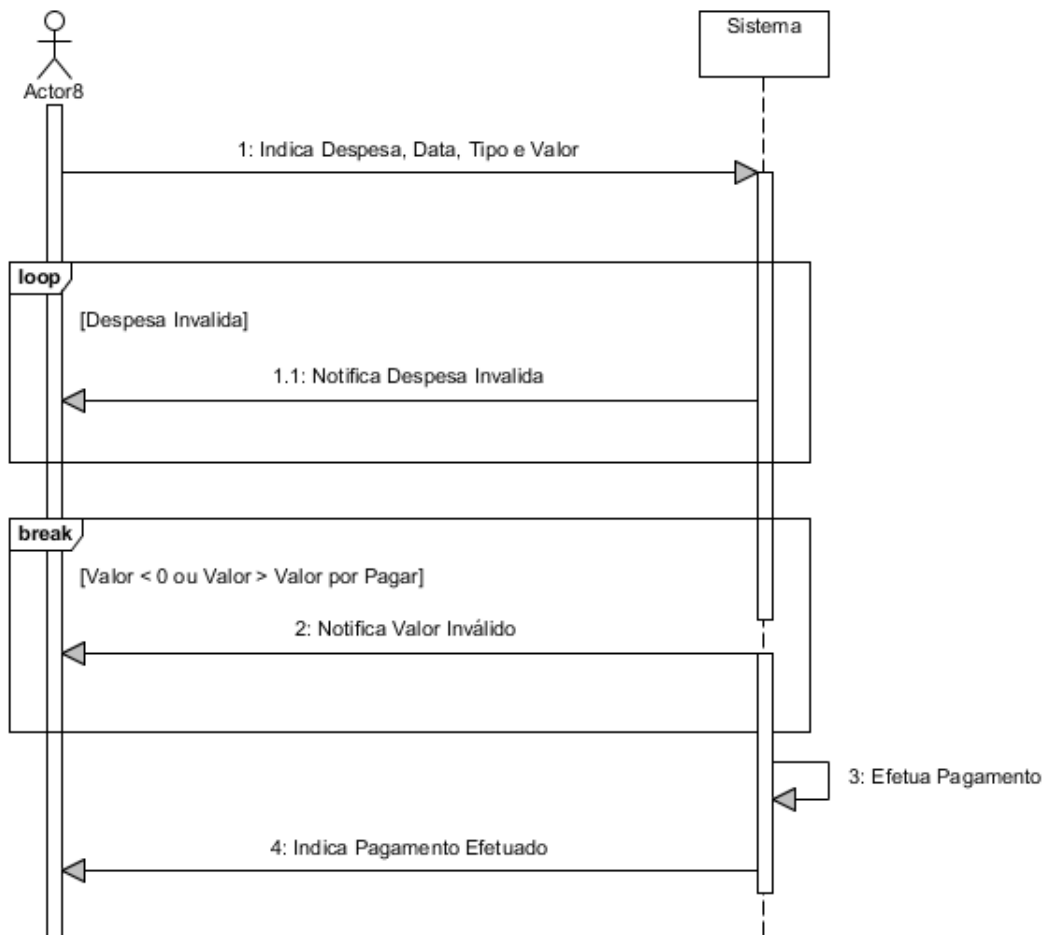
### Registrar Morador:



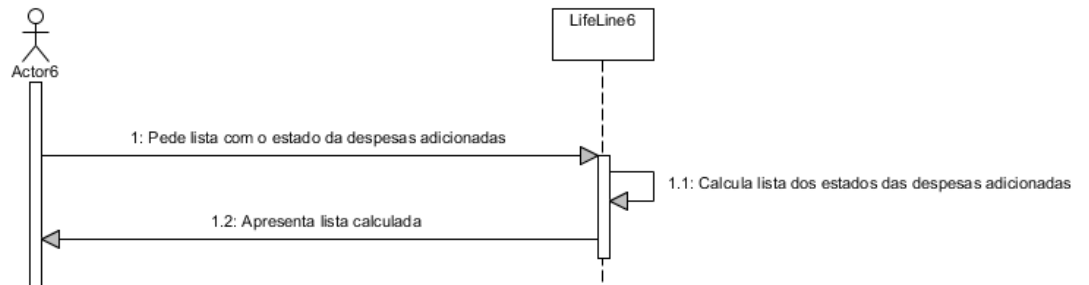
## Adicionar Despesa – Equitativo



## Efetuar Pagamento



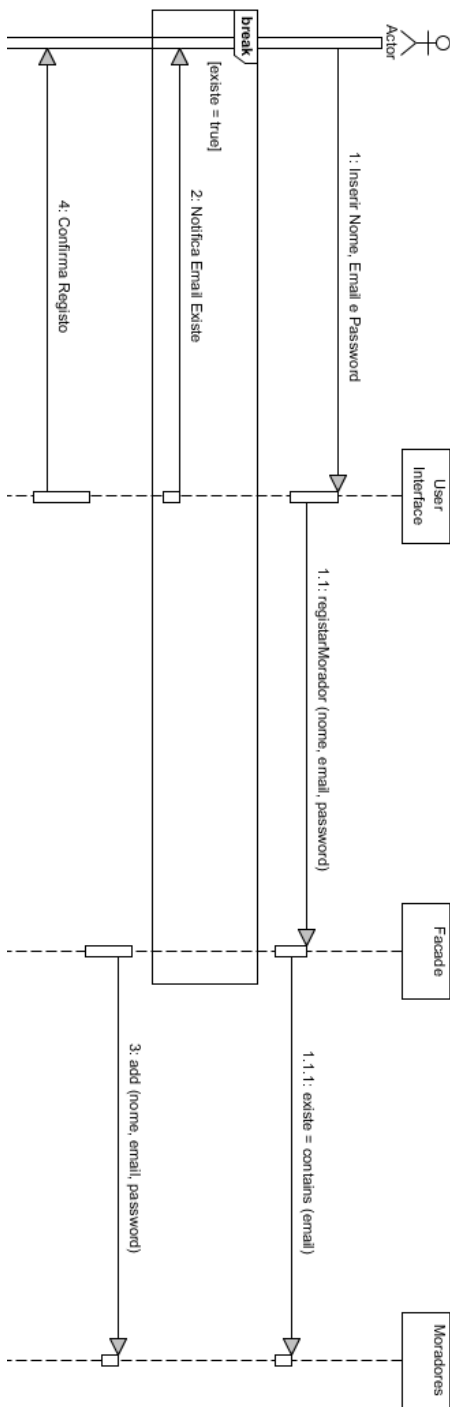
## Consultar Despesas



O passo seguinte que adotamos foi transformar os diagramas de sequencia de sistema em diagramas de sequencia de subsistemas. Para isso consideramos os seguintes subsistemas: User Interface, Facade, Moradores, Despesas e Pagamentos.

De seguida apresentamos esses diagramas dos mesmos use cases apresentados anteriormente:

Registrar Morador



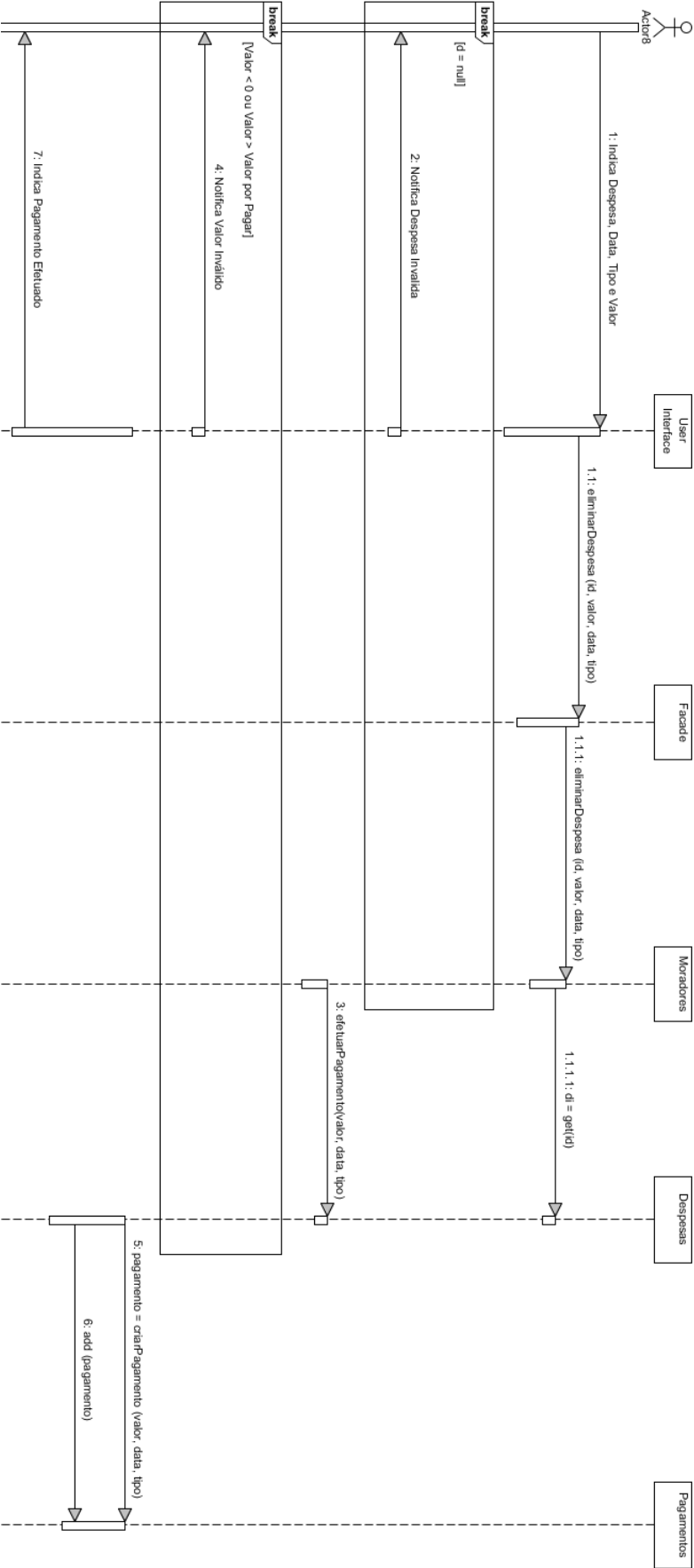
```

sequenceDiagram
    participant UI as User Interface
    participant Facade
    participant Mortadons
    participant Despesas

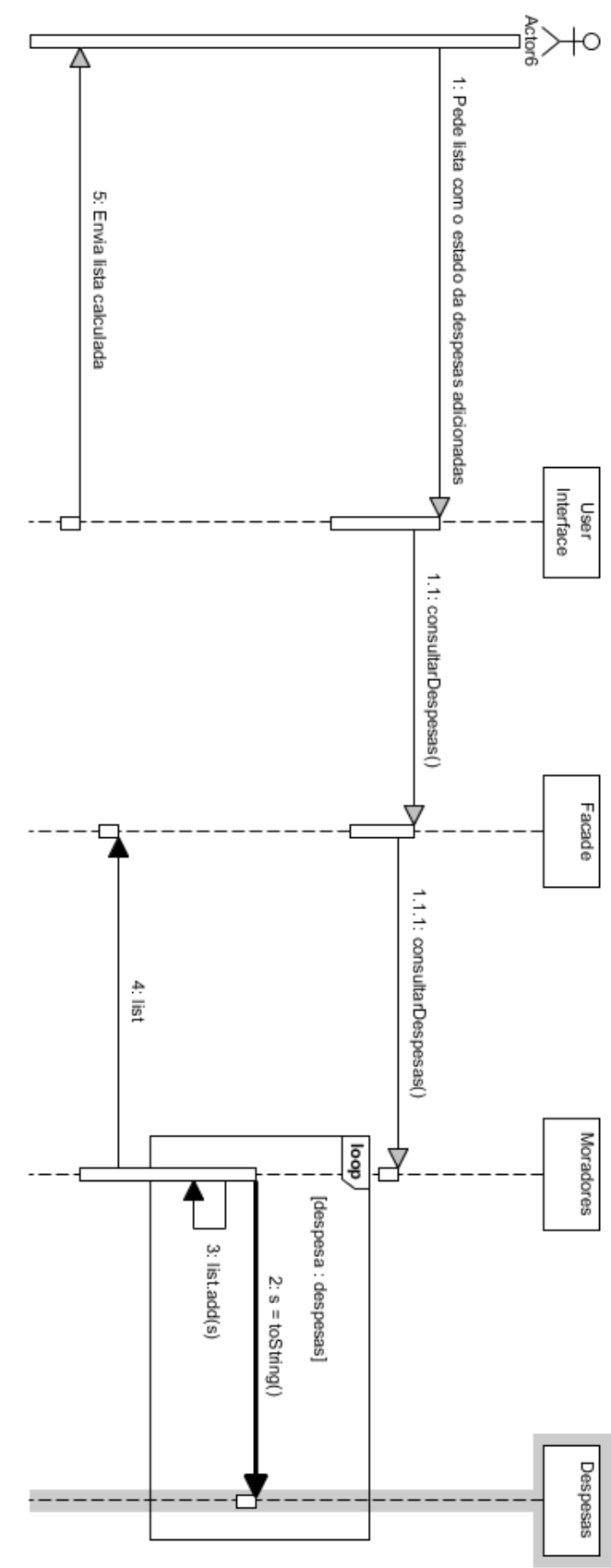
    UI->>Facade: 1: Inserir Descrição, Valor Total, Lista Emails, Data Limite e Tipo
    Facade->>Mortadons: 1.1: adicionarDespesa (descricao, valorTotal, listEmails, dataLimite, tipo)
    Mortadons->>Despesas: 6.1: d = novaDespesa (descricao, valorTotal, listEmails, dataLimite, tipo)
    Despesas->>Despesas: 7: valor
    Despesas->>Despesas: 8: d = criaDespesaIndividual (descricao, valor, dataLimite, tipo)
    Despesas->>Despesas: 9: add (d)
    Despesas->>Mortadons: 10: add (d)
    Mortadons->>Facade: 5: listMortadons.add(mortador)
    Facade->>UI: 3: mortador = get (email)
    Facade->>Facade: 4: Notifica Email Inválido
    Facade->>Facade: 11: Notifica Despesa Criada
    
```

The diagram illustrates the process of adding an expense to the system. It involves four main components: the User Interface, the Facade, the Mortadons (Managers) layer, and the Despesas (Expense) layer. The process begins with the User Interface sending a request to the Facade to add an expense. The Facade then delegates this task to the Mortadons layer. The Mortadons layer interacts with the Despesas layer to create and add individual expenses. Once the expense is added, the Mortadons layer returns the result to the Facade, which then sends a response back to the User Interface. The diagram also includes error handling for invalid emails and a notification for successful expense creation.

Efetuar Pagamento



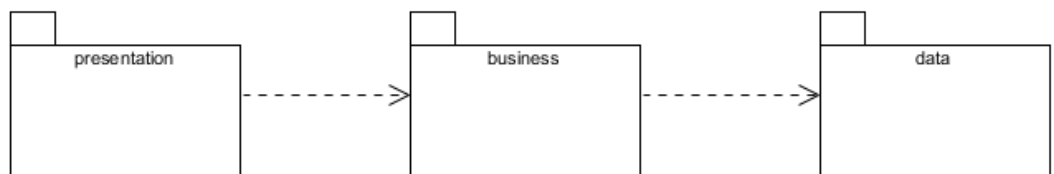
Consultar Despesas





## Diagrama de Package

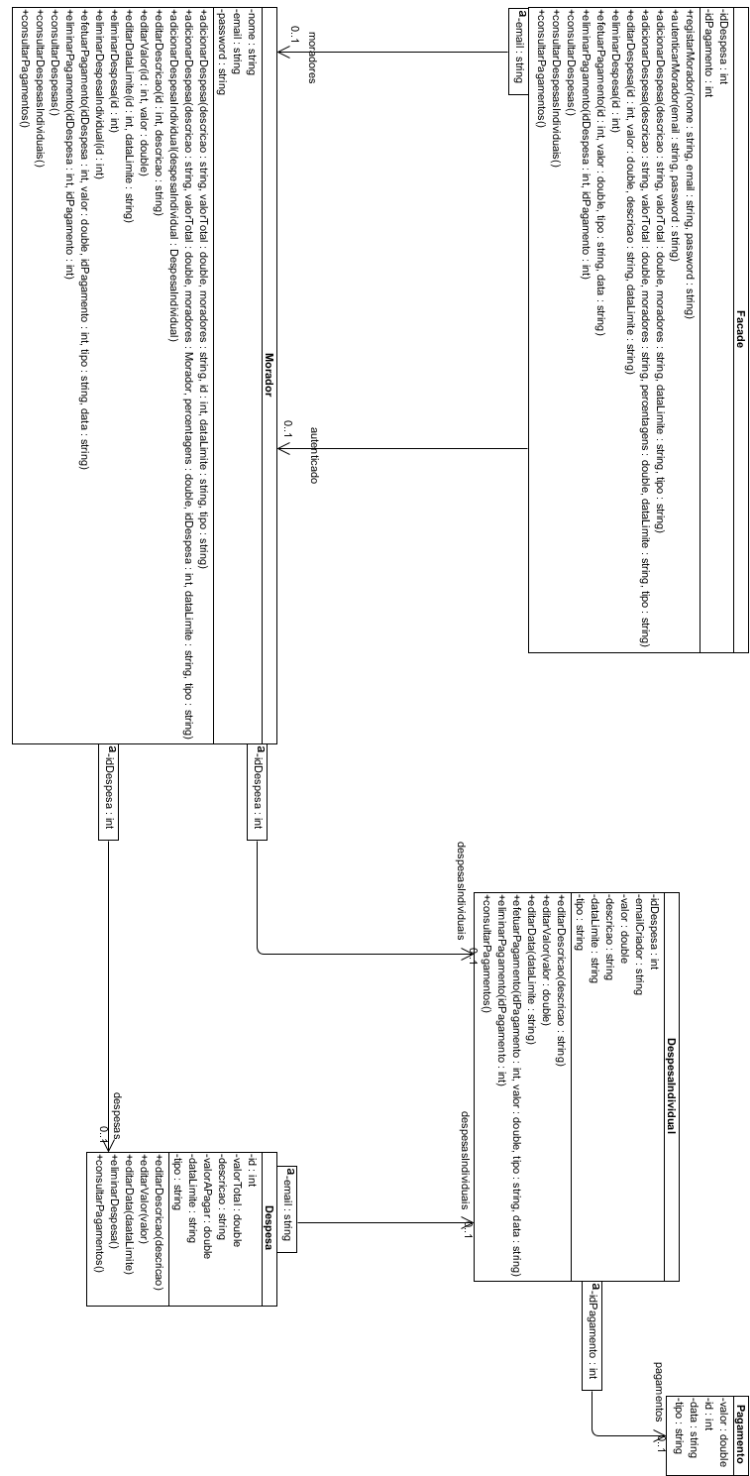
Depois de construir os diagramas de sequencia com subsistemas, desenvolvemos um diagrama de package. Rapidamente chegamos à conclusão que seria necessário ter os seguintes packages: presentation, business e data. Pensamos em criar os subpackages Despesas, Moradores e Pagamentos no package business, mas não concretizamos esta ideia, pois o numero de classes da camada de negocio será reduzido, logo não se justifica a criação de mais packages. Assim, chegamos a este diagrama:



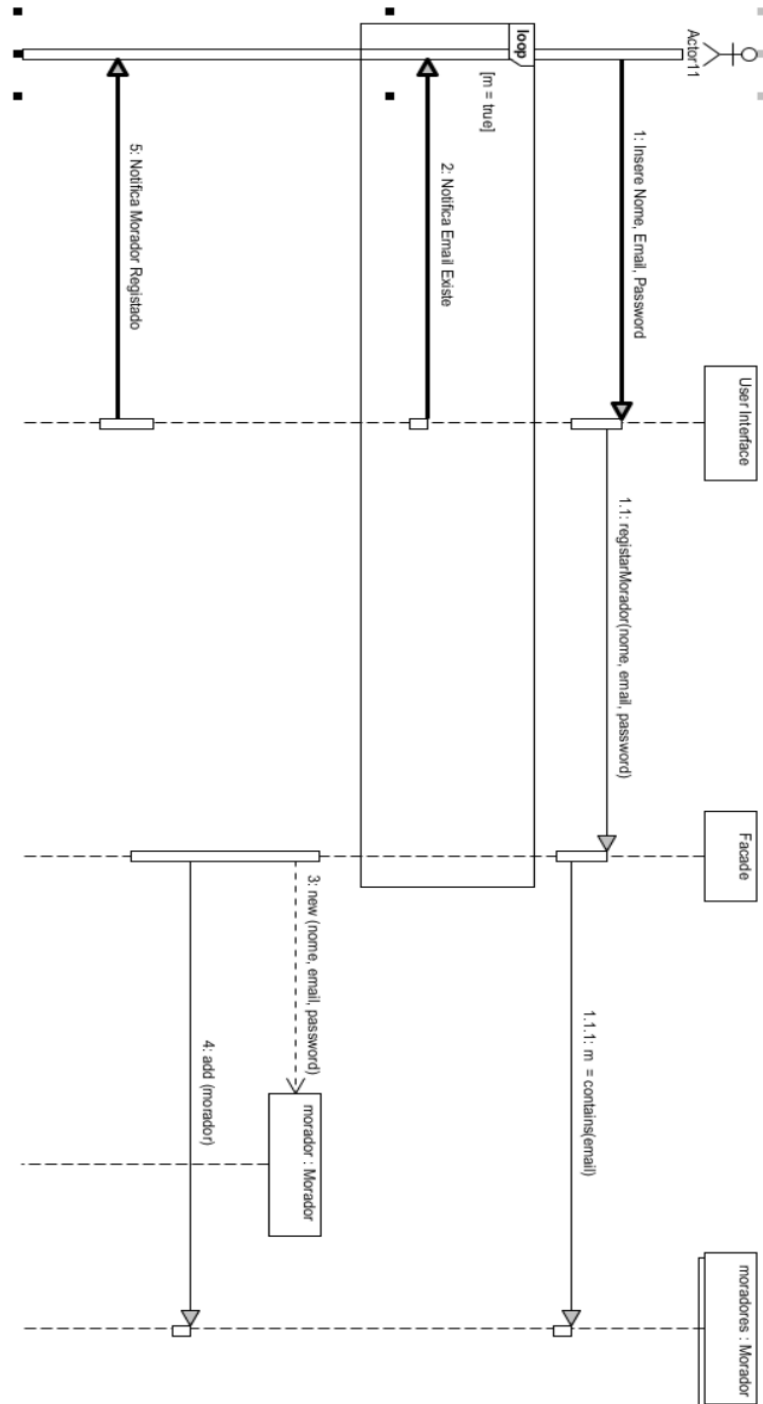
# Diagramas de Classe e de Sequência

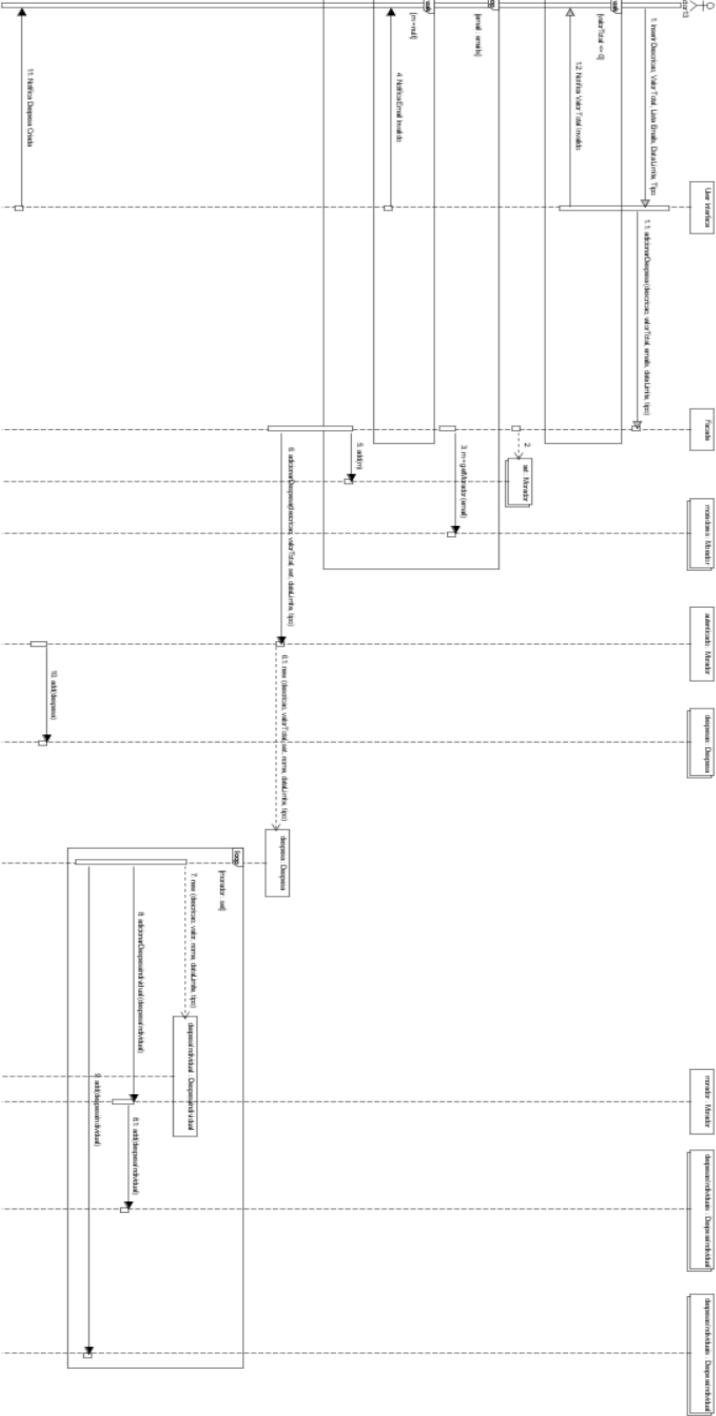
Nesta fase construímos os diagramas de classe e de sequencia (usando as classes) em simultâneo, partindo dos diagramas de sequencia de subsistema construídos anteriormente.

De seguida, apresenta-se o diagrama de classe implementado com Maps e alguns dos diagramas de sequencia construídos de acrodo com este diagrama de classe.

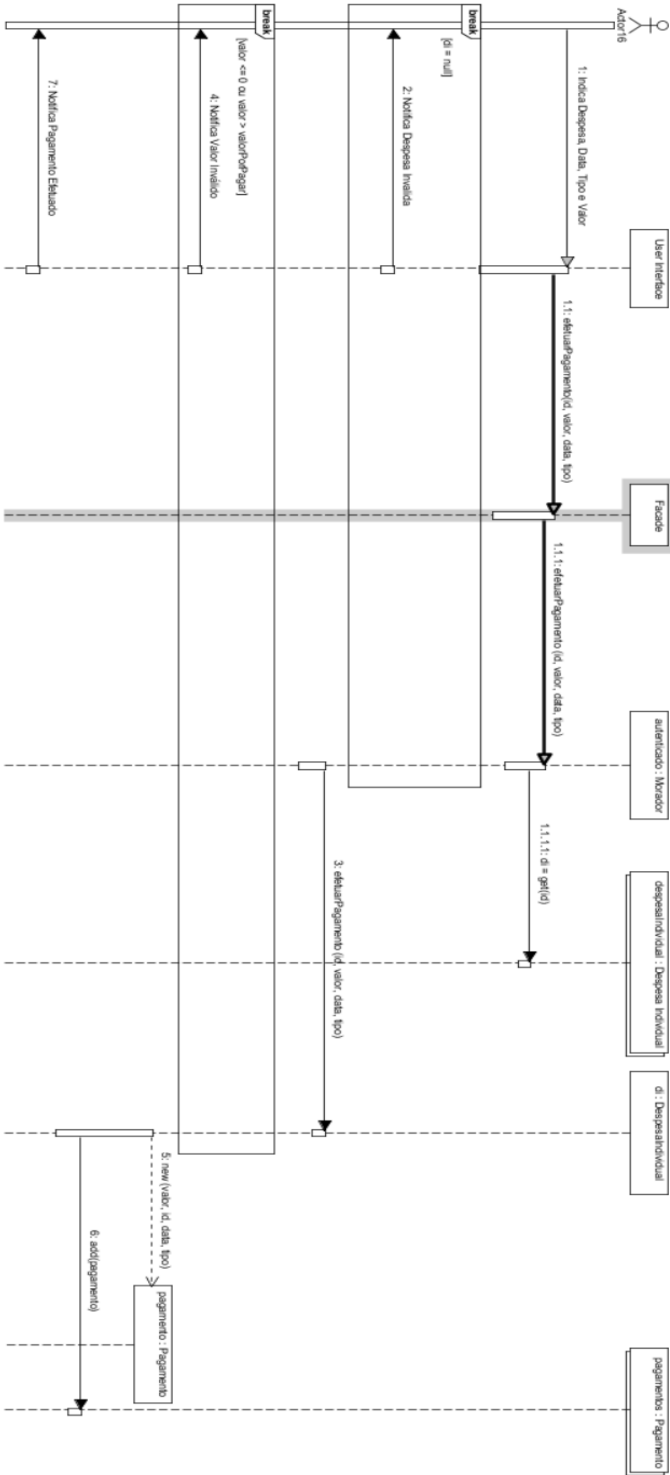


## Registrar Morador

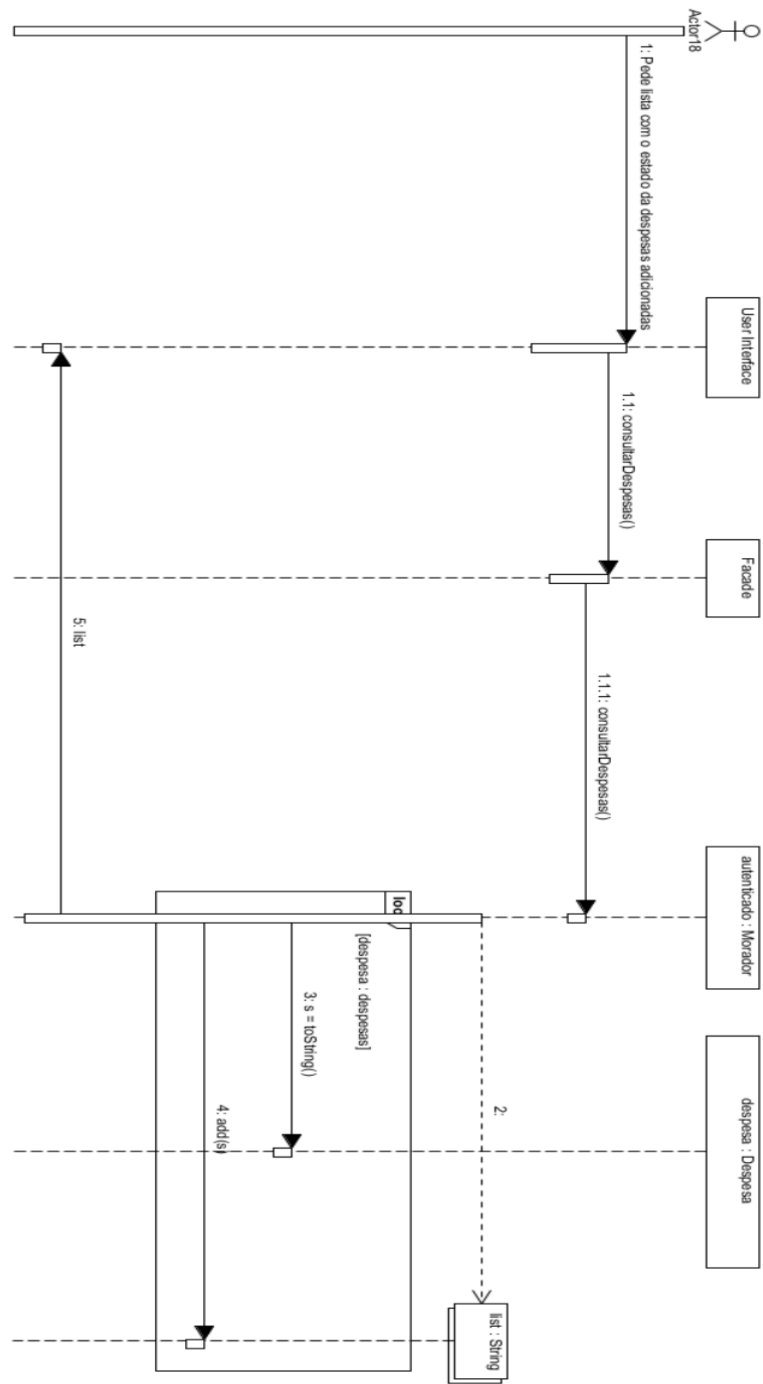




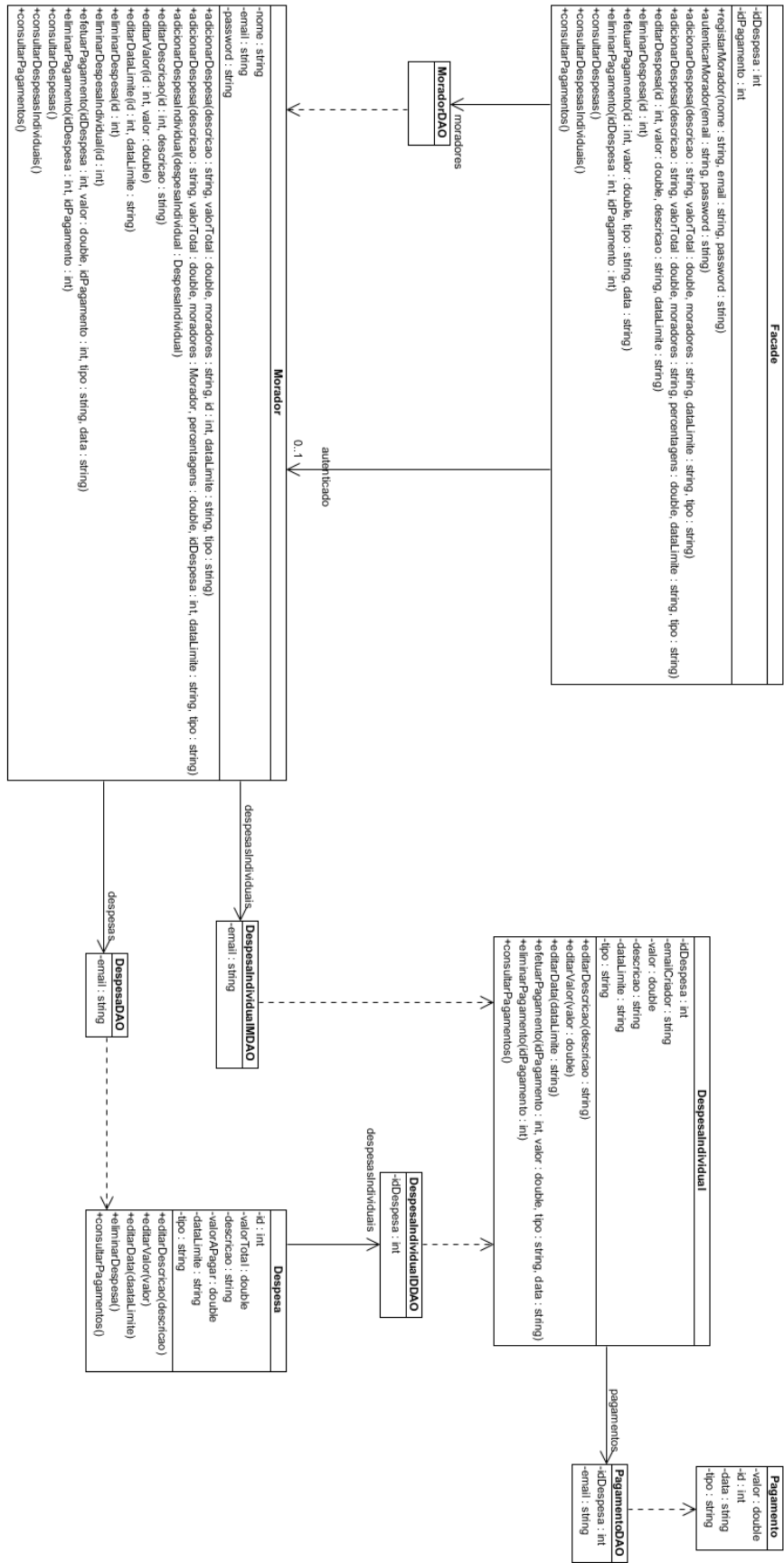
Efetuar Pagamento



Consultar Despesas



Para garantir persistência, substituímos os Maps por DAOs, obtendo o seguinte diagrama de classes:



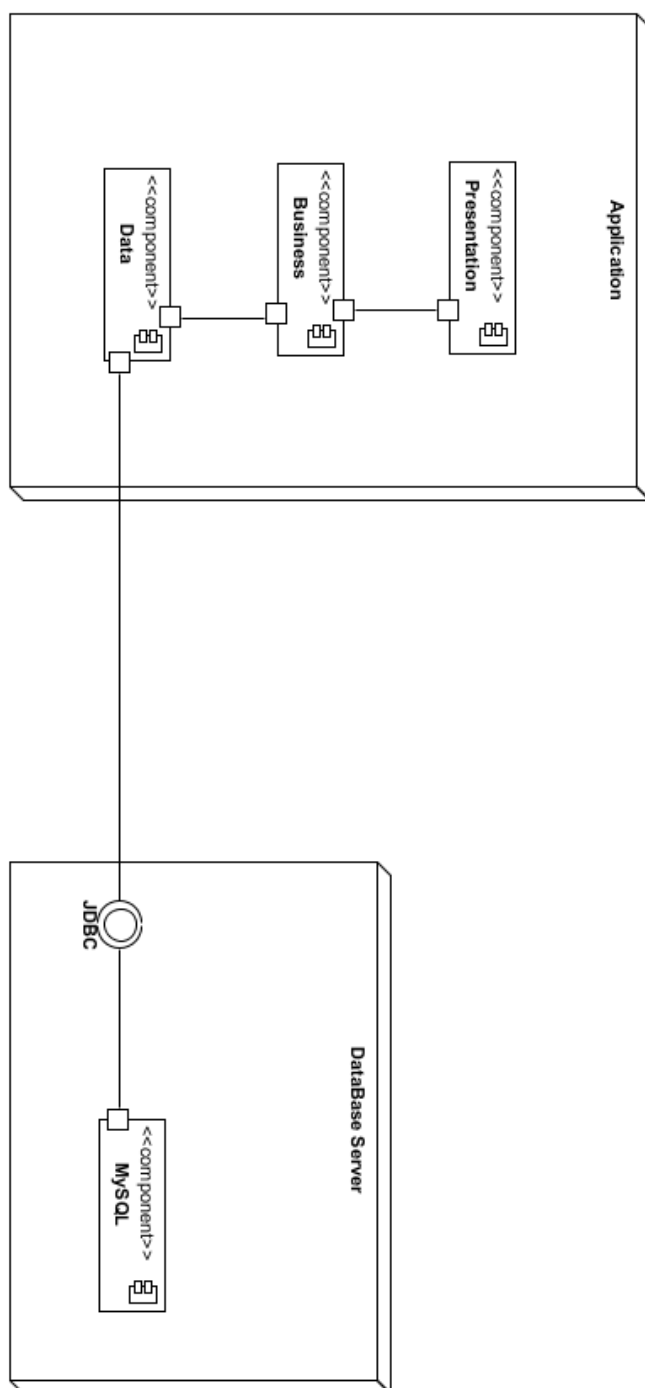
Alguns dos DAOs criados tem variáveis de instancia. Isto deve-se à necessidade de distinguir os dados que se pretende obter da Base de Dados entre objetos. Por exemplo, cada morador tem um conjunto de despesas. Se o DAO não tivesse o email do morador como variável de instancia, ao invocar o método *values()* sobre esse conjunto, obtinha-se todas as despesas adicionadas por todos os moradores, quando se pretendia obter apenas as despesas adicionadas por um determinado morador.

Assim, com a variável de instancia *email* é possível, ao executar uma query sobre a base de dados, limitar as despesas que se pretende.



## Diagrama de Instalação

Antes de proceder à implementação em Java criamos o seguinte diagrama de instalação:



---

## Implementação

---

Por fim, passamos para a implementação em Java.

Começamos pela lógica de negocio. Criamos as classes e definimos todas as variáveis de instancia. De seguida implementamos os use cases de acordo com os diagramas de sequencia desenvolvidos para cada um.

Depois da logica de negocio estar desenvolvida passamos para a logica de dados. Para isso criamos as tabelas necessárias para persistir os dados em MySQL e desenvolvemos os DAOs, implementando os métodos necessários em cada um de acordo com a logica de negocio e com as tabelas criadas.

Por fim, passamos para o desenvolvimento da camada de apresentação, usando a ferramenta Swing, disponível de forma intuitiva no IDE NetBeans.

---

## Conclusão

---

Na primeira fase do trabalho pratico desenvolvemos o diagrama de domínio da nossa aplicação, bem como o diagrama de use case, subdiagramas e as suas respetivas especificações.

Na segunda fase, desenvolvemos os mokups da interface e a maquina de estado a eles associados. Criamos também os diagramas de sequencia (de sistema, de subsistema e de classes), bem como os diagramas de package, de classes e de instalação.

Também nesta fase, procedemos ao desenvolvimento da aplicação com uma interface gráfica intuitiva e uma camada de persistência de dados.

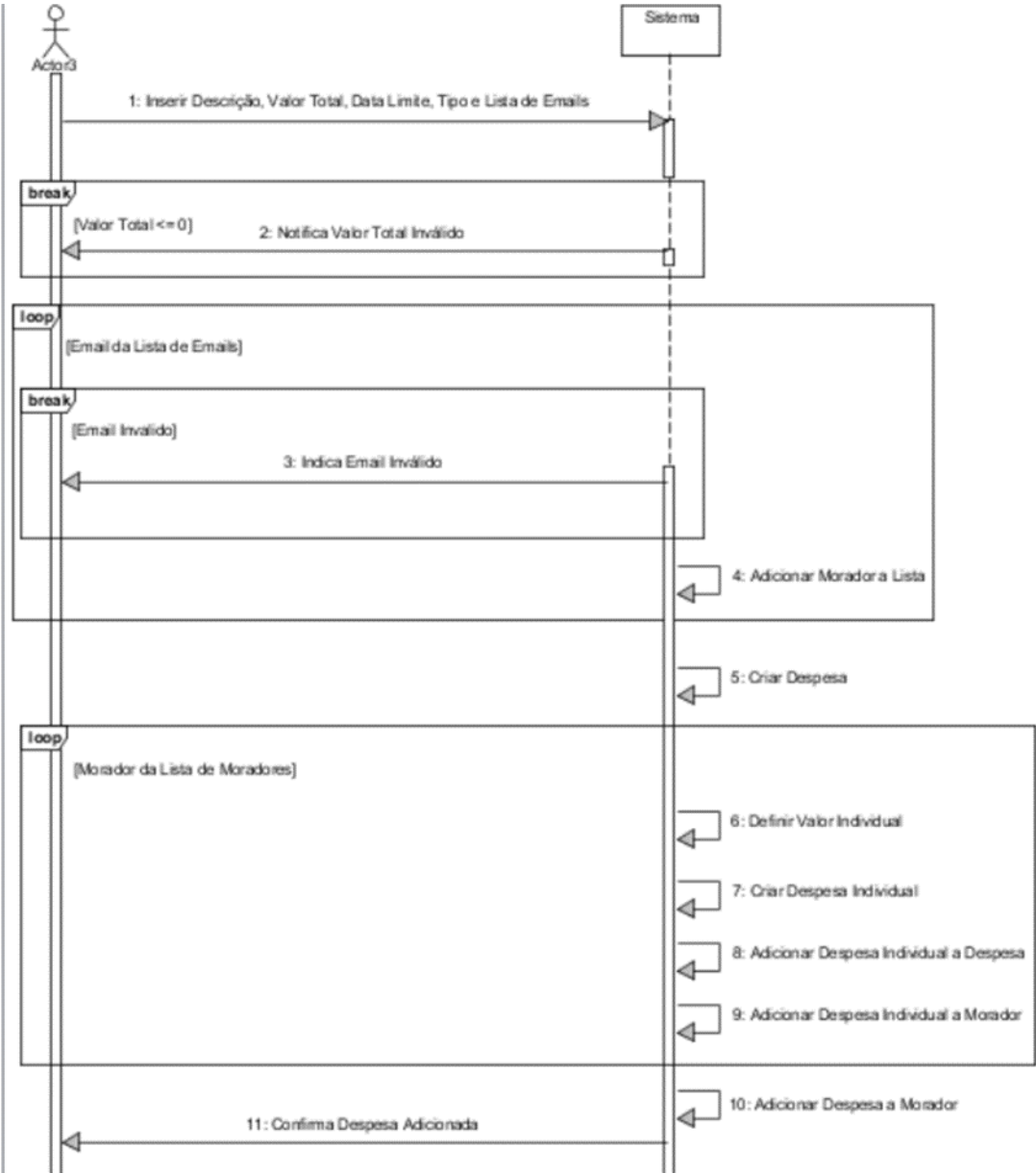
A nossa aplicação cumpre, assim, os requisitos que o grupo estipulou na primeira fase deste trabalho, sendo possível o registo e login de moradores, a possibilidade de adicionar, editar e eliminar despesas e, ainda, efetuar e eliminar pagamentos.

Como pontos fortes destacamos:

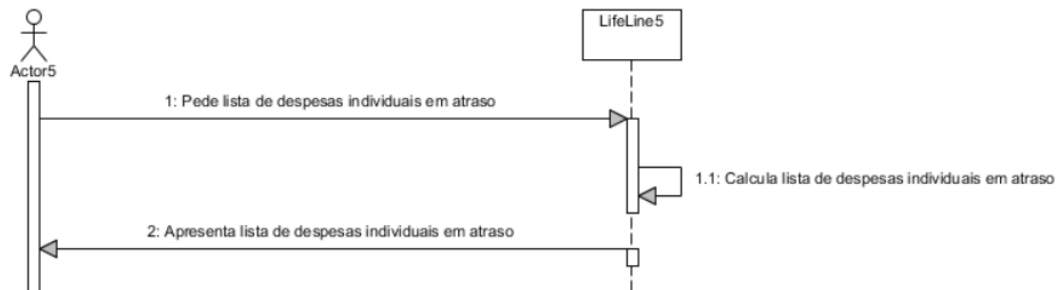
- Ao atribuirmos um método de divisão às despesas, possibilitamos que mais tarde seja possível adicionar facilmente novas maneiras de partilhar despesas. É possível editar despesas, no caso em que é cometido algum erro de introdução.
- O nosso sistema de registo/login, possibilita ainda que possam ser partilhadas despesas com outros utilizadores registados, fazendo com que seja possível a pessoas que não moram na mesma residência partilhar despesas. Para além disso, garante a privacidade e segurança dos dados, uma vez que apenas os utilizadores que introduziram tanto as despesas como os pagamentos os conseguem editar/cancelar.

Anexos

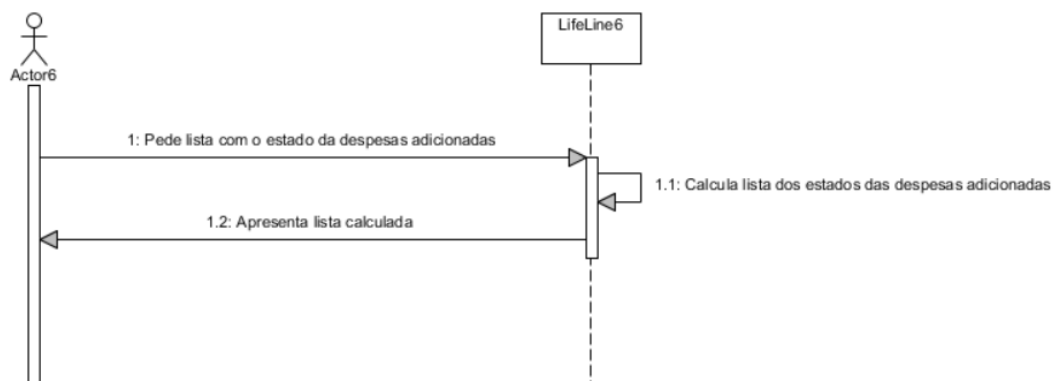
Adicionar despesa percentual



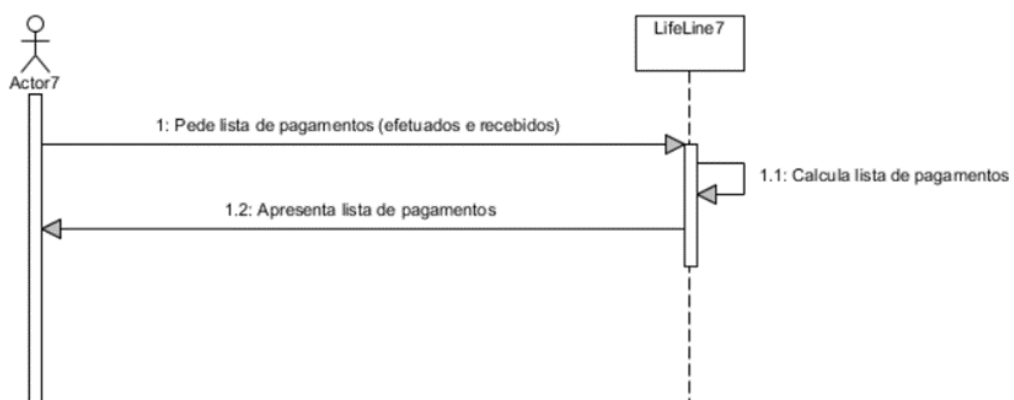
### Consultar despesas individuais em atraso



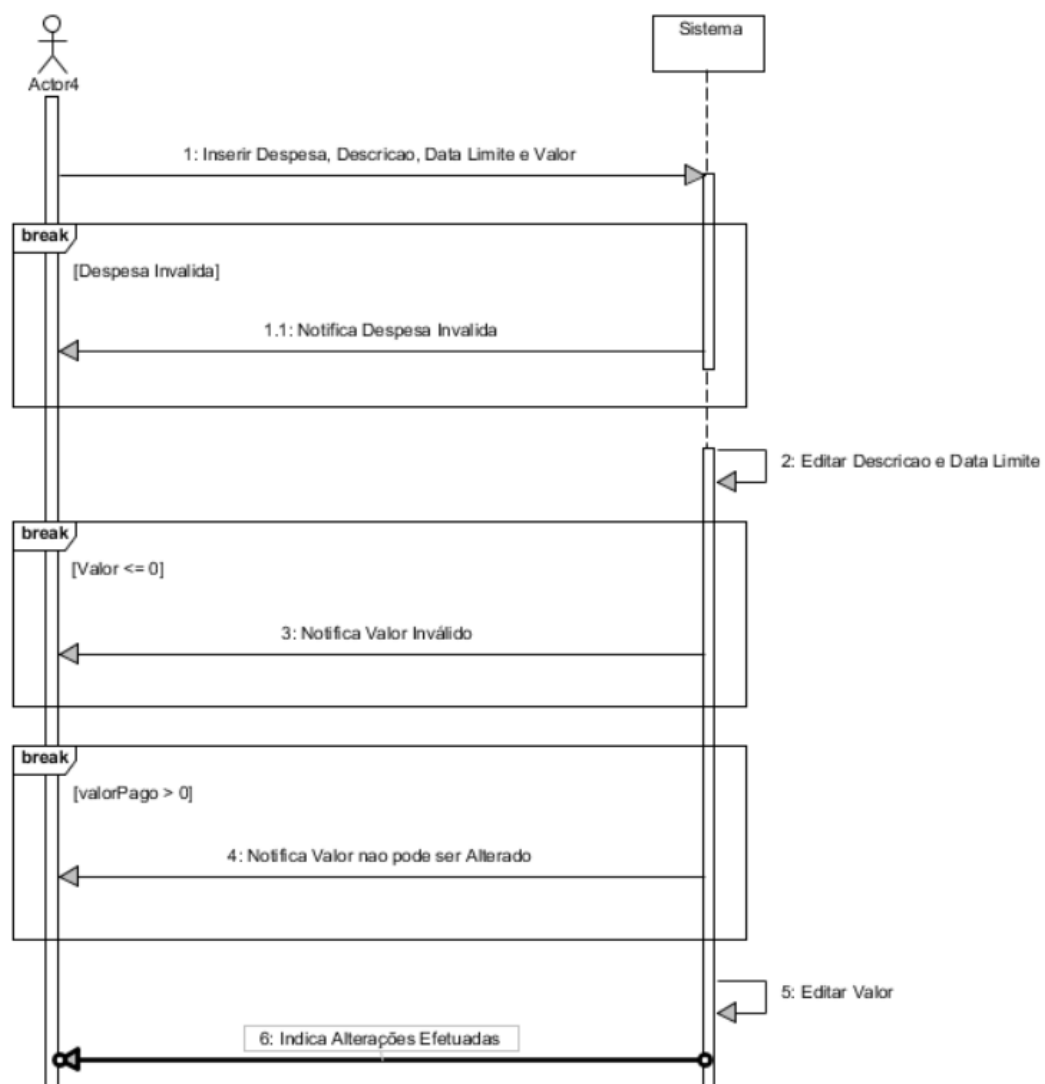
### Consultar estado das despesas adicionadas



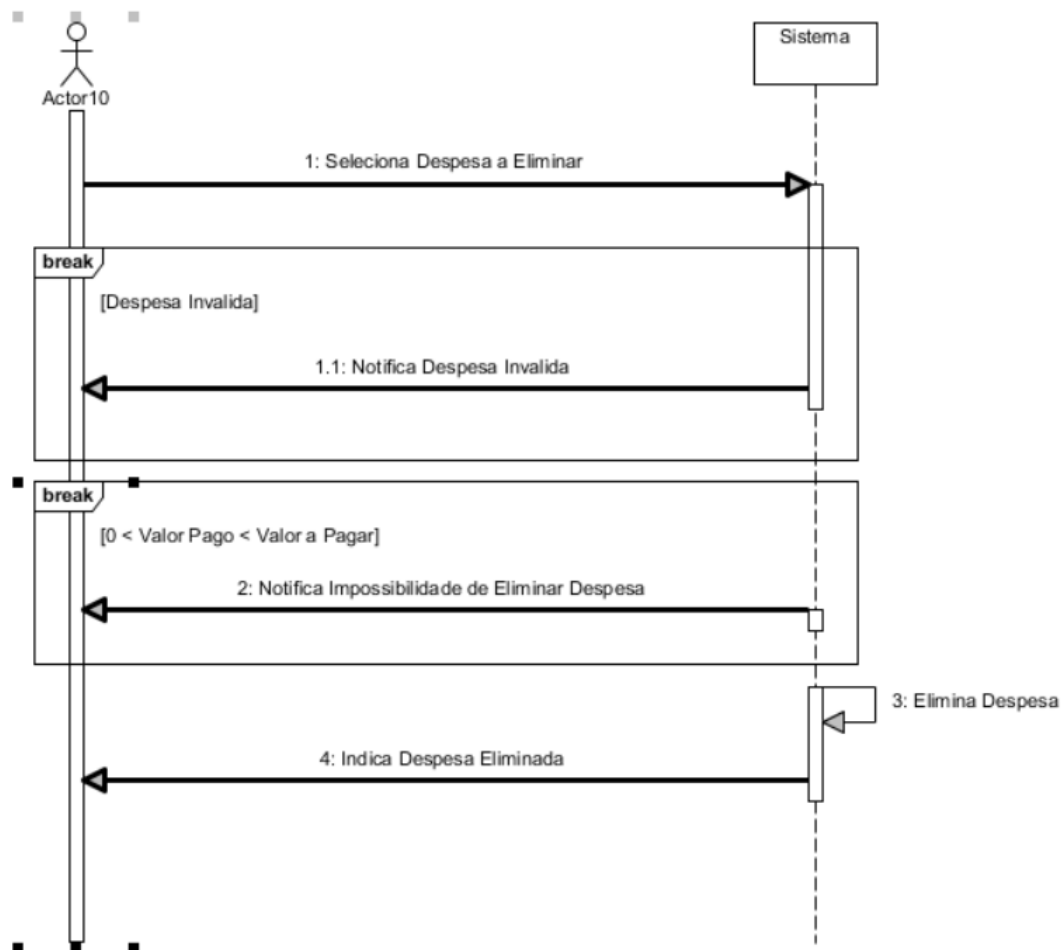
### Consultar pagamentos



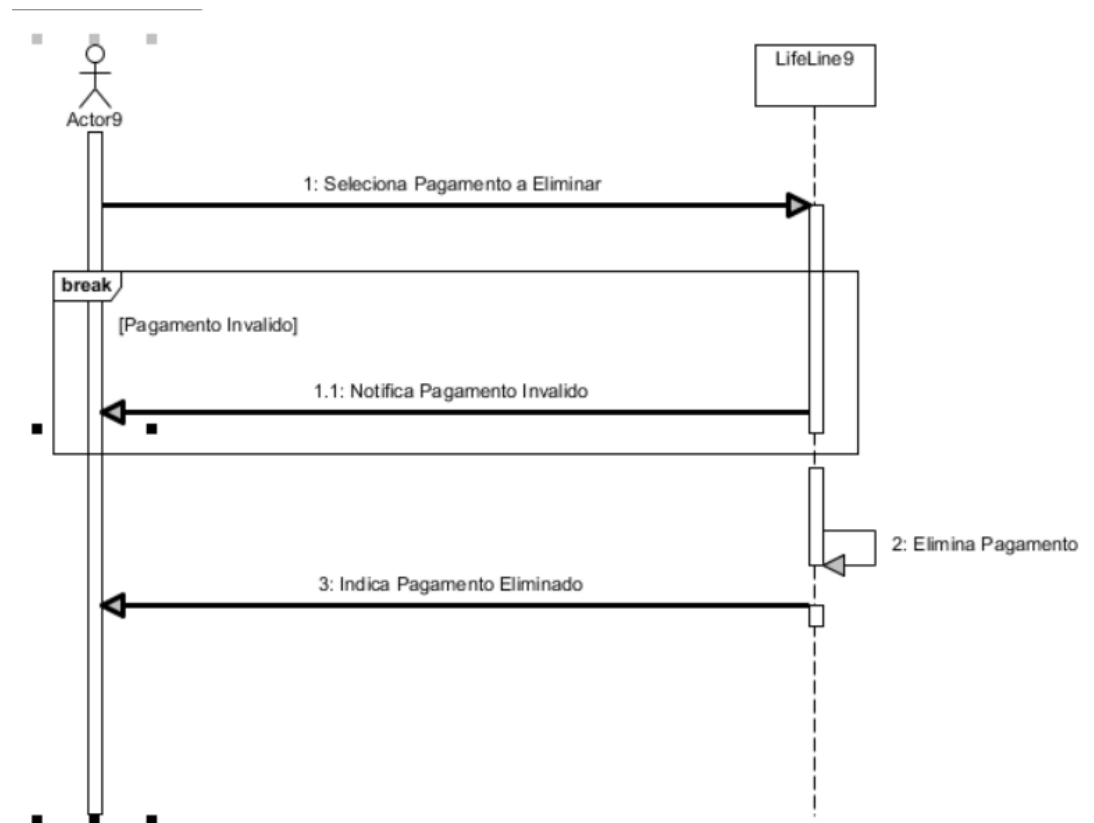
## Editar Despesa



## Eliminar despesa

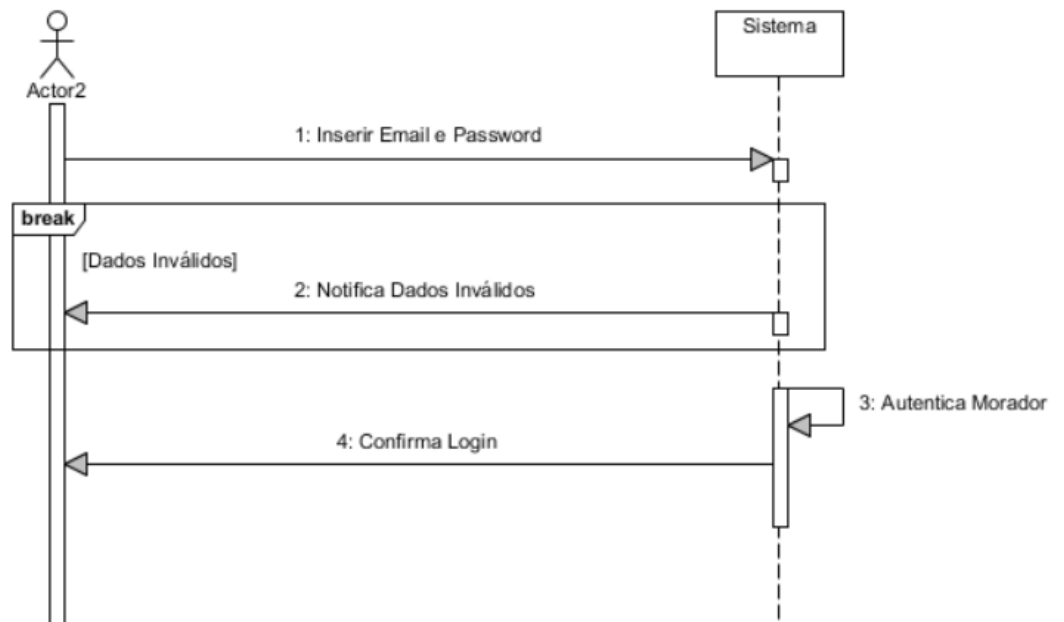


## Eliminar pagamento

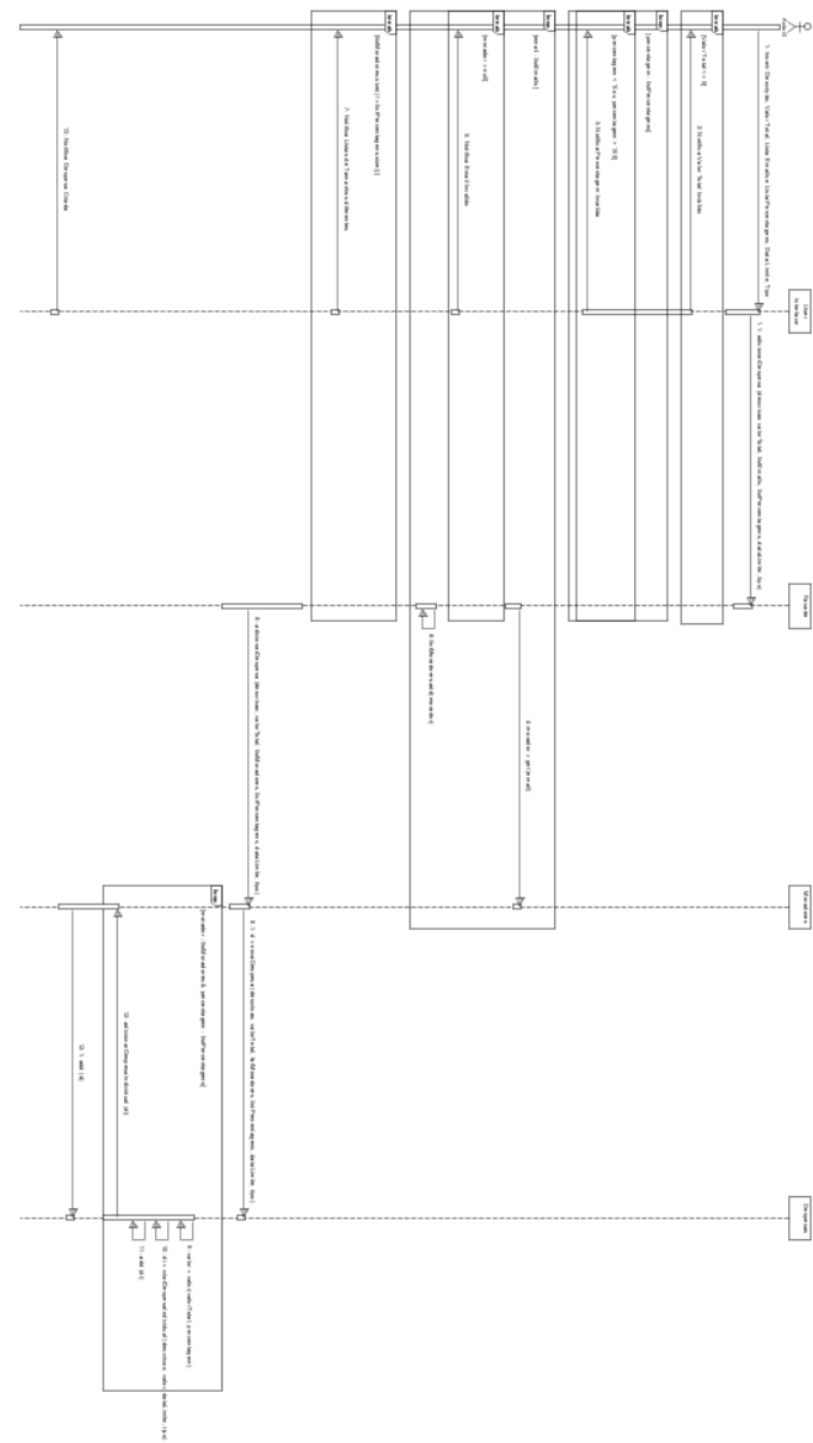




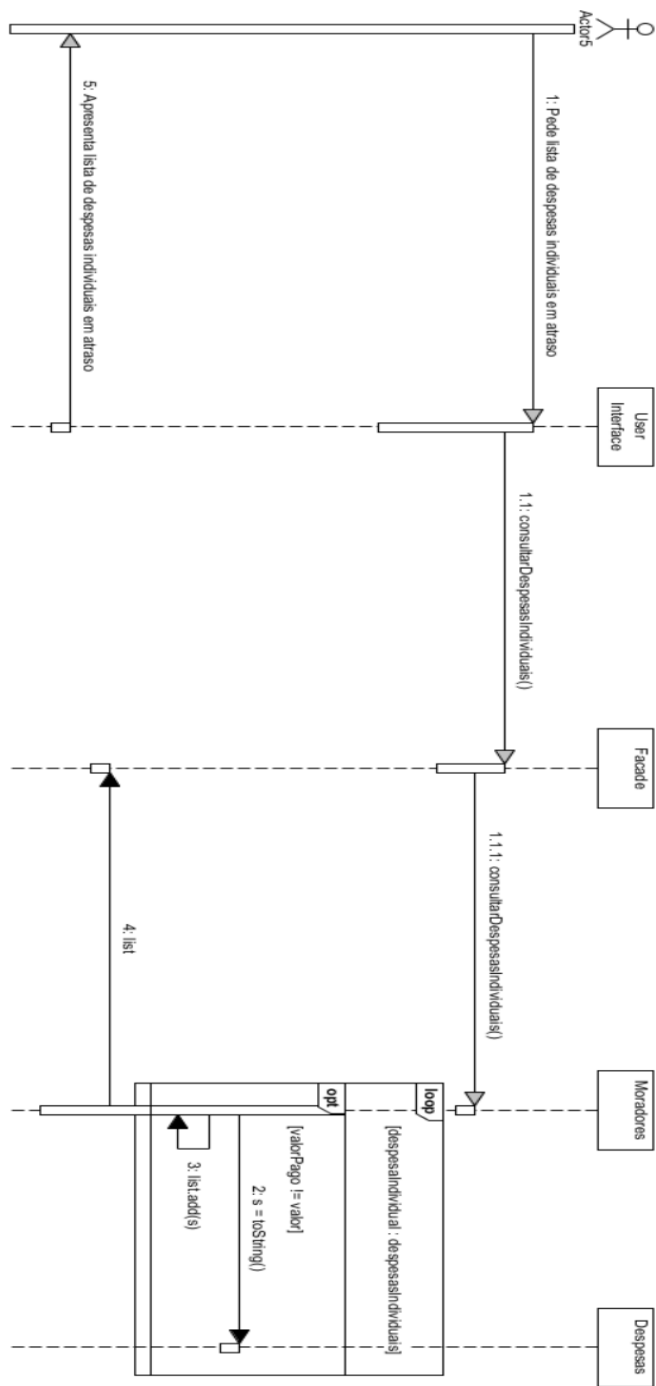
## Login



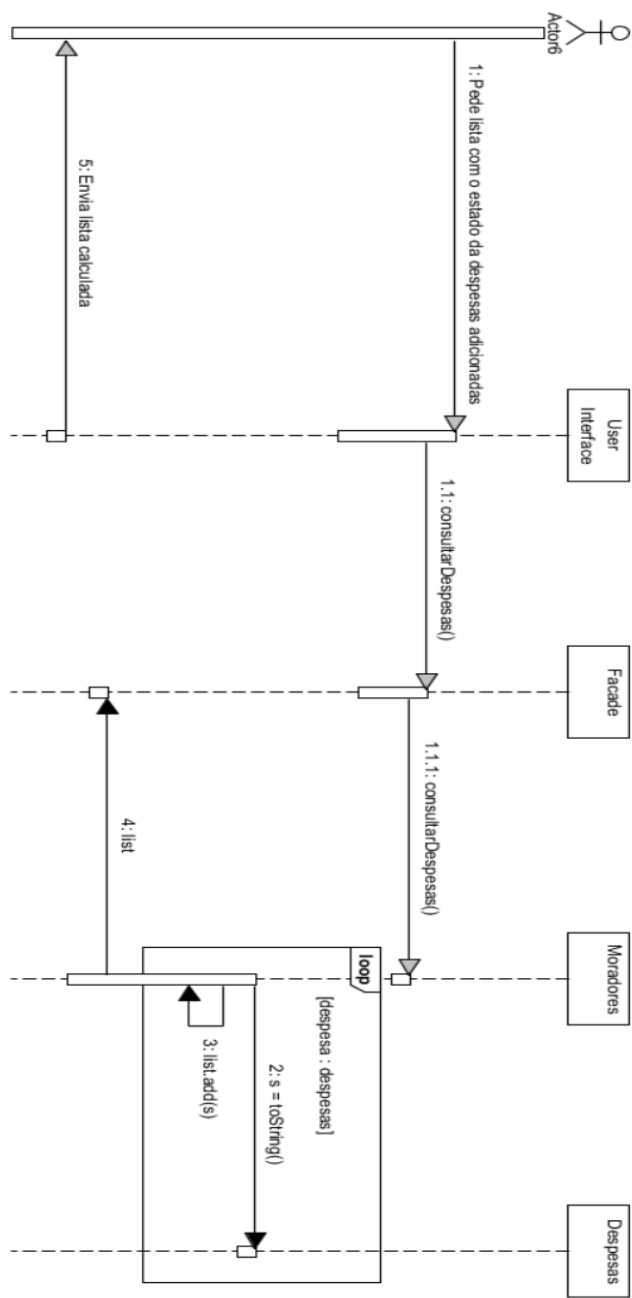
Adicionar despesa – Percentual (Subsistemas)



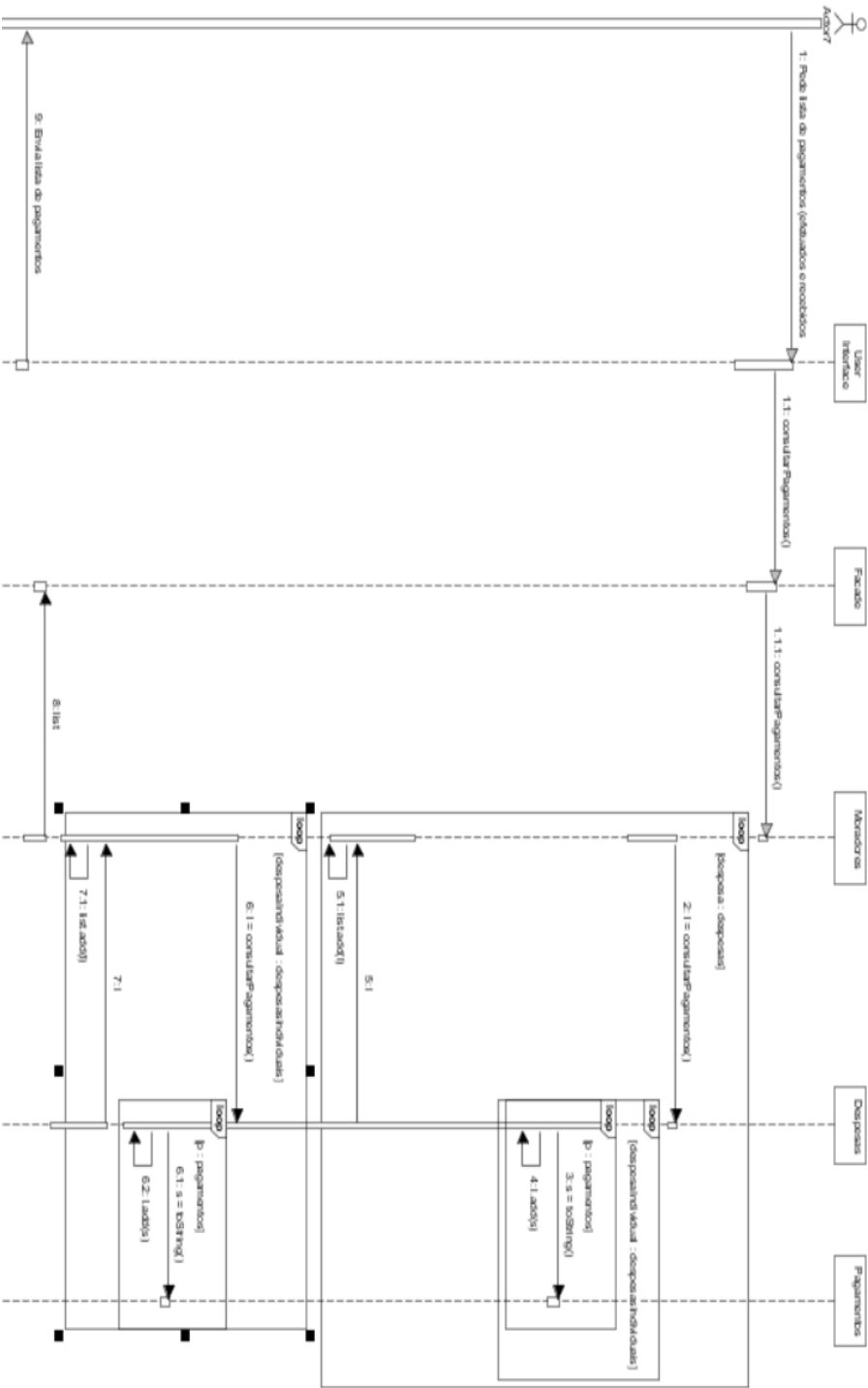
Consultar despesas individuais (Subsistemas)

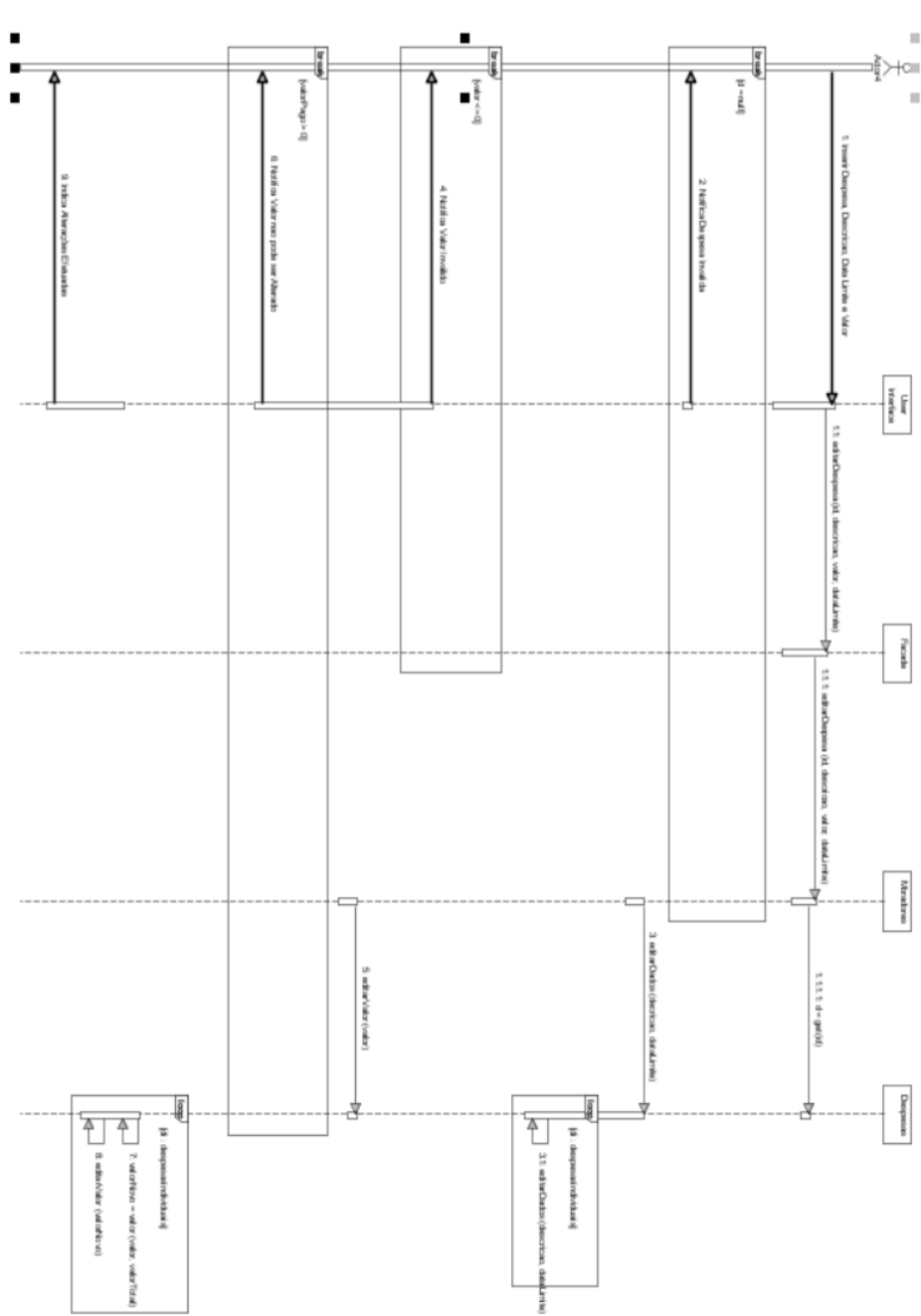


Consultar estado das despesas adicionadas (Subsistemas)

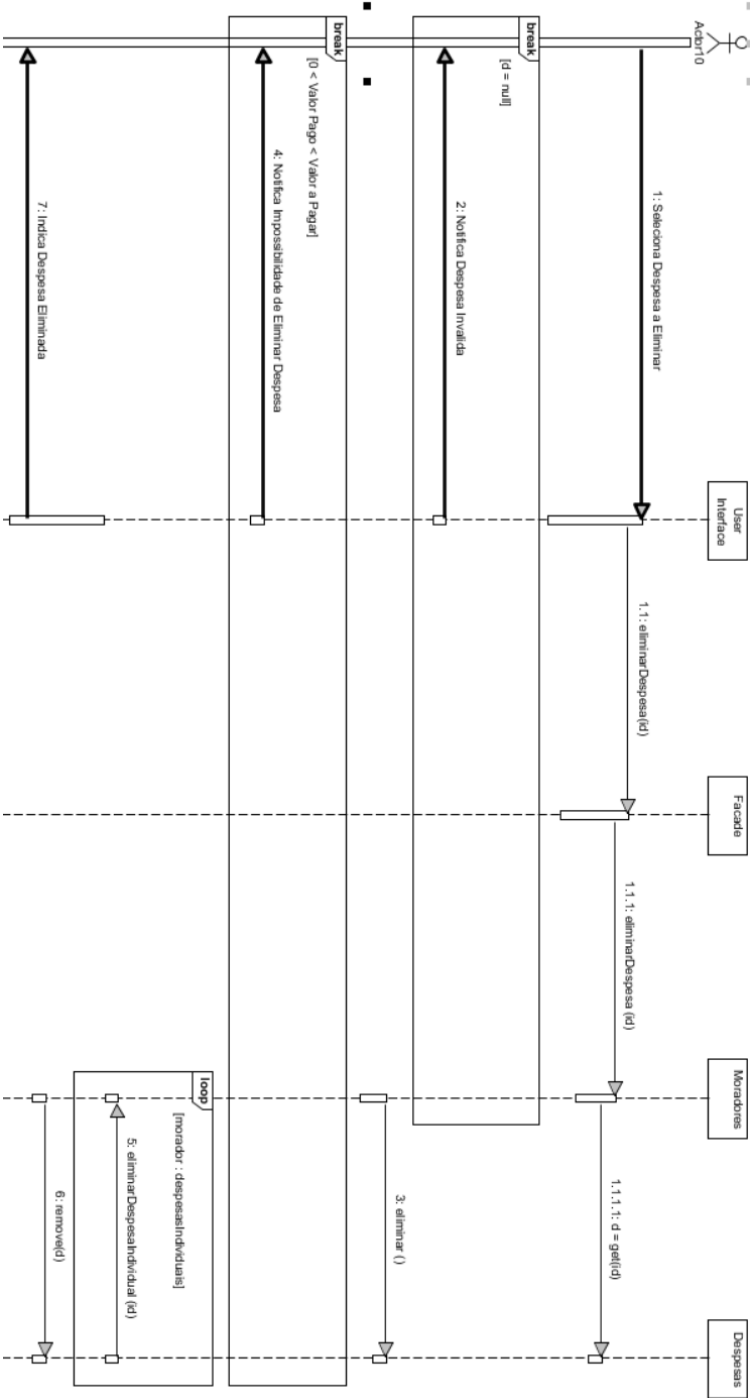


Consultar pagamentos (Subsistemas)

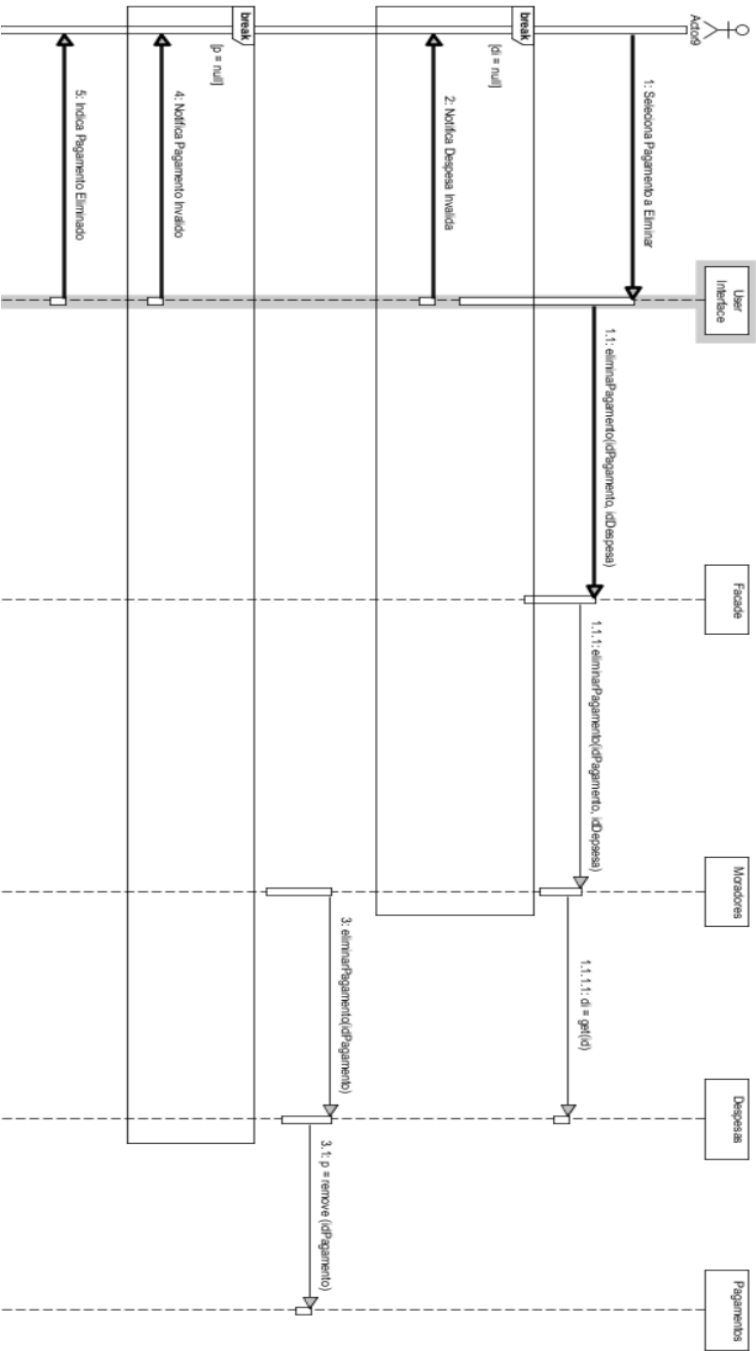




Eliminar despesa (Subsistemas)

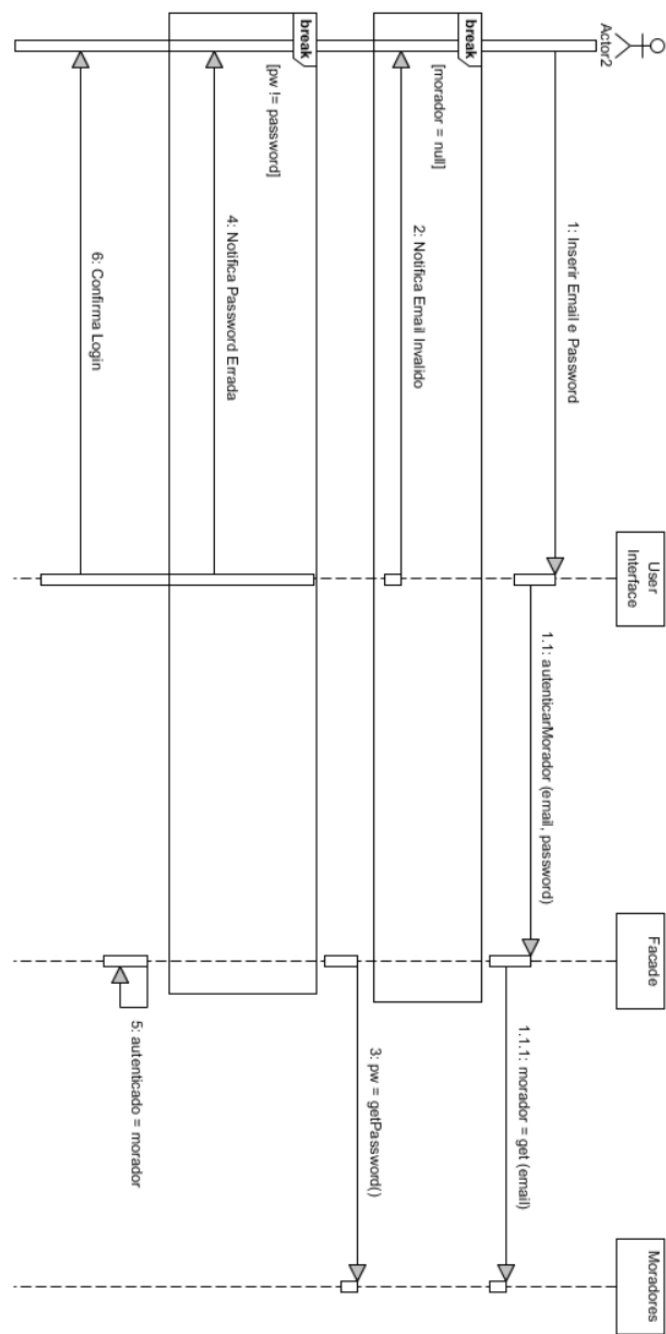


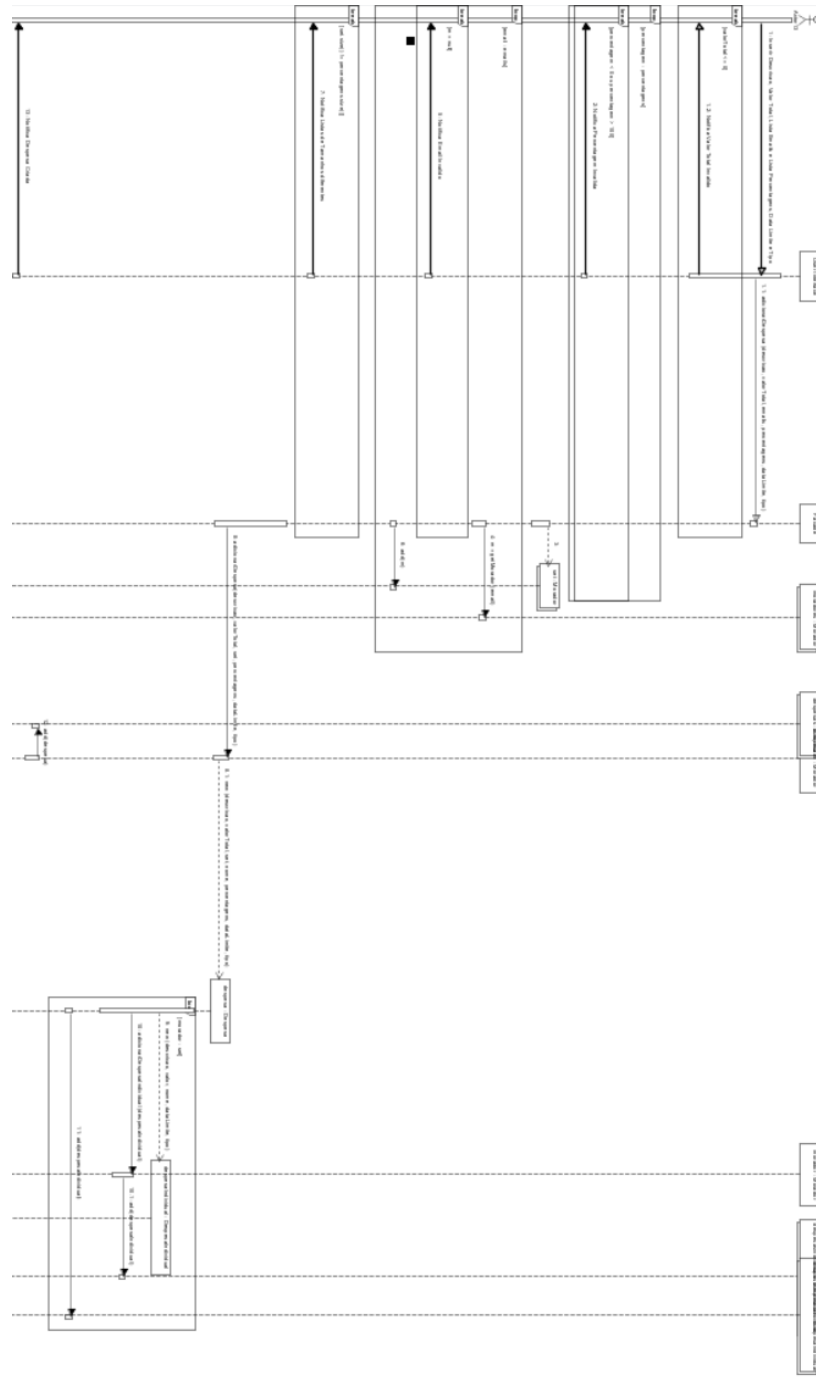
Eliminar pagamento (Subsistemas)



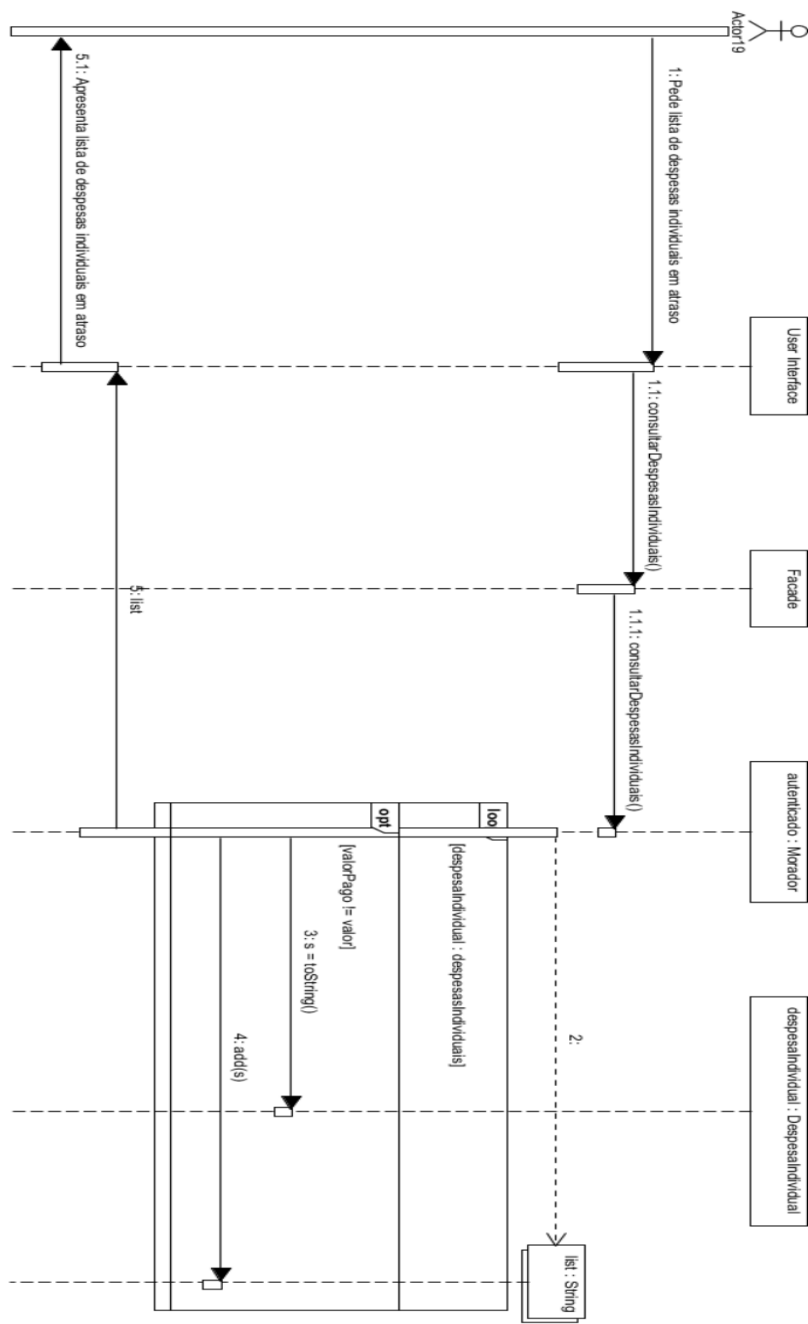


Login (Subsistemas)

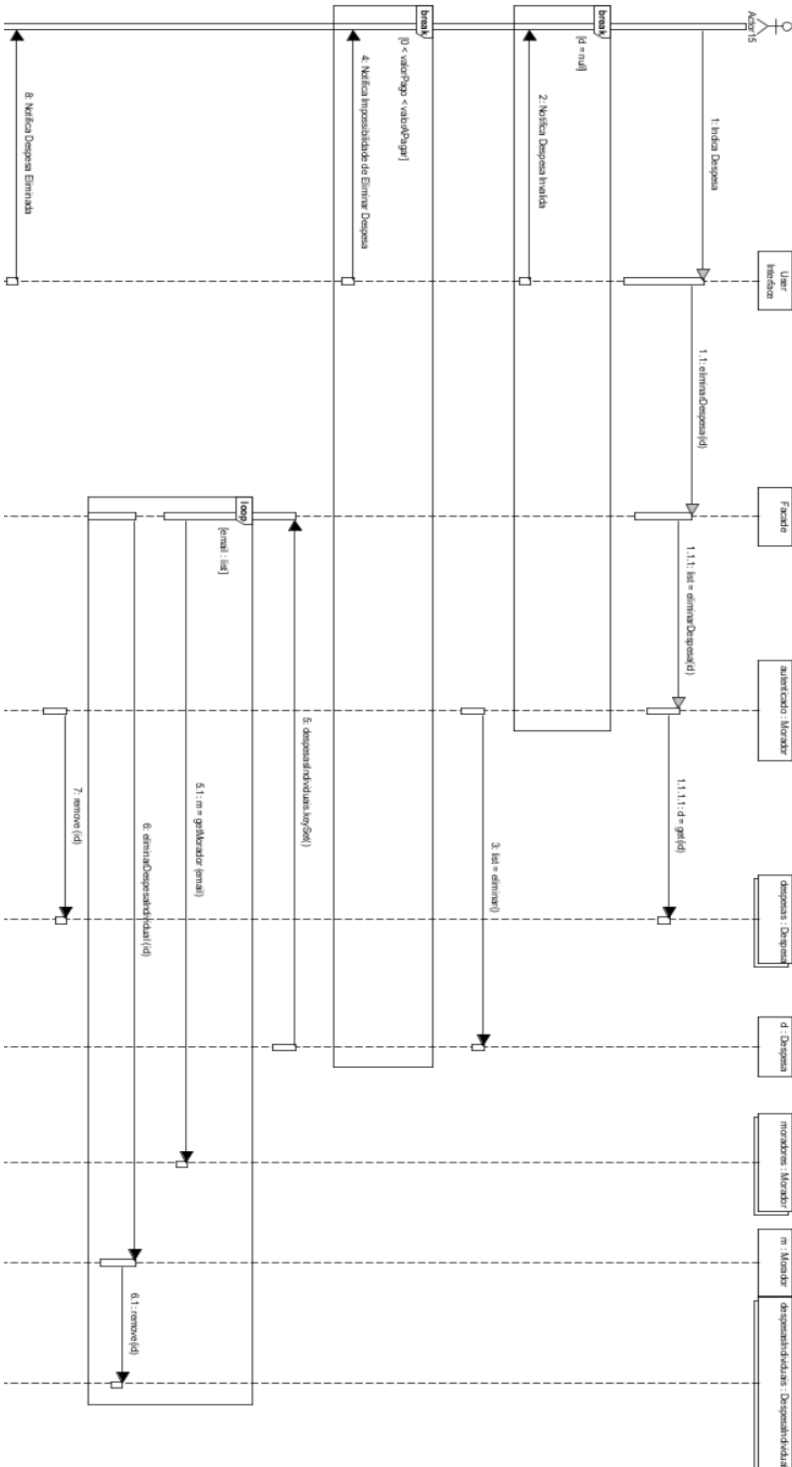


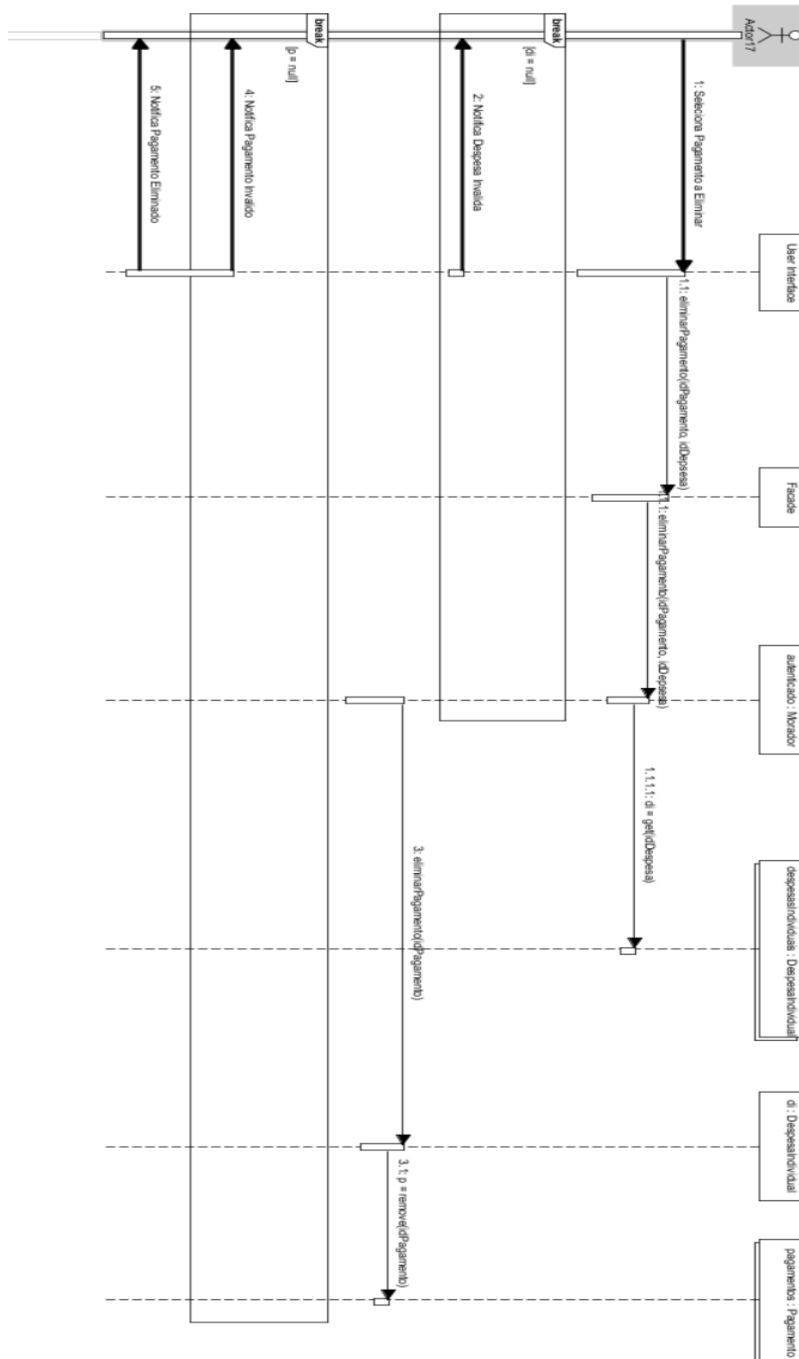


Consultar despesas individuais (Classes)









Login (Classes)

