



**Gestão de Trajes**

**Engenharia de Software II**

**Engenharia Informática**

**Trabalho realizado por:**

André Pedro Nº1011418

# Índice

Descrição do Tema .....	3
Diagrama de Contexto.....	4
Resumo QuittingTime.....	5
SpiralDevelopment.....	6
TwoTierReview .....	7
Tabela de Atores e respetivos casos de uso .....	8
Casos de Uso .....	9
Descrição dos Casos de Uso .....	10
Diagramas de Sequência .....	18
Diagrama de Estados.....	22
Diagrama de Classes.....	23
Atividades e Tempos Gastos.....	24
Dicionário de dados.....	25
Diagrama de Componentes.....	25
Diagrama de Pacotes.....	25
Diagrama de Atividade .....	25
Diagrama de Instalação .....	25
Protótipo .....	26
Conclusão .....	27

# Descrição do Tema

---

Irei criar uma aplicação “Gestão de Trajes” de gestão de artigos numa loja de trajes tendo como objectivo ajudar na organização de compras e arranjos.

Esta aplicação ira permitir a criar fichas de clientes alterar artigos, criar vendas, criar arranjos entre outras.

A aplicação será feita em c# numa Windows form.

# Diagrama de Contexto

O **DFD** (diagrama de fluxo de dados) de mais alto nível que representa todo o sistema como um único processo é conhecido como **diagrama de contexto**, e é composto por fluxos de dados que mostram as interfaces entre o sistema e as entidades externas. O diagrama é uma forma de representar o objeto do estudo, o projeto, e sua relação ao ambiente.

Um diagrama de contexto permite identificar os limites dos processos, as áreas envolvidas com o processo e os relacionamentos com outros processos e elementos externos à empresa (ex.: clientes, fornecedores) e mostra as características do sistema como podemos ver na figura seguinte o diagrama de contexto relativo à aplicação que está a ser desenvolvido.

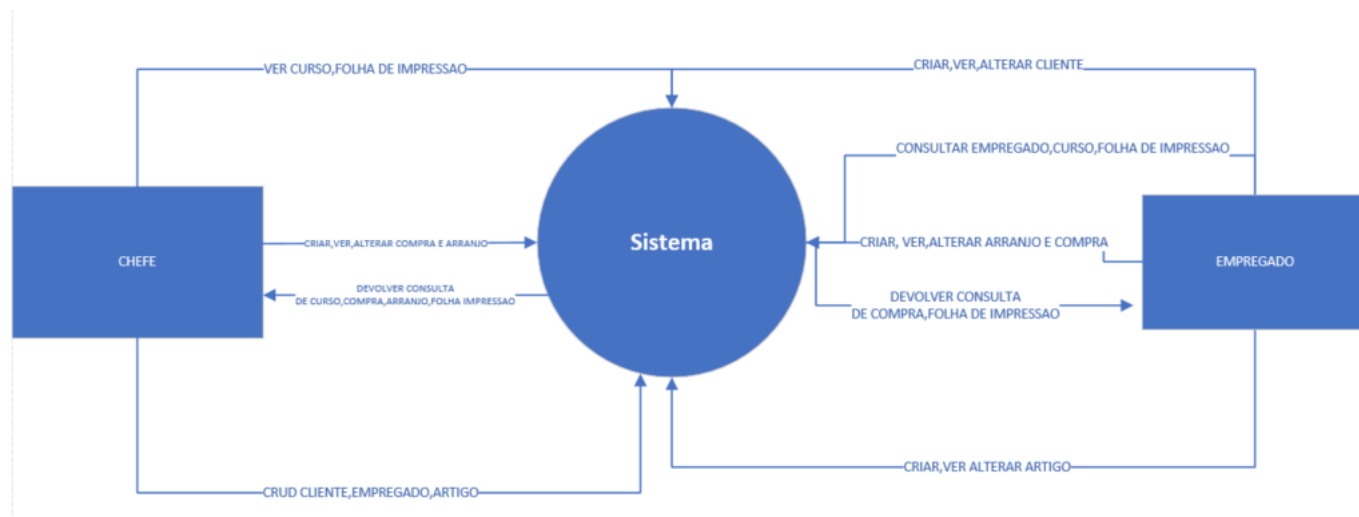


Figura 1 – Diagrama de Contexto

# Resumo QuittingTime

---

Escrever e criar casos de uso a mais do que o cliente aprova é um desperdício de tempo, recursos e ainda pode atrasar o projeto. Os requerimentos devem ser especializados para reduzir o risco de falhanço do projeto e devemos seguir esses procedimentos rigorosamente para correr tudo como o cliente quer e definiu. Contudo nem sempre este planeamento é tao linear como parece.

As pessoas, como são inexperientes, não sabem ao certo tudo o que querem. Ou seja, novas ideias estão sempre a vir ao de cima. Os casos de uso para serem bons tem de ser equilibrados e sobretudo, devem ter o essencial para o projeto funcionar, ou seja, poucos casos de uso é mau e ter muitos também é, daí ter de existir um meio-termo. Por vezes é complicado deixar de escrever casos de uso porque temos de alternar e não dificultar o processo porque por vezes temos de escolher entre uns casos e outros para não acrescentar demasiada informação, muita dela inútil. Existem casos em que um cliente, após a realização dos casos de uso pretendem mudar as coisas e adicionar ou retirar casos de uso. Isto é bastante complicado para o programador porque um projeto destes deve ser bem lineado e pensado cautelosamente. A palavra Formalidade é bastante importante na realização dos projetos para reduzir a ambiguidade de algumas situações. Estas situações ambíguas podem destruir um projeto visto que, como podem ter vários significados o cliente pode explicar e querer algo que o programador pode interpretar de maneira diferente e depois existe uma insatisfação das 2 partes. Parar de escrever casos de uso demasiado cedo é a maneira mais fácil e ingénua de adicionar ambiguidades a um projeto.

Nas empresas existem sempre pessoas com uma vasta experiencia o que ajuda a que não aconteçam este tipo de erros e ambiguidades. Requerimentos específicos e detalhados podem comprometer os requerimentos pré-propostos. No projeto um erro ao princípio pode ser muito melhor porque são mais fáceis de reparar do que for um mais tarde porque pode acontecer que tenhamos de começar tudo de novo o que é bastante mau para um projeto de um cliente e para o próprio programador. Não devemos adicionar casos de uso a mais do que aqueles que já foram aprovados pelo cliente. Estas são as 3 perguntas para que um cliente saiba se os casos de uso estão completos ou não.

- Foram identificados e documentados todos os objetivos e todos os atores?
- O cliente ou alguém que o represente reconheceu e aprovou todos os casos de uso antes o desenvolvimento do projeto?
- O designer pode implementar os casos de uso?

Se a resposta a estas perguntas for não ainda há trabalho para fazer. Caso seja sim então deve se avançar com o projeto.

Exemplo: Wings Over The World.

Ahmed defende que os casos de uso não devem ser completos ate estar tudo bem detalhado e discutido.

Ralph defende que não se deve perder tempo e avançar com o projeto sem quaisquer preocupações. Qual deles terá razão?

“Quando é que se sabe quando os casos de uso estão completos?” Esta é uma pergunta bastante difícil nesta área. O QuittingTime equilibra o risco do atraso do projeto com o risco da falta de requerimentos.

Estes fatores são bastante comuns em qualquer empresa. Deve se ter sempre cuidado no que se faz e implementa. A comunicação entre cliente e programador deve ser bastante cautelosa para que não haja erros e para que não haja falsas interpretações.

# SpiralDevelopment

---

Desenvolver casos de uso num único passo é difícil e pode tornar difícil a tarefa de incorporar novas informações neles, pode até atrasar a descoberta de fatores de risco.

Escrever casos de uso interactivamente fornece uma perspetiva de vista em que facilmente podemos recuar e rescrever ou riscar algo se descobrirmos que esta errado ou que não esta a funcionar corretamente. Podemos perder algum trabalho, mas será muito menor do que teríamos perdido se tivéssemos escrito os casos de uso todos de uma vez. Mais importante podemos identificar e confrontar potenciais problemas mais cedo usando um método de aproximação iterativo.

Pode levar um longo tempo para perceber o comportamento de um sistema, e quanto mais complicado for, mais tempo ira demorar para se compreender. Atrasos custam recursos, e os requerimentos iniciais são fundamentais para o sucesso do produto, mas isto e só uma parte do projeto. Muitas pessoas podem ter deadlines que só são possíveis de cumprir se os casos de uso fornecidos forem bons.

Os requerimentos provavelmente irão mudar durante a sua análise, visto que os requerimentos são voláteis, e facilmente alterados quando sabemos mais deles. Um requerimento que parece concreto no início e necessário pode ficar inútil e desnecessário numa análise mais profunda. Todo o processo acaba por estabilizar, mas só depois de muita análise e reconstrução.

O custo de erros iniciais é alto, sendo que no inicio ate pode custar muito pouco reparar certos erros, mas a medida que o processo vai evoluindo o custo do erro e dificultando a sua deteção.

# TwoTierReview

---

Muitas pessoas têm a necessidade de rever os casos de uso, mas isto é dispendioso, leva tempo.

As revisões são necessárias para validar, verificar e avaliar a sua plenitude na parte inicial do processo de desenvolvimento.

É difícil para os developers encontrarem os seus próprios erros pois fazem inferências que podem não estar lá e estão demasiados familiarizados com o trabalho. As pessoas que leem um documento pela primeira vez estão mais propícias a encontrarem erros porque estes não os deixam entender o propósito do documento.

Os stakeholders têm um interesse nos casos de uso, assim como cada grupo de dev's têm em determinados casos de uso pois dependem destes para os ajudar a fazer o trabalho deles. Por isso e no interesse de todos que estes sejam consultados o mais cedo possível e regularmente para que os casos de uso estejam o mais corretos possível e não sejam precisas alterações no futuro.

É demasiado caro, esgotante e lento envolver todos os elementos de um projeto nestas revisões e também é ineficiente pois elas vão acabar por "atropelar" umas as outras por isso a SmallWritingTeam é mais eficiente, mas como a equipa é pequena nem todos os interesses dos stakeholders são incorporados. É difícil se não impossível para uma equipa pequena representar a visão de todos os elementos do projeto. Este tipo de revisões deve ser usado com sensatez marcando-as estritamente quando necessário.

Assim sendo o twotierreview tem dois tipos de revisões o primeiro com uma equipa pequena interna e fá-las diversas vezes. A segunda pelo grupo completo pelo menos uma vez. Primeiro deve rever-se os casos de uso internamente para verifica se são fáceis de se entender, implementar e a sua precisão. Esta revisão pode ser informal, formal ou uma combinação das duas. Podem ser precisas varias revisões e vários para as diversas áreas assim cada stakeholder pode avaliar em profundidade os casos de uso com o seu ponto de vista. Por fim nas revisões internas as equipas assertaram que é QuittingTime e que esta tudo em ordem.

## Tabela de Atores e respetivos casos de uso

Atores	O que faz?
Empregado	Criar Cliente
Empregado	Alterar Cliente
Empregado	Consultar Cliente
Empregado	Consultar Empregado
Empregado	Criar Artigo
Empregado	Alterar Artigo
Empregado	Consultar Artigo
Empregado	Criar Compra
Empregado	Alterar Compra
Empregado	Consultar Compras
Empregado	Criar Arranjo
Empregado	Alterar Arranjo
Empregado	Consultar Arranjo
Empregado	Consultar Escolas
Empregado	Consultar Folha de Impressao
Chefe	Criar Cliente
Chefe	Alterar Cliente
Chefe	Consultar Cliente
Chefe	Eliminar Cliente
Chefe	Criar Empregado
Chefe	Consultar Empregado
Chefe	Alterar Empregado
Chefe	Eliminar Empregado
Chefe	Criar Artigo
Chefe	Alterar Artigo
Chefe	Consultar Artigo
Chefe	Eliminar Artigo
Chefe	Criar Compra
Chefe	Alterar Compra
Chefe	Consultar Compras
Chefe	Criar Arranjo
Chefe	Alterar Arranjo
Chefe	Consultar Arranjo
Chefe	Consultar Escolas
Chefe	Consultar Folha de Impressao

Figura 2 - Tabela de Atores e Respetivos casos de uso



# Casos de Uso

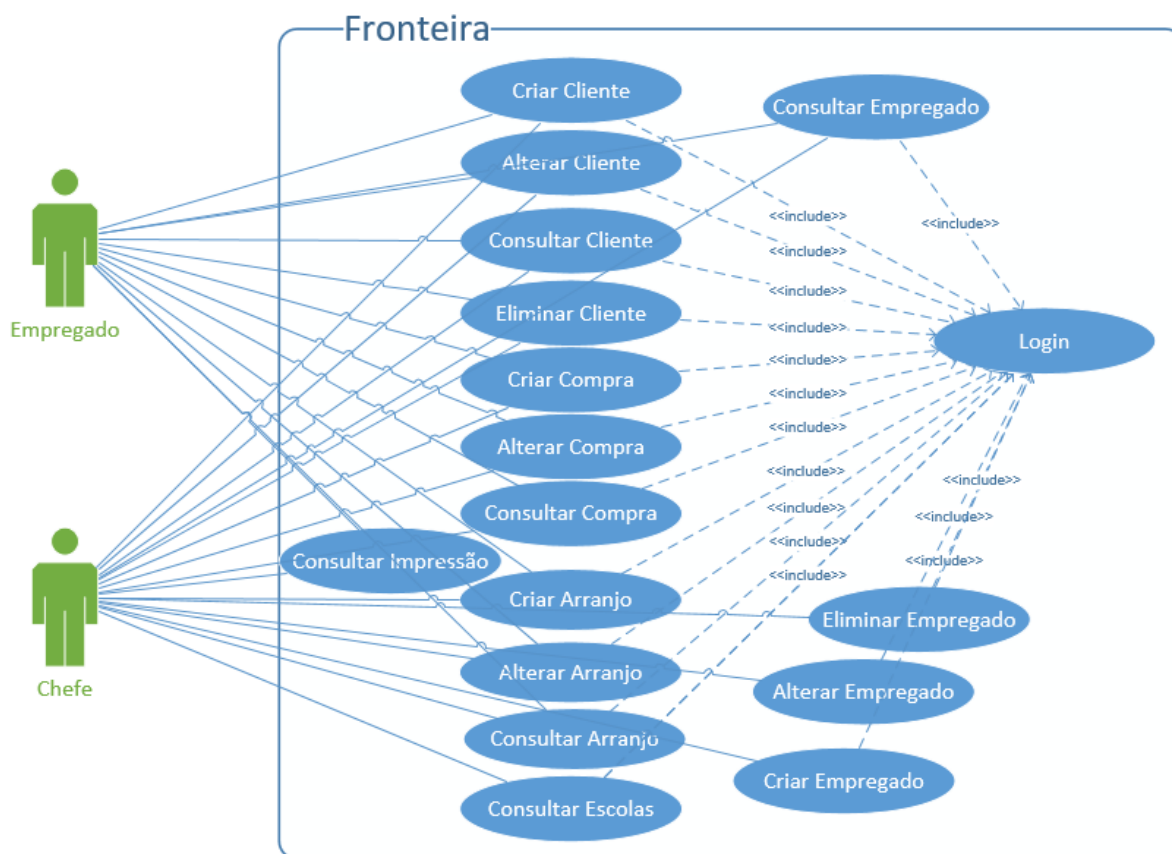


Figura 3 - Diagrama de Casos de Uso

## Descrição dos Casos de Uso

Caso de Uso	
nome	Criar Cliente
descrição	O empregado vai criar um novo cliente
pré-condição	
caminho principal	
	1 clicar na aba criar novo cliente
	2 o sistema devolve um formulário com dados para preencher sobre o cliente
caminhos alternativos	
suplementos e adornos	
	1.1 texto do botão
	1.2 cores
	1.3 verificar se o botão faz o indicado
	2.1 verificar se o sistema devolve o formulário
	3.1 verificar se o ator consegue escrever no formulário
	4.1 verificar se o sistema guarda os dados do formulário
pós-condição	nenhum

Caso de Uso	
nome	Consultar Cliente
descrição	O empregado vai consultar os dados de uma cliente
pré-condição	
caminho principal	
	1 clicar na aba consultar cliente
	2 o sistema devolve uma caixa de texto para inserir o nome do cliente
	3 o sistema devolve um formulario com os dados do cliente
caminhos alternativos	
suplementos e adornos	
	1.1 texto do botão
	1.2 cores
	1.3 verificar se o botão faz o indicado
	2.1 verificar se o sistema devolve o formulario
	3.1 verificar se o ator consegue escrever no formulario
	4.1 verificar se o sistema guarda os dados do formulario
pós-condição	nenhum

Caso de Uso	
nome	Editar Cliente
descrição	O empregado vai alterar cliente
pré-condição	tem que existir dados do cliente
caminho principal	
	1 clicar na aba alterar cliente
	2 o sistema devolve uma caixa de texto para inserir o nome do cliente
	3 o sistema devolve um formulario com os dados do cliente para alterar
	4 O ator faz as alterações e guarda
caminhos alternativos	
suplementos e adornos	
	1.1 texto do botão
	1.2 cores
	1.3 verificar se a aba faz o indicado
	2.1 verificar se o sistema devolve o formulario
	3.1 verificar se o ator consegue escrever no formulario
	4.1 verificar se o sistema guarda as alterações no formulario
pós-condição	nenhum

Caso de Uso	
nome	Eliminar Cliente
descrição	O empregado vai apagar cliente
pré-condição	tem que existir dados do cliente
caminho principal	1 clicar na aba apagar cliente 2 o sistema devolve uma caixa de texto para inserir o nome do cliente 3 o sistema devolve um formulário com os dados do cliente com a opção para eliminar 4 O ator confirma que quer eliminar
caminhos alternativos	
suplementos e adornos	1.1 texto do botão 1.2 cores 1.3 verificar se a aba faz o indicado 2.1 verificar se o sistema devolve o formulário 3.1 verificar se o ator consegue escrever no formulário 4.1 verificar se o sistema guarda as alterações
pós-condição	nenhum

Caso de Uso	
nome	Visualizar Empregado
descrição	O ator vai visualizar os empregados
pré-condição	tem que existir dados do empregado
caminho principal	1 clicar na aba Controlo 2 Selecionar a de empregados 3 o sistema devolve um formulário com os dados dos empregados
caminhos alternativos	
suplementos e adornos	1.1 texto do botão 1.2 cores 1.3 verificar se a aba faz o indicado 2.1 verificar se o sistema devolve o formulário
pós-condição	nenhum

Caso de Uso	
nome	Alterar Empregado
descrição	O Chefe vai alterar um empregado
pré-condição	
caminho principal	
	1 clicar na aba Controlo
	2 Selecionar a de empregados
	3 o sistema devolve um formulario com os dados dos empregados
	4 o ator altera os dados que pretende e clica no confirmar
caminhos alternativos	
suplementos e adornos	
	1.1 texto do botão
	1.2 cores
	1.3 verificar se a aba faz o indicado
	3.1 verificar se o sistema devolve o formulario
	4.1 verificar se o sistema guarda as informações
pós-condição	nenhum

Caso de Uso	
nome	Criar Empregado
descrição	O Chefe vai criar um empregado
pré-condição	
caminho principal	
	1 clicar na aba Controlo
	2 Selecionar a de criar empregados
	3 o sistema devolve um formulario com os dados do empregado
	4 O ator preenche os dados e confirma
caminhos alternativos	
suplementos e adornos	
	1.1 texto do botão
	1.2 cores
	1.3 verificar se a aba faz o indicado
	3.1 verificar se o sistema devolve o formulario
	4.1 verificar se o sistema guarda as informações
pós-condição	nenhum

Caso de Uso	
nome	Eliminar Empregado
descrição	O Chefe vai eliminar um empregado
pré-condição	
caminho principal	
	1 clicar na aba Controlo
	2 Selecionar a de criar empregados
	3 o sistema devolve um formulario com os dados do empregado
	4 O ator confirma que quer apagar
caminhos alternativos	
suplementos e adornos	
	1.1 texto do botão
	1.2 cores
	1.3 verificar se a aba faz o indicado
	3.1 verificar se o sistema devolve o formulario
	4.1 verificar se o sistema guarda as informações
pós-condição	nenhum

Caso de Uso	
nome	Criar Compra
descrição	O Ator vai criar uma compra
pré-condição	Tem que existir um cliente criado
caminho principal	
	1 clicar na aba Compra
	2 Selecionar a de criar compra
	3 o sistema devolve um formulario com os artigos
	4 O ator seleciona os artigos e confirma
	5 O sistema guarda os artigos na compra criada
caminhos alternativos	
suplementos e adornos	
	1.1 texto do botão
	1.2 cores
	1.3 verificar se a aba faz o indicado
	3.1 verificar se o sistema devolve o formulario
	4.1 verificar se o sistema guarda as informações
pós-condição	nenhum

Caso de Uso	
nome	Alterar Compra
descrição	O Ator vai alterar uma compra
pré-condição	Tem que existir uma compra criada
caminho principal	
	1 clicar na aba Compra
	2 Selecionar a de alterar compra
	3 o sistema devolve um formulário com os artigos dessa compra
	4 O ator faz as alterações e confirma
	5 O sistema guarda as alterações
caminhos alternativos	
suplementos e adornos	
1.1	texto do botão
1.2	cores
1.3	verificar se a aba faz o indicado
3.1	verificar se o sistema devolve o formulário
4.1	verificar se o sistema guarda as informações
pós-condição	nenhum

Caso de Uso	
nome	Consultar Compra
descrição	O Ator vai consultar uma compra
pré-condição	Tem que existir uma compra criada
caminho principal	
	1 clicar na aba Compra
	2 Selecionar a de ver compra
	3 o sistema devolve um formulário com o id da compra
	4 O ator seleciona o id da compra e confirma
	5 O sistema devolve a compra com os respectivos artigos
caminhos alternativos	
suplementos e adornos	
1.1	texto do botão
1.2	cores
1.3	verificar se a aba faz o indicado
3.1	verificar se o sistema devolve o formulário
4.1	verificar se o sistema guarda as informações
pós-condição	nenhum

Caso de Uso	
nome	Criar Arranjo
descrição	O Ator vai criar um arranjo
pré-condição	Tem que existir um cliente
caminho principal	
	1 clicar na aba Arranjo
	2 Selecionar a criar Arranjo
	3 o sistema devolve um formulario para criar o arranjo
	4 O ator escreve os dados do arranjo e confirma
	5 O sistema guarda os dados do arranjo
caminhos alternativos	
suplementos e adornos	
	1.1 texto do botão
	1.2 cores
	1.3 verificar se a aba faz o indicado
	3.1 verificar se o sistema devolve o formulario
	4.1 verificar se o sistema guarda as informações
pós-condição	nenhum

Caso de Uso	
nome	Alterar Arranjo
descrição	O Ator vai alterar um arranjo
pré-condição	Tem que existir um arranjo
caminho principal	
	1 clicar na aba Arranjo
	2 Selecionar a de alterar Arranjo
	3 o sistema devolve uma caixa de texto para inserir o id do arranjo
	4 O ator escreve o id do arranjo e confirma
	5 O sistema devolve o formulário com os dados do arranjo do id selecionado
	6 O ator faz as respetivas alterações e confirma
caminhos alternativos	
suplementos e adornos	
	1.1 texto do botão
	1.2 cores
	1.3 verificar se a aba faz o indicado
	3.1 verificar se o sistema devolve o formulario
	4.1 verificar se o sistema guarda as informações
pós-condição	nenhum



Caso de Uso	
nome	Consultar Arranjo
descrição	O Ator vai consultar um arranjo
pré-condição	Tem que existir um arranjo
caminho principal	
	1 clicar na aba Arranjo
	2 Selecionar a de Consultar Arranjo
	3 o sistema devolve uma caixa de texto para inserir o id do arranjo
	4 O ator escreve o id do arranjo e confirma
	5 O sistema devolve o formulário com os dados do arranjo do id selecionado
caminhos alternativos	
suplementos e adornos	
	1.1 texto do botão
	1.2 cores
	1.3 verificar se a aba faz o indicado
	3.1 verificar se o sistema devolve o formulario
	4.1 verificar se o sistema guarda as informações
pós-condição	nenhum

# Diagramas de Sequência

Os diagramas de Sequência das ferramentas UML que representa as interações entre o utilizador e o sistema. Estes diagramas foram construídos a partir dos casos de uso:

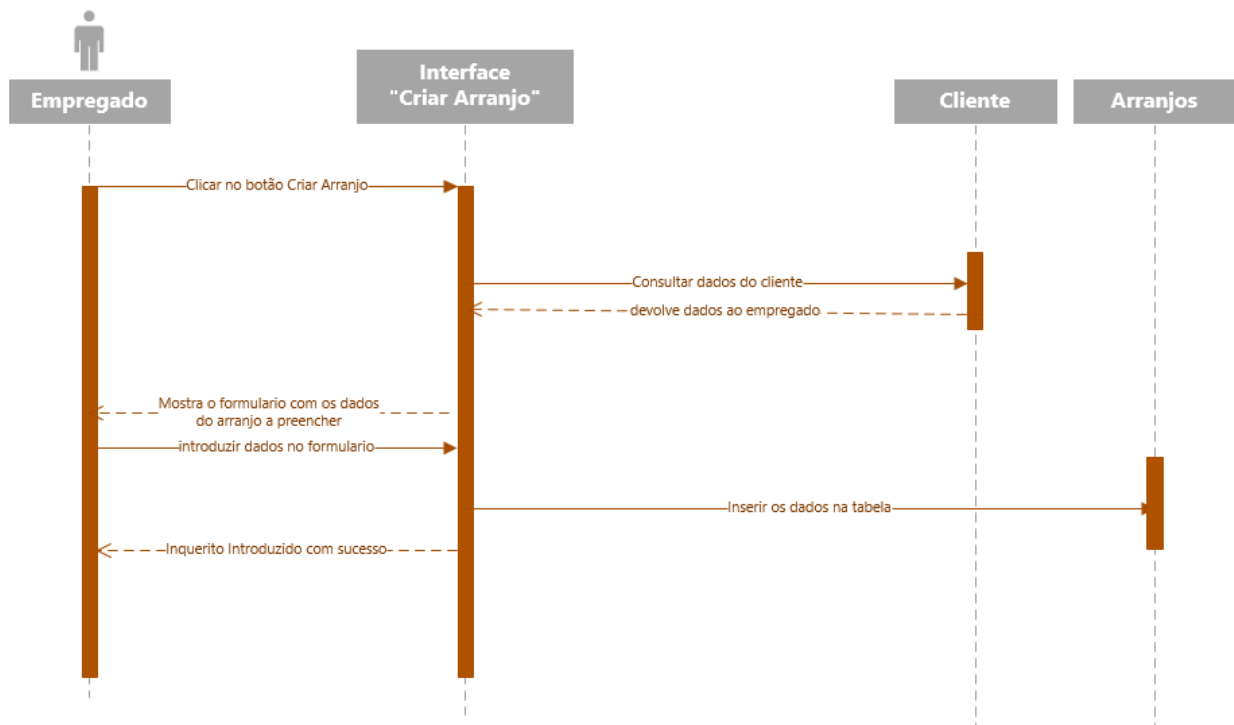


Figura 1 - Diagrama de Sequencia Criar Arranjo

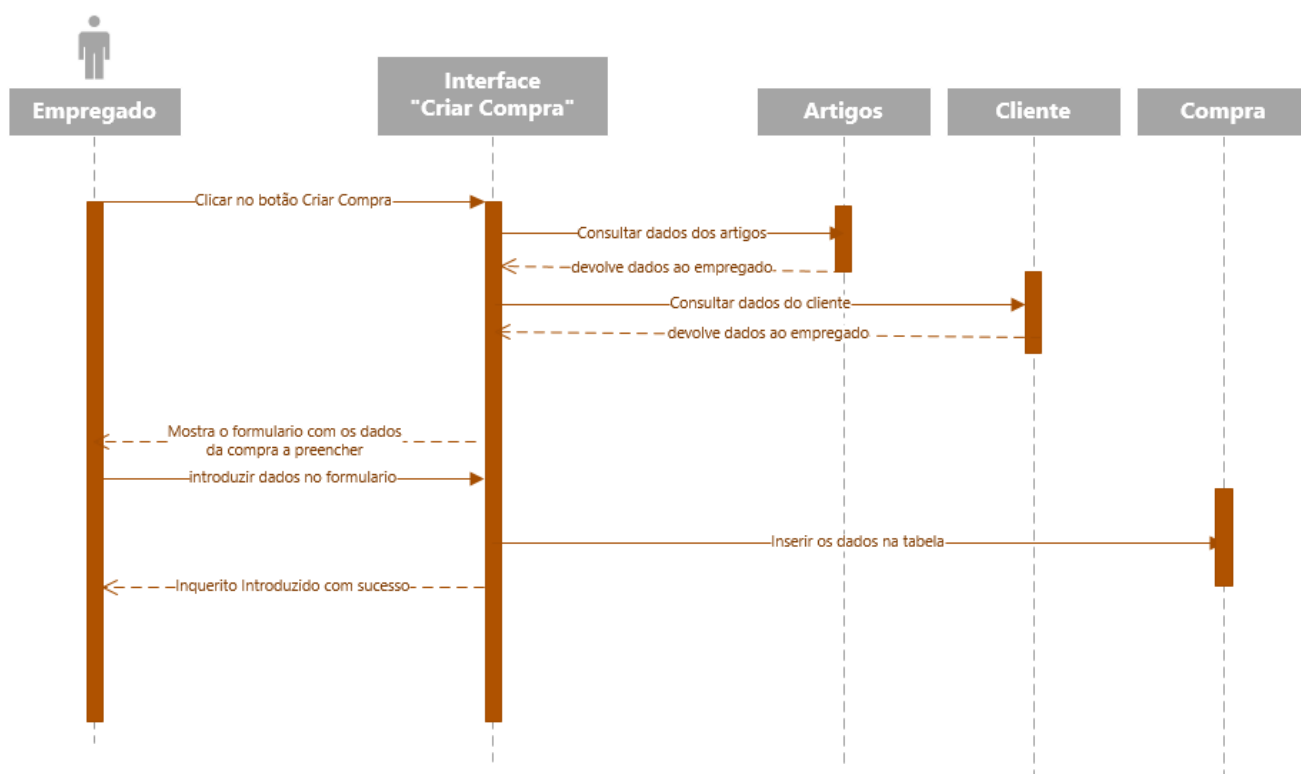


Figura 2 - Diagrama de Sequencia Criar Compra

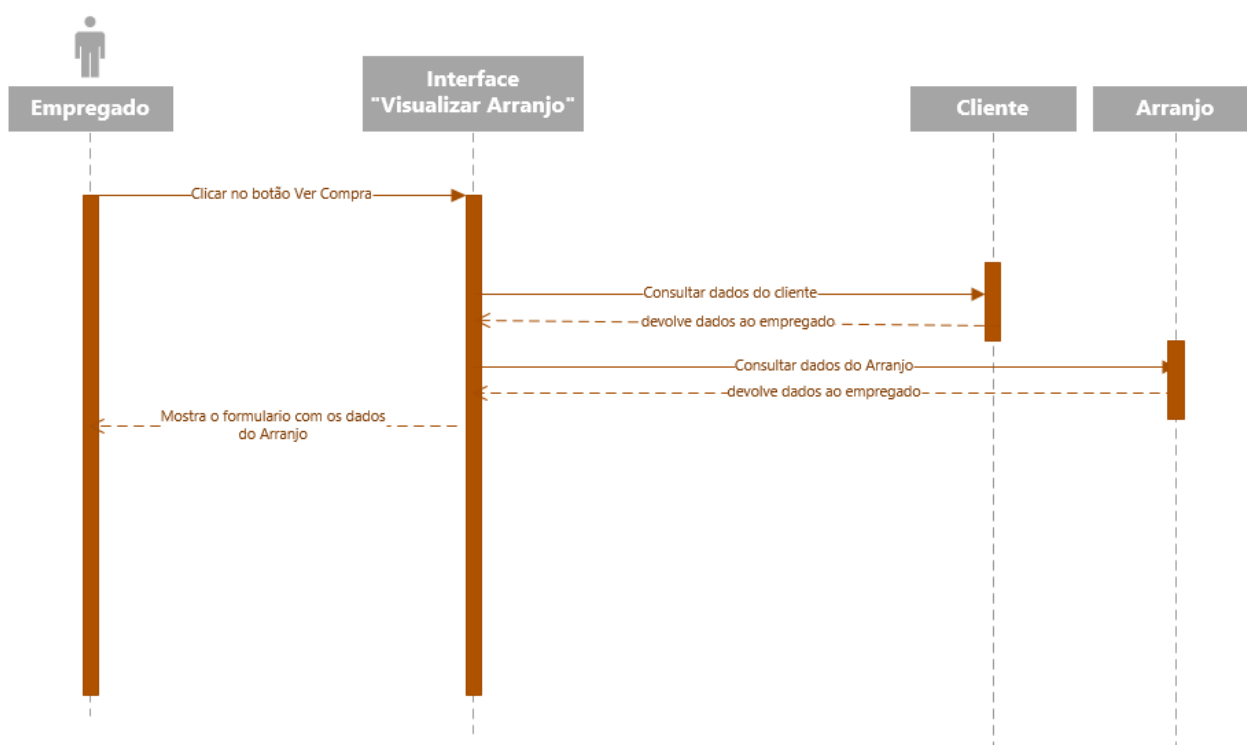


Figura 3 - Diagrama de Sequencia Visualizar Arranjo

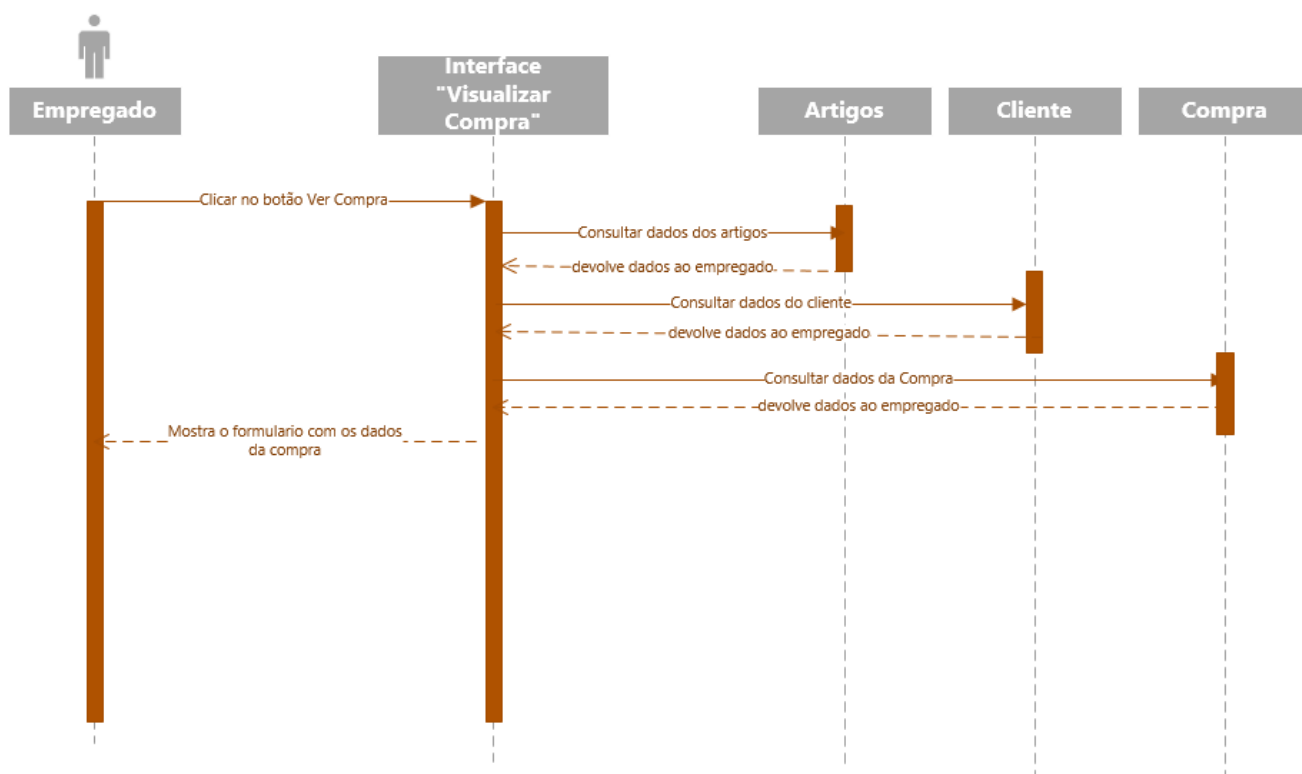


Figura 4 - Diagrama de Sequencia Visualizar Compra

# Diagrama de Estados

---

# Diagrama de Classes

O diagrama de classes é composto pelas tabelas, os seus atributos e as ações que irem ser realizadas em cada tabela.

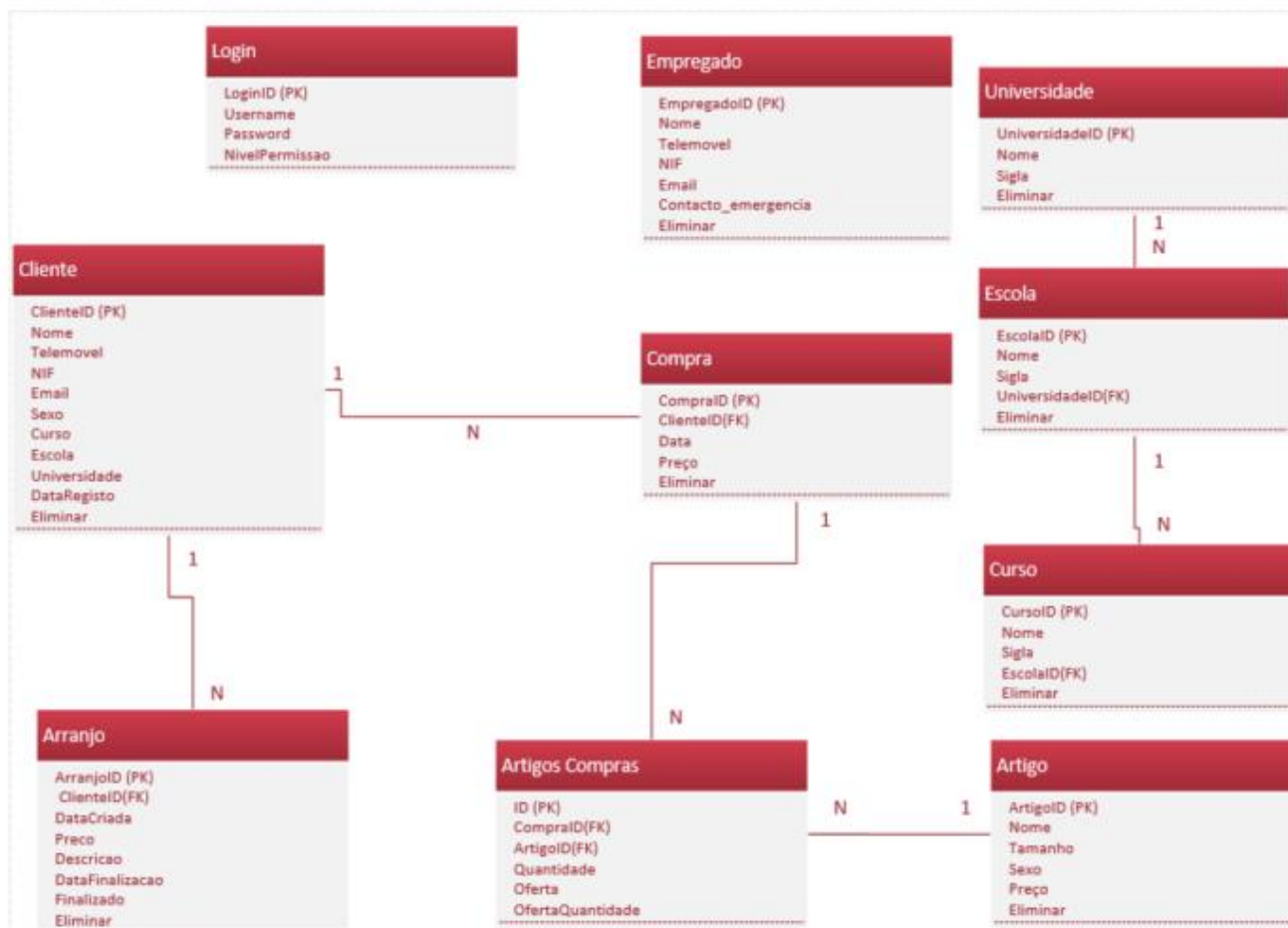


Figura 4 - Diagrama de Classes

# Atividades e Tempos Gastos

---

Diagrama de Contexto	1h
Resumos	5h
Tabelas de Atores e C.U.	3h
Diagrama de Casos de uso	1h
Descrição dos Casos de Uso	2h
Diagramas de Sequencia	1.30h
Diagrama de Classes	9h
Prototipos	9h
Relatório	3h



## Dicionário de dados

---

## Diagrama de Componentes

---

## Diagrama de Pacotes

---

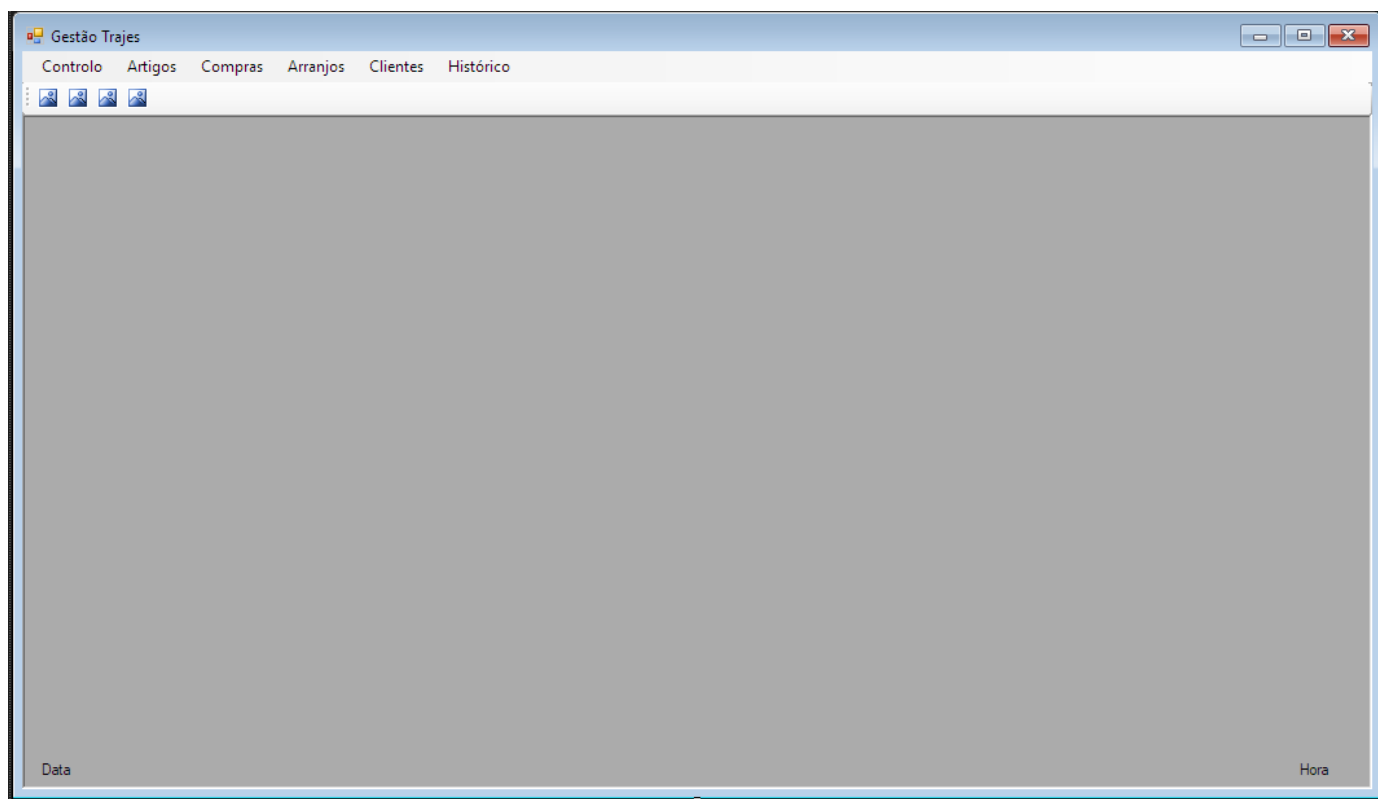
## Diagrama de Atividade

---

## Diagrama de Instalação

---

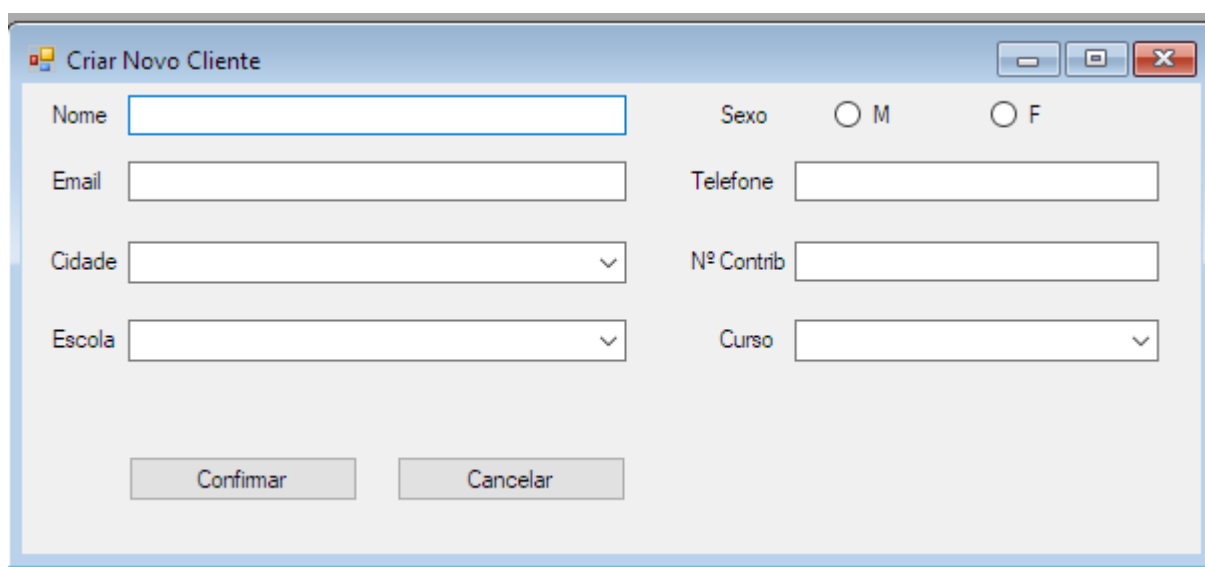
# Protótipo



Gestão Trajes

Controlo Artigos Compras Arranjos Clientes Histórico

Data Hora



Criar Novo Cliente

Nome

Sexo ☐ M ☐ F

Email

Telefone

Cidade

Nº Contrib

Escola

Curso

Confirmar Cancelar

# Conclusão

---