Técnico em Desenvolvimento de Sistemas

Docente: Rafael Sacramento

Sobre o Professor

- Formado em Bacharel em Sistemas de Informação, pós-Graduado em Análise de Sistemas com Ênfase em Governança e Docência da Tecnologia.
- Ex militar das Forção Armadas.
- Tenho 39 anos
- Atuo por mais de 14 anos como professor.

Histórico do Senac

O Senac – Foi criado em 10 de janeiro de 1946 através do decreto-lei 8.621. É uma entidade privada com fins públicos que recebe contribuição compulsória das empresas do comércio e de atividades assemelhadas. A nível nacional é administrado pela Confederação Nacional do Comércio.

Sobre o curso:

Carga Horária: 1.200 horas

Dividido em 12 Unidades Curriculares - UC;

Analisar requisitos e funcionalidades da aplicação – 108h;

Desenvolver algoritmos - 60h;

Auxiliar na Gestão de Projetos de Tecnologia da Informação- 108h;

Analisar programação estruturada e orientada a objetos- 48h;

Desenvolver aplicações desktop - 140h;

Criar e manter Banco de Dados - 108h;

Desenvolver aplicações web - 140h;

Desenvolver aplicações mobile - 140h;

Realizar operações de atualização e manutenção em aplicações desenvolvidas - 96h;

Realizar testes nas aplicações desenvolvidas - 108h;

Realizar operações de suporte junto ao usuário - 84h;

Projeto Integrador Desenvolvedor de aplicações - 60h;

Com Técnico em Desenvolvimento de Sistemas e no cargo de Desenvolvedor de Sistemas **se inicia ganhando R$ 2.221,00** de salário e **pode vir a ganhar até R$ 4.754,00.**

O que é um Técnico em Desenvolvimento de Sistemas?

Técnico em Desenvolvimento de Sistemas é um **profissional** que **possui conhecimentos e habilidades** para atuar na **criação, manutenção e suporte de sistemas de computador.** Eles são responsáveis por desenvolver softwares e aplicações de acordo com as **necessidades específicas de uma organização ou cliente.**

As principais responsabilidades de um Técnico em Desenvolvimento de Sistemas podem?

1 - **Análise de requisitos:** Compreender **as necessidades dos usuários e traduzi-las em especificações técnicas** para o desenvolvimento de sistemas de software.

2 - **Desenvolvimento de software:** Escrever **código**, criar **algoritmos** e **implementar funcionalidades** de acordo com as **especificações definidas.**

3 - **Teste e depuração:** Realizar **testes** para **garantir a funcionalidade e a qualidade do software**, além de **identificar e corrigir eventuais erros (bugs).**

4 - **Manutenção de sistemas:** Realizar **atualizações**, **correções** e **melhorias nos sistemas** existentes **conforme necessário.**

5 - **Suporte técnico:** Prestar **suporte aos usuários finais** para **resolver problemas relacionados ao software** e fornecer **orientações sobre seu uso adequado.**

6 - **Documentação: Documentar** o **processo de desenvolvimento**, bem como as **especificações e funcionalidades do software.**

Sobre você:

- Nome e como gostaria de ser chamado(a)?
- Local onde mora?
- Porque está fazendo este curso?
- Quais são suas expectativas?

Briefing (Resumo)

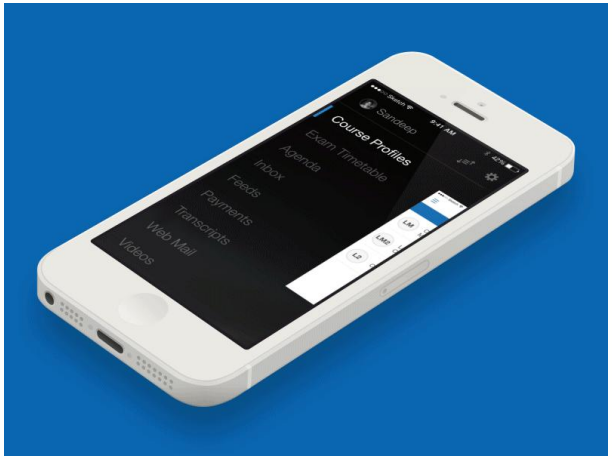
Um amigo meu definiu **briefing** como um “**Resumo”** para umas perguntinhas que se faz ao cliente sobre o que ele quer no site, aplicativo e etc”.

Basicamente, é mais ou menos isso, mesmo: o **briefing** é um **conjunto de perguntas e/ou atividades que servem para determinar como se dará o projeto,** elaboração e execução de determinada coisa

Protótipo

Um **protótipo de um site ou aplicativo é uma representação visual** e interativa do design e da funcionalidade do produto final.

Ele geralmente é criado durante a fase de desenvolvimento para ajudar na **comunicação das ideias, testar a usabilidade e obter feedback dos usuários antes da implementação completa.**



Canvas Pró
O que é?

Lançado em 2013, o **Canva é uma plataforma online de design** e **comunicação visual** que tem como missão colocar o poder do design ao alcance de todas as pessoas do mundo, para que elas **possam criar o que quiserem e publicar suas criações** onde quiserem.

O Canvas de **Modelo de Negócio** é uma **ferramenta** muito utilizada por empreendedores e gestores para **criar, analisar e ajustar modelos de negócio de forma eficaz.**

Canva


caneca
do dev.com

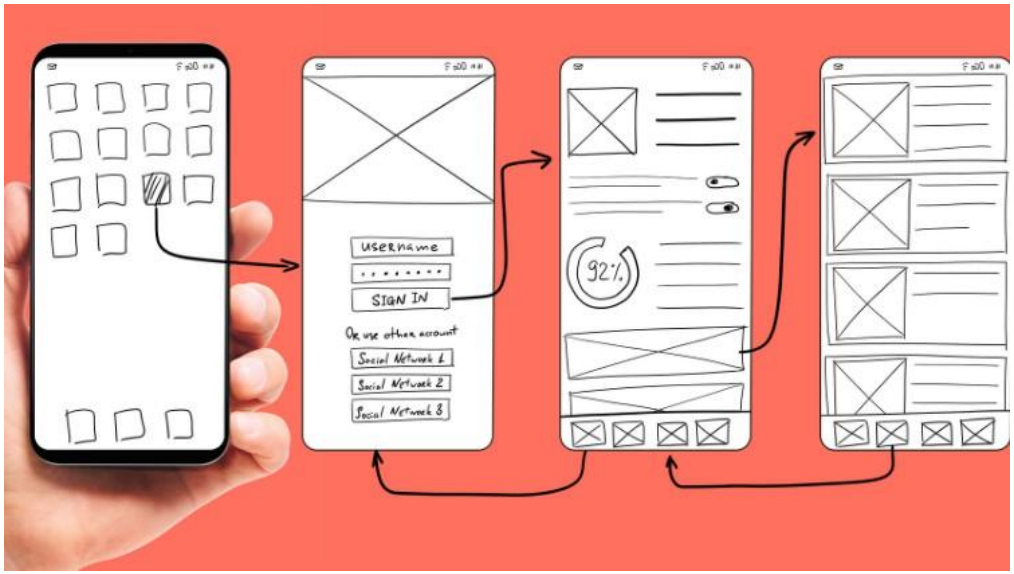
Figma
O que é?

É uma **poderosa ferramenta de design** colaborativo baseada na **nuvem**, usada principalmente para **criar interfaces de usuário, protótipos interativos e designs gráficos**. **Permite que equipes de design trabalhem juntas em tempo real**, facilitando a colaboração e a comunicação durante o processo de design.

Site: figma.com

Protótipo (WEB e Aplicativo)

É um design ou um modelo mais detalhado de algo que você pretende construir. Criar um protótipo significa construir uma amostra do produto, app, site, página ou ferramenta e testá-la para ver se funciona e se os processos ou funções são executados corretamente.



O que é Analise de Requisitos?

É uma das etapas **mais importantes do desenvolvimento** de **um projeto de software.** Ela faz parte da **fase** de **planejamento** e tem como **objetivo geral mapear o conjunto de ações e características que precisam compor o software**.

Como funciona a análise de requisitos?

1 - Identificar as necessidades;

Nesta primeira fase, os analistas precisam compreender as **perspectivas dos usuários, suas necessidades, dificuldades e outros problemas**. Além disso, aqui o sistema é especificado e o planejamento é realizado**.**

2 - Compreender as necessidades e soluções;

Os responsáveis estudam as necessidades e **reconhecem as informações que são importantes para a experiência do usuário e para o software.** Nesta fase também são selecionadas as melhores soluções para os problemas..

3 - Modelar o sistema;

Esta etapa consiste em utilizar o **recurso da modelagem para suportar a síntese da solução,** pois ele proporcionará ferramentas para simplificar o entendimento do software, incluindo seu comportamento e funcionalidades.

4 - Especificar os requisitos;

Nesta fase, os **profissionais irão estabelecer as interfaces, performances, funções, restrições e contexto do software**.

5 - Revisar o projeto;

Em conjunto com o usuário, aqui o **analista irá avaliar o objetivo f**inal para conferir **se existem falhas, omissões, redundâncias e inconsistências**.

O que é UML?

Linguagem de Modelação Unificada
(Unified Modeling Language)

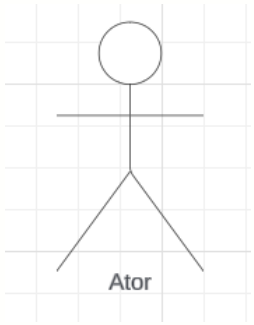
É uma **linguagem** padrão **utilizada** para **visualizar**, **especificar**, **construir** e **documentar artefatos** de **sistemas de software**.

O **UML** oferece uma **variedade de diagramas** que permitem aos desenvolvedores **representar diferentes aspectos de um sistema**, tais **como estrutura, comportamento, interações e fluxos de dados.** Esses **diagramas incluem diagramas de classes**, diagramas de **sequência, diagramas de atividades**, diagramas **de casos de uso**, entre outros. O **UML** é uma ferramenta valiosa no **processo de desenvolvimento de software**, pois **ajuda a comunicar as ideias de projeto de forma clara** e concisa entre os membros da equipe de desenvolvimento e os stakeholders do projeto.

Diagrama de Casos de Uso

A funcionalidade do sistema é definida por um conjunto de casos de uso. Os **casos** de uso têm por **objetivo caracterizar os requisitos funcionais do sistema** e identificar entidades relevantes e sua interação com o sistema. **Cada caso de uso representa uma sequência de ações e deve ser descrito textualmente**. A descrição textual é um documento narrativo que descreve **sequência de eventos/ações realizadas pelo sistema**, quando estimulados por um ator que interage ou usa o sistema. Os **casos de uso devem ser passíveis de compreensão tanto por desenvolvedores como por usuários** e precisam ser completos, consistentes e não ambíguos.

Os **diagramas UML** também fornecem os **diagramas de casos de uso**, que é o **conjunto de notações gráficas** que **permite representar** os casos de uso, **atores e associações entre eles**.



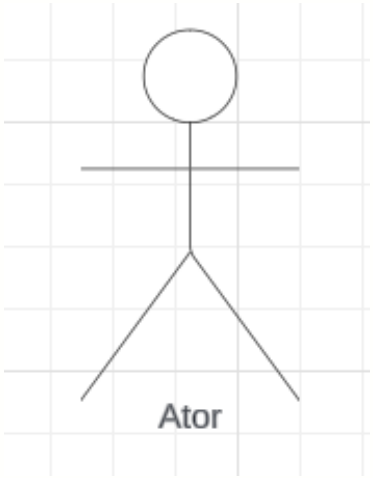
Ator

Caso de uso

Relacionamento

Ator

Ator pode **ser** uma **pessoa**, **organização** ou **sistema externo** que **interage** com seu **aplicativo** ou **sistema**. Eles devem ser objetos externos que produzam ou consumam dados.



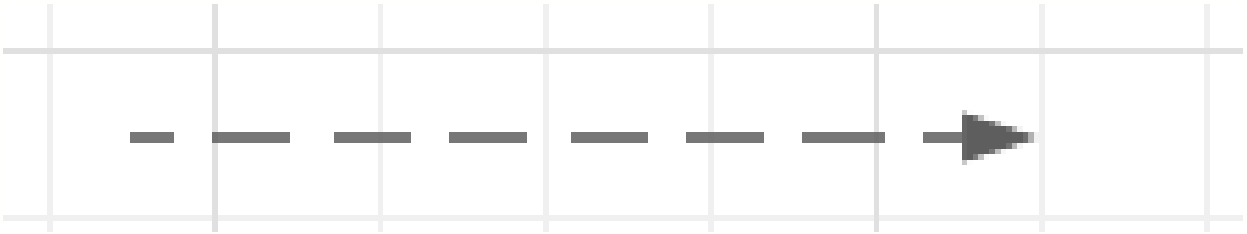
Caso de uso

Descreve as **interações** entre **um** ou **mais Ator** e o **sistema** para gerar um resultado **de valor observável para o ator iniciador**.

Relacionamento

Mostram **casos de uso, atores e os relacionamentos entre eles.**

Por exemplo, o **relacionamento** entre um **ator** e um **caso de uso** ilustra que o **ator** pode usar certas funcionalidades do sistema de negócios.



A **Figura A.2** apresenta um **exemplo de diagrama de casos de uso.**

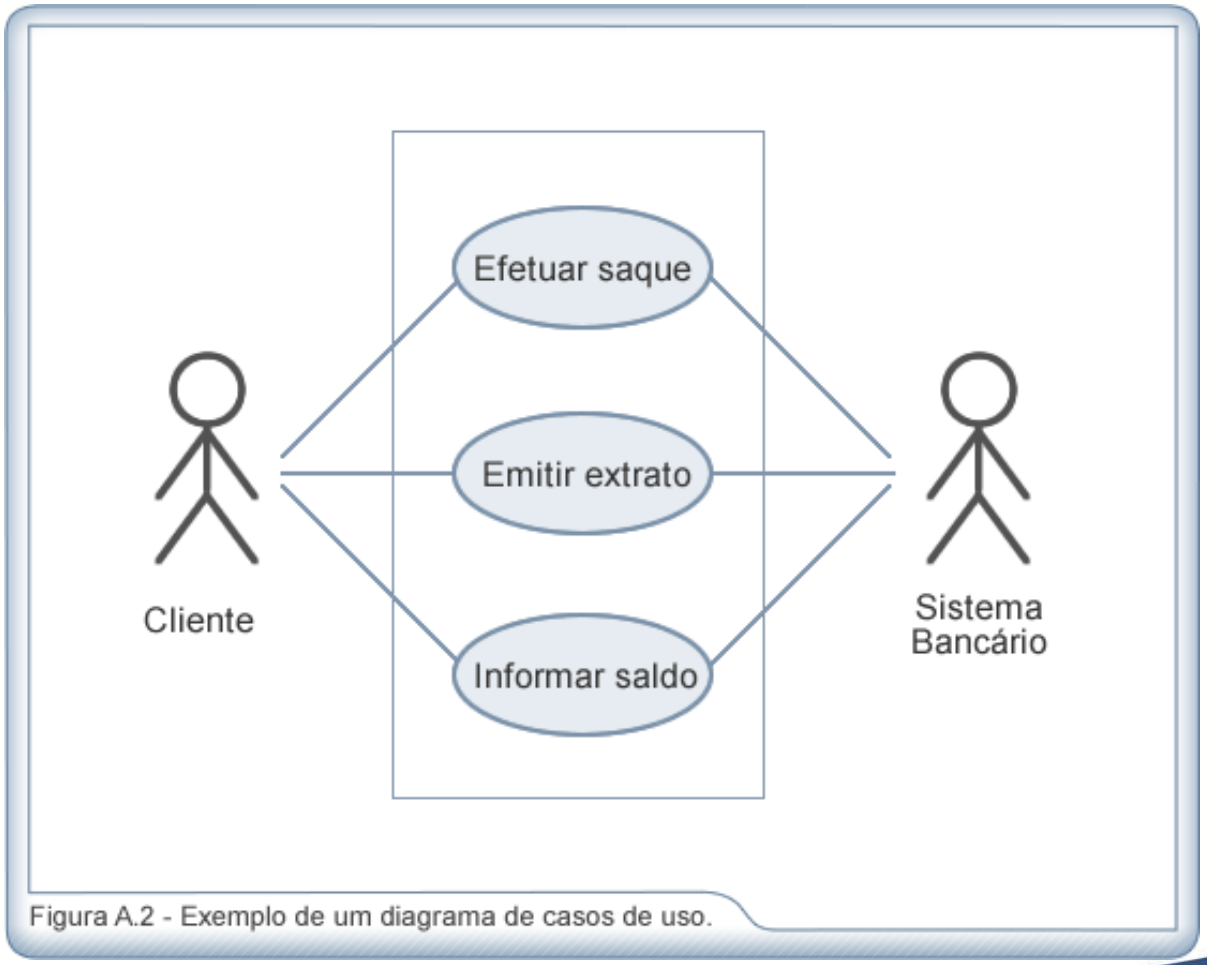
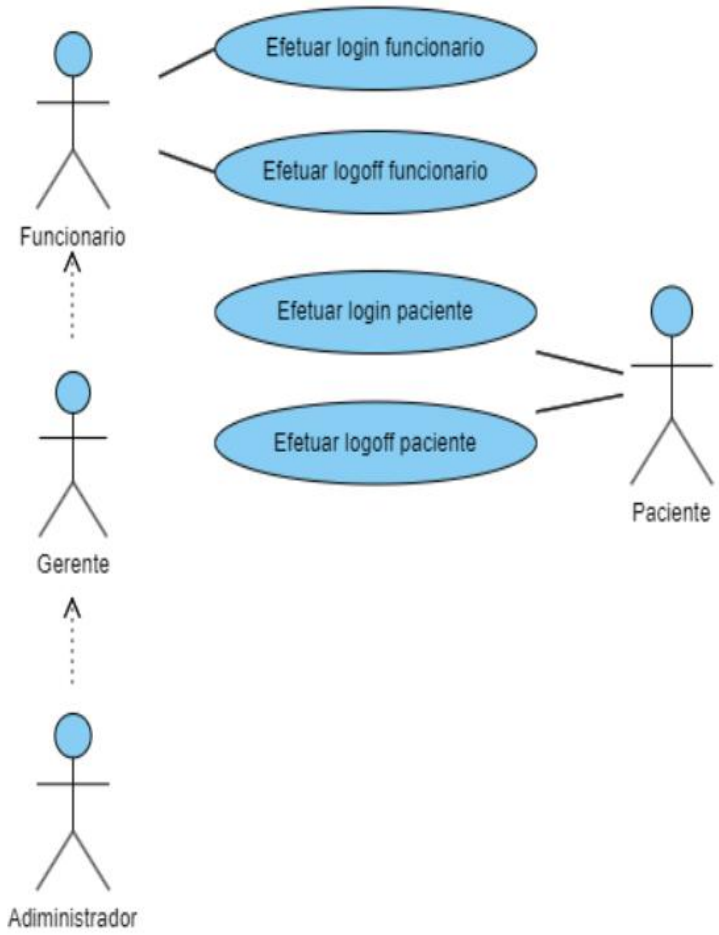


Figura A.2 - Exemplo de um diagrama de casos de uso.

Gerenciar Acesso



Gerenciar Paciente

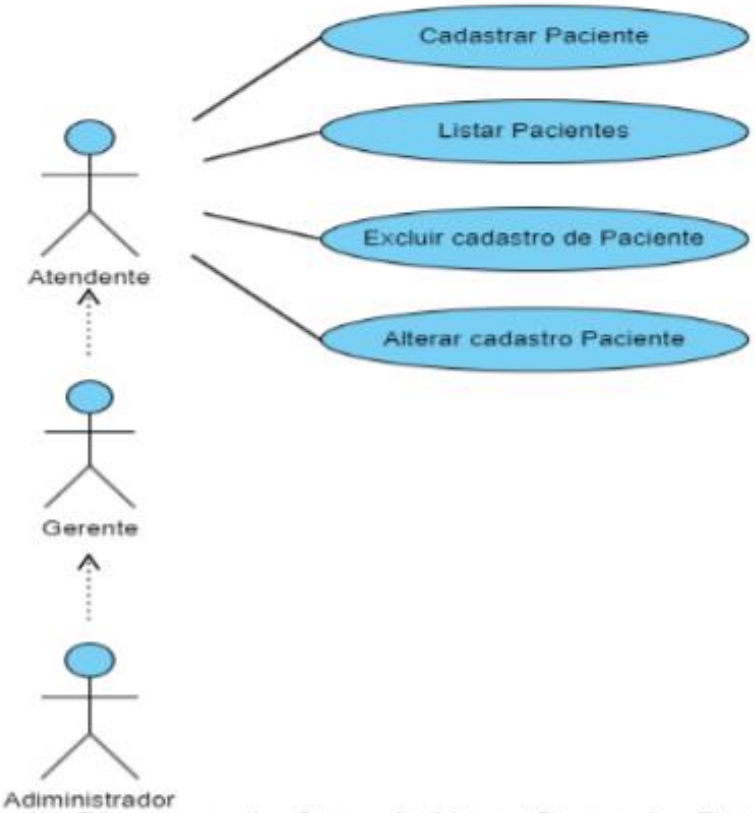
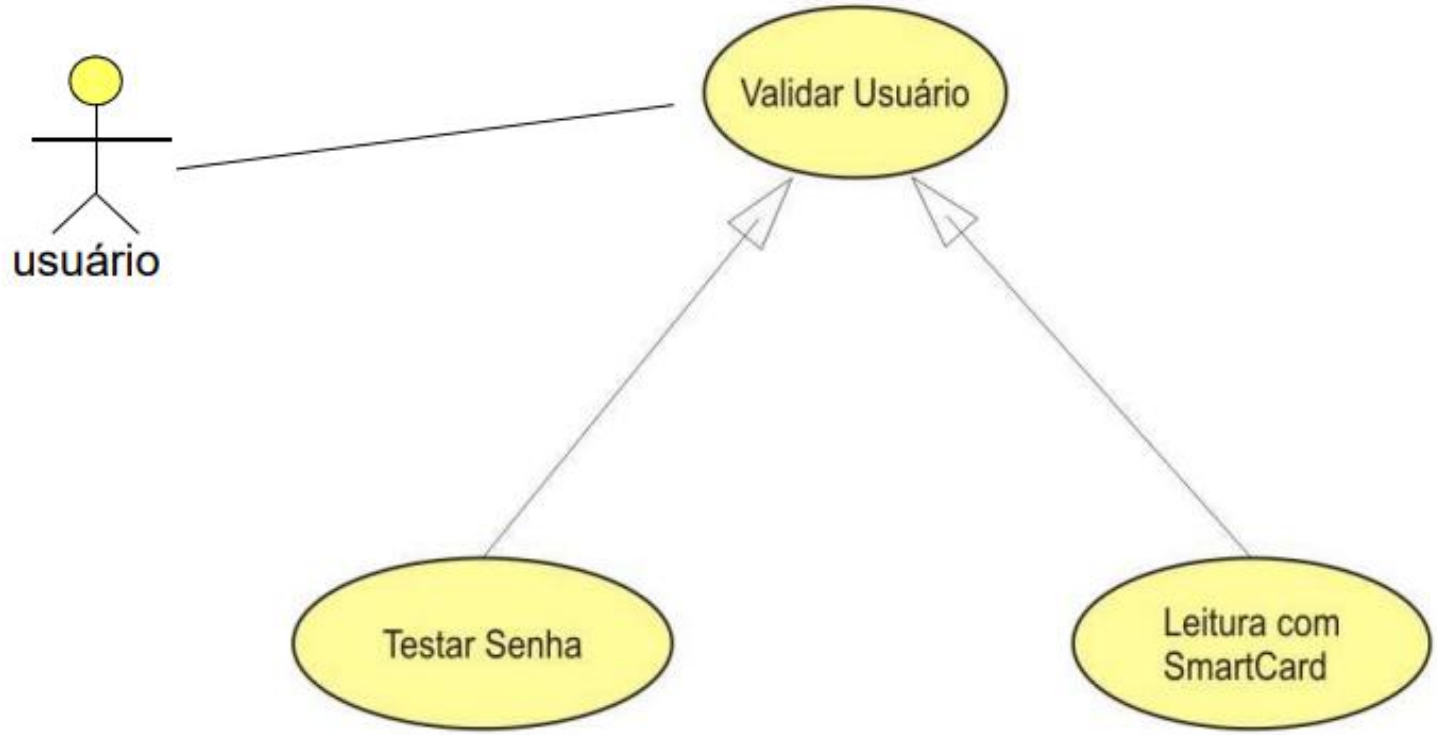


Figura 3 - Diagrama de Caso de Uso - Gerenciar Paciente

Generalização em Casos de Uso

É utilizada quando você encontra **dois ou mais casos de uso** que têm **comportamento**, **estrutura e finalidade comuns**. Quando isso ocorre, você pode descrever as partes compartilhadas em um caso de uso novo, geralmente abstrato, que é especializado pelos casos de uso filho.

O caso de uso “**Validar Usuário**” é especializado em outros dois, que utilizam diferentes mecanismos de identificação do usuário: “**Testar Senha**” e “**Leitura com Smartcard**”.



Caso de Uso Include

É utilizada quando o caso de uso A “inclui” o caso de uso B, significa que sempre que o caso de uso A for executado o caso de uso B também será executado. A direção do relacionamento é do caso de uso que está incluindo para o caso de uso incluído.

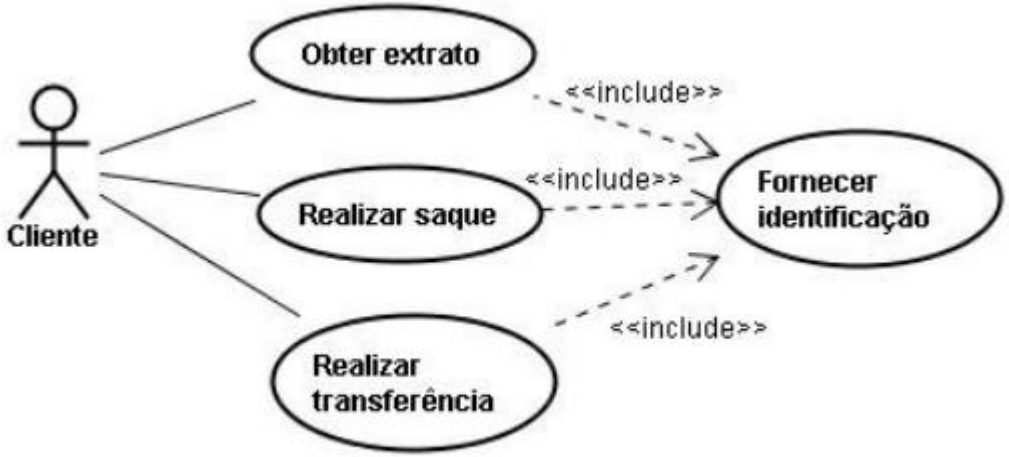


Figura relacionamento <<include>>

Caso de Uso Extend

É utilizada para representar situações em que certas **funcionalidades opcionais** podem ser ativadas em determinadas circunstâncias ou quando certos eventos ocorrem durante a execução do sistema.

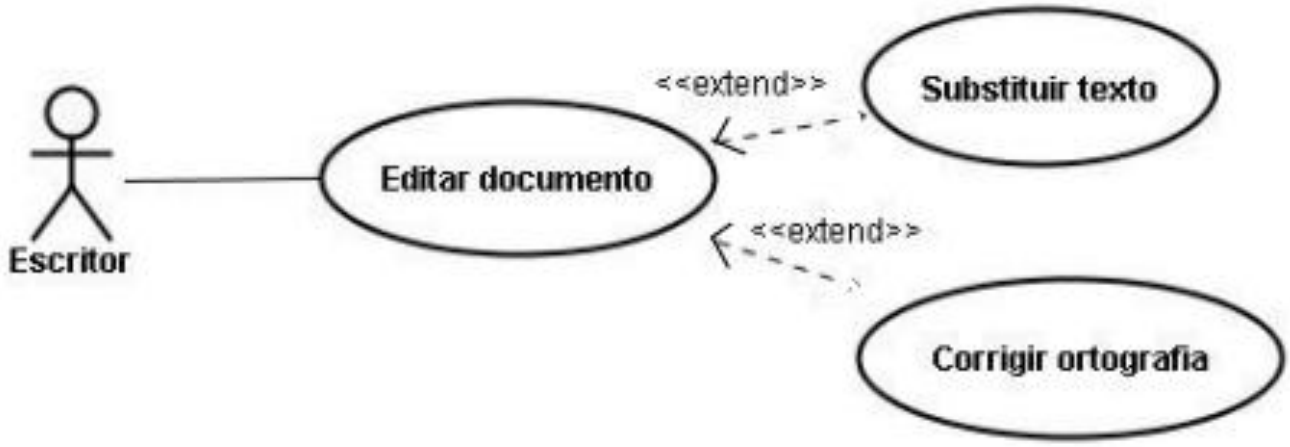
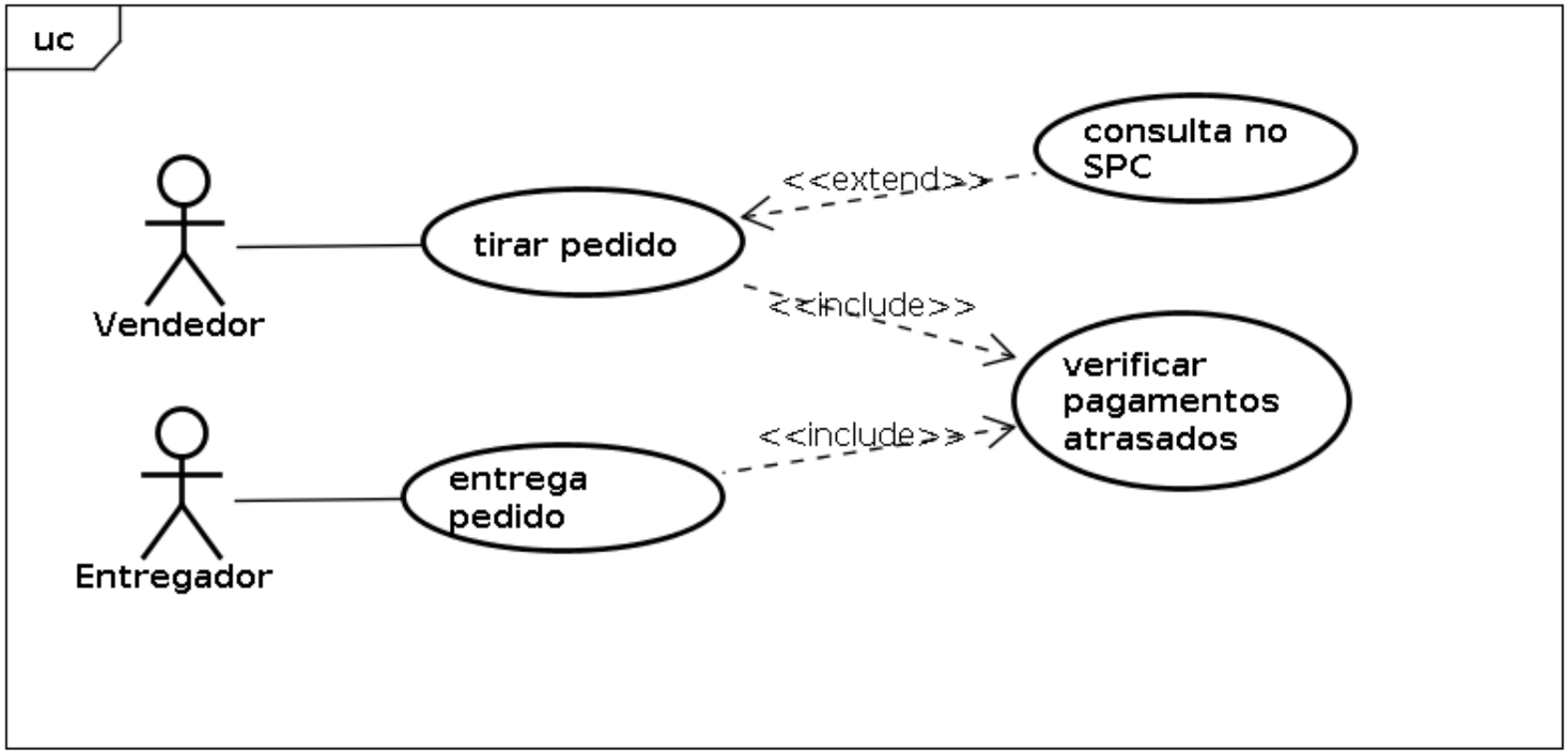
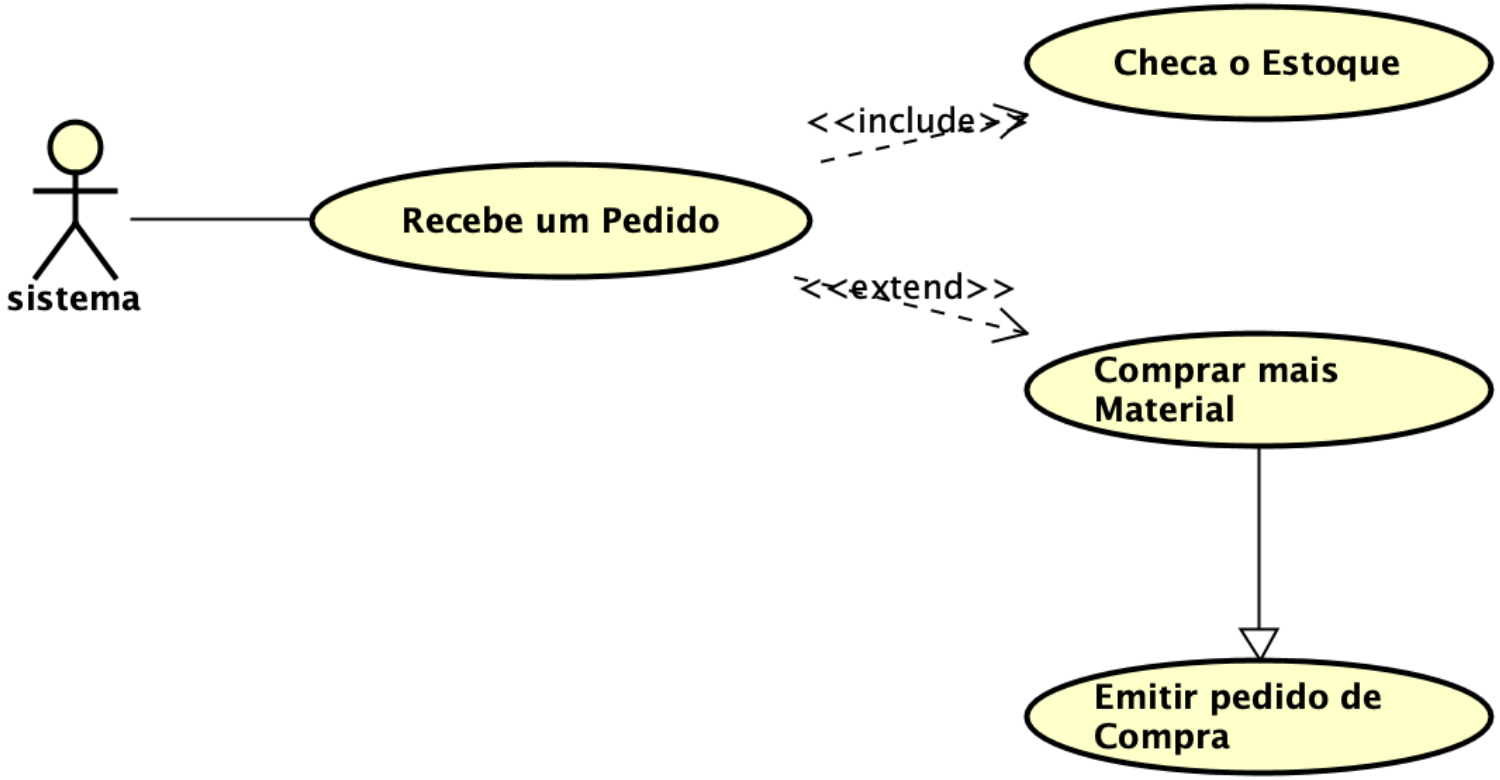


Figura relacionamento <<extend>>


uc
consulta no SPC
<<extend>>
tirar pedido
Vendedor
<<include>>
verificar pagamentos atrasados
<<include>>
entrega pedido
Entregador


Checa o Estoque
<<include>>
sistema
Recebe um Pedido
<<extend>>
Comprar mais Material
Emitir pedido de Compra

Responsabilidades do Caso de Uso

É **descrever** o **comportamento do sistema em termos de interações entre os atores externos e o sistema**. Ele detalha as **ações** que o **sistema realiza em resposta a estímulos externos**, normalmente representados por atores. O caso de uso também descreve as condições pré e pós-condição para cada ação. Em resumo, **ele ajuda a entender como o sistema funciona em um nível mais alto**, sem entrar em detalhes de implementação.

Tabela 1 – Responsabilidades de cada Ator

Funcionário/Dentista	▪ Efetuar Login ▪ Manter Paciente ▪ Manter Consultas ▪ Manter Arquivos ▪ Pesquisar Financeiro ▪ Manter Procedimento
Administrador	▪ Efetuar Login ▪ Manter Paciente ▪ Manter Consultas ▪ Manter Arquivos ▪ Pesquisar Financeiro ▪ Manter Procedimento ▪ Manter Funcionário

Abreviações

A área de **tecnologia** é **cheia** de **termos** e **siglas** que as **vezes confundem os usuários** e **até** os **profissionais com experiência** no assunto.

Geralmente são formadas pelas **primeiras letras de cada palavra de uma expressão para tornar a escrita mais eficiente e rápida.** Esse **método** é **conhecido** como **sigla**.

Ao utilizar apenas as primeiras letras, é possível condensar a informação sem perder o significado da expressão original. **Isso facilita a comunicação escrita**, especialmente em **contextos onde a brevidade é importante**, como em textos **técnicos, acadêmicos ou em comunicações formais.** Além disso, as siglas permitem que termos longos e complexos sejam facilmente reconhecidos e compreendidos por pessoas familiarizadas com o contexto em questão.

LISTA DE ABREVIATURAS E SIGLAS

BD .....Banco de Dados

RF.....Requisito Funcional

RNF.....Requisito Não Funcional

RN .....Regra de Negócio

REQ .....Requisito

UC.....User Case (ou Caso de Uso)

DAO .....Data Access Object (ou Objeto de Acesso a Dados)

UML.....Unified Modeling Language (ou Linguagem de Modelagem Unificada)

SQL.....Structured Query Language (ou Linguagem de Consulta Estruturada)

PDF.....Portable Document Format (ou Formato de Documento Portável)

REQUISITO FUNCIONAL

REQUISITO NÃO FUNCIONAL

Requisitos Funcionais (RF)

Os requisitos funcionais são as instruções que definem o que essa máquina deve fazer.

No **contexto** de um **software**, eles **delineiam** as **ações específicas** que o **sistema deve ser capaz de executar para satisfazer as necessidades dos usuários.**

Exemplo Prático: Aplicativo de Lista de Tarefas

- **Permitir que** o **usuário crie uma nova tarefa;**
- **Oferecer a opção** de **marcar uma tarefa como concluída;**
- **Proporcionar um meio** de **excluir tarefas;**
- **Facilitar** a **edição** de **tarefas já criadas**. um **meio** de **excluir tarefas;**

Essas são as tarefas essenciais que definem o funcionamento do aplicativo.

E O Que São Requisitos Não Funcionais (RNF)?

Enquanto os **requisitos funcionais são as ações**, os requisitos não funcionais **caracterizam como essas ações serão realizadas.**

Eles **são** os **atributos** de **qualidade** que **determinam** a **eficiência**, a **usabilidade** e a **robustez** do **software**.

Exemplo Prático: Aplicativo de Lista de Tarefas

- **Tempo de resposta:** O aplicativo deve responder a todas as interações dentro de 2 segundos;
- **Usabilidade:** Deve ser intuitivo, permitindo que o usuário o utilize sem treinamento prévio;
- **Segurança:** Os dados dos usuários devem ser armazenados e transmitidos de forma segura;
- **Disponibilidade:** O aplicativo deve estar acessível para uso 99,9% do tempo.

Esses **parâmetros garantem que o aplicativo não apenas funcione**, mas que **também ofereça uma experiência de qualidade ao usuário.**

- **A Importância da Integração Entre Funcional e Não Funcional**

Um software é mais que a soma de suas partes.

É vital integrar os **requisitos funcionais** e **não funcionais desde** o **início do desenvolvimento.**

Um aplicativo que realiza todas as suas funções, mas **falha em ser rápido e confiável**, é **tão problemático quanto um aplicativo** que **é eficiente e seguro**, **mas não realiza as tarefas necessárias.**

Priorizar requisitos é uma arte.

Ao começar, é **importante focar primeiro no que seu software precisa fazer (requisitos funcionais)** e **depois em como ele deve fazer (requisitos não funcionais)**.

Como Definir Requisitos de Forma Eficaz

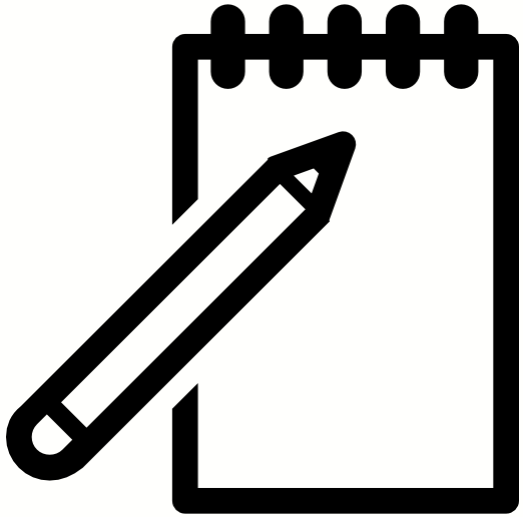
Aqui estão algumas estratégias para acertar:

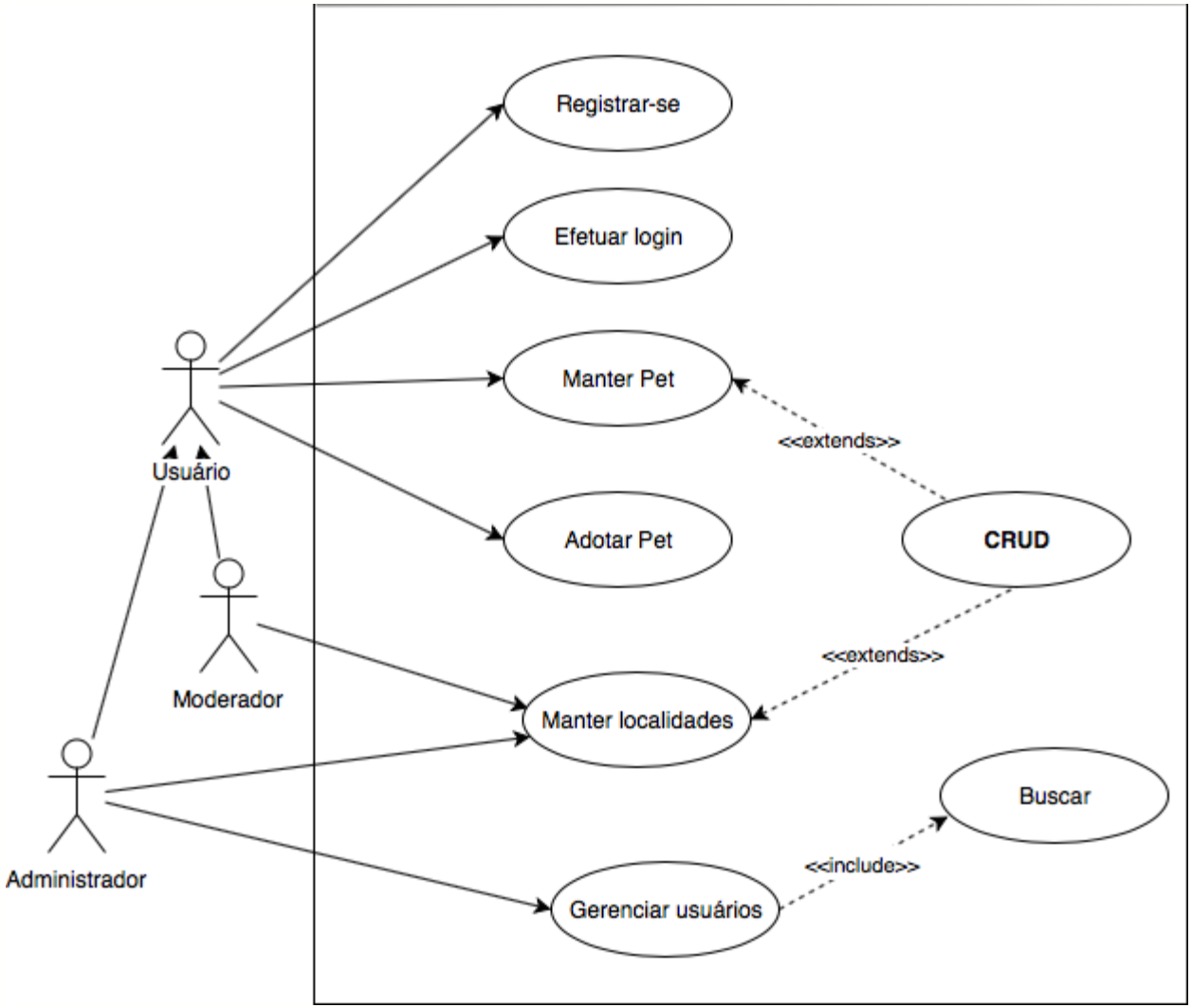
- **Entenda seu usuário:** Conheça profundamente quem irá usar seu aplicativo e o que eles precisam dele.
- **Priorize os requisitos:** Nem tudo precisa ser desenvolvido imediatamente. Foque no essencial primeiro.
- **Clareza é chave:** Requisitos mal definidos podem levar a mal-entendidos e a um produto final inadequado.
- **Esteja aberto a mudanças:** Requisitos podem evoluir à medida que você aprende mais sobre as necessidades dos usuários.

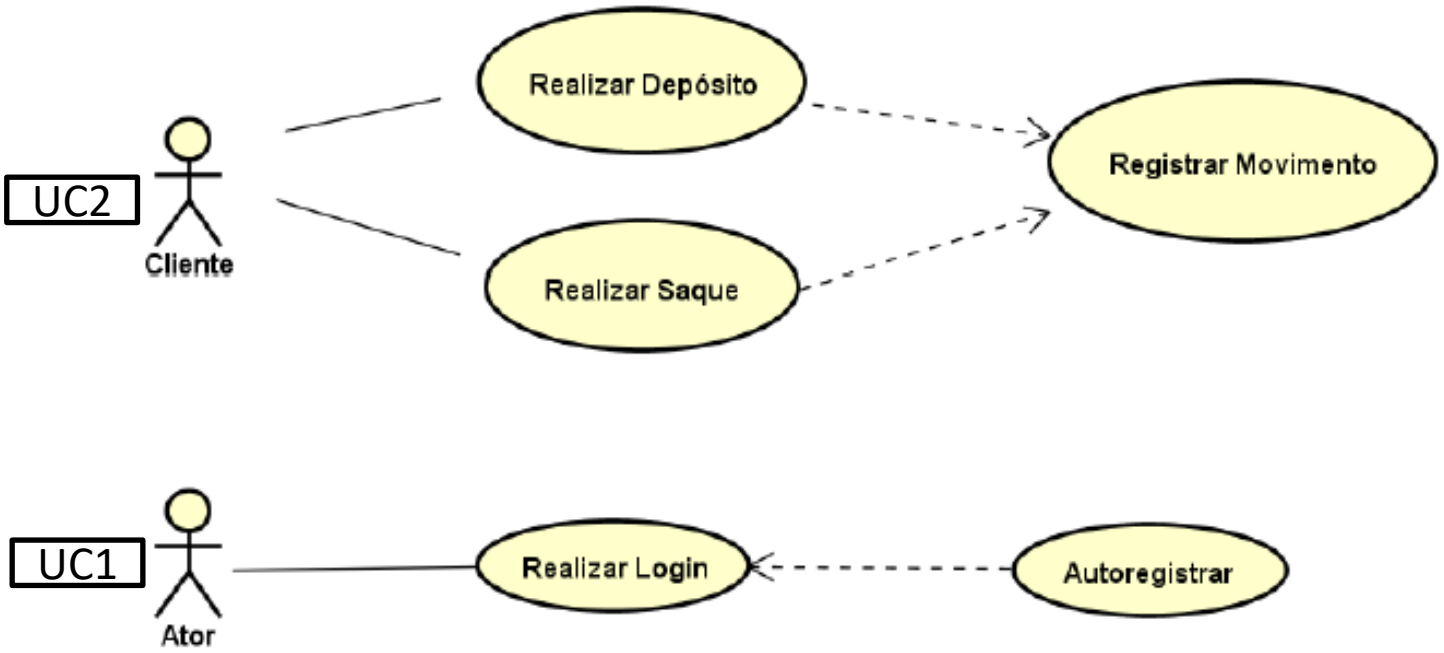
Dominar os conceitos de requisitos funcionais e não funcionais é essencial para qualquer aspirante a desenvolvedor de software.

Especificação dos Casos de Uso

É uma **técnica de especificação** que **descreve uma sequência de ações que o sistema** deve **realizar para produzir uma resposta para um ator.** Na **realidade**, tem-se **uma sequência da interação** entre **caso de uso e ator.**




Registrar-se
Efetuar login
Manter Pet
Adotar Pet
CRUD
<<extends>>
<<extends>>
Manter localidades
Buscar
<<include>>
Gerenciar usuários
Usuário
Moderador
Administrador


UC2
Cliente
Realizar Depósito
Realizar Saque
Registrar Movimento
UC1
Ator
Realizar Login
Autoregistrar

Especificação dos casos de uso

5.1 Efetuar Login

Tabela 5.1.1 - Tabela de Caso de Uso Realizar Login

Nome UC001	Efetuar Login	**Rastreabilidade**	RF002 RN002
Atores	Todos possíveis usuários.		
Participação do ator	O ator será capaz de interagir com o sistema através da tentativa de login.		
Pré-condições			
Abrir o sistema.			
Pós-condições			
Acesso a partes do sistema, dependendo do nível de acesso, definido no cadastro de usuário.			
Fluxo Básico			
1	O ator inicia o sistema.		
2	O sistema abre uma tela e apresenta os seguintes campos de formulário: Login e Senha, e botões de entrar **[FA02]** e sair **[FA01]**. Caso o ator deixe um ou os dois campos em branco e tente entrar, o sistema retornará o fluxo de exceção **[E01]**.		
3	O caso de uso será encerrado.		

Fluxo de Alternativo (FA)

Descrevem o que acontece **quando o ator faz uma escolha alternativa**, diferente da descrita no fluxo principal, para alcançar seu objetivo:

✓**Podem descrever escolhas exclusivas entre si.**

✓ **Pós-condição – estado que o sistema alcança após o caso de uso ter sido realizado.**

Fluxos Alternativos	
FA01 - Sair	
1	O ator escolhe a opção de sair do sistema.
2	É apresentada uma janela contendo a **[M010].**
3	O ator escolhe Sim.
4	O sistema é finalizado.
FA02 - Entrar	
1	O ator escolhe a opção de entrar no sistema.
2	O sistema valida o login de acordo com o Ponto de Inclusão, deste caso de uso.

Fluxo de Execução (FE)

A execução sempre começa no primeiro comando de um programa. Os comandos são executados um a um, em ordem, de cima para baixo. A definição de funções não altera o fluxo de execução do programa, mas lembre-se que comandos dentro de uma função não são executados até que a função seja chamada.

Fluxos de Exceção	
E01- Campos Obrigatórios Em Branco	
1	No passo **[2]** do fluxo básico o sistema retorna a seguinte mensagem de erro **[M03]**.
2	O sistema envia o ator ao passo **[2]** do fluxo básico.
Pontos de Extensão	
Seção não aplicável para este caso de uso.	
Pontos de Inclusão	
Referente ao caso de uso UC002 – Validar Login.	

Glossário de Mensagens

É uma **espécie de dicionário de palavras não tão conhecidas**, seja porque **são palavras de uso técnico ou porque são palavras regionais e de outro idioma.** Então, de forma geral, **o glossário serve para explicar o significado de alguns termos que, por algum motivo, o leitor ou leitora pode não conhecer**. Segue o modelo:

Tabela 6.1.1 - Tabela de Mensagens do Sistema

Mensagens	
M01	Cadastro efetuado com sucesso!
M02	Login ou senha inválido.
M03	Campos obrigatórios em branco.
M04	Cliente com cadastro incompleto, por favor finalize o cadastro.
M05	Tem certeza que deseja excluir esses dados?
M06	O informado no campo de confirmação da senha é diferente do informado no campo senha.
M07	Alterado com sucesso!
M08	Registro excluído!
M09	[usuario] Bem vindo!
M010	Deseja sair?
M011	Tem certeza que deseja realizar este estorno?
M012	Descontado com sucesso.

Metodologia Squad (Pendências)?

Consiste na formação de um time multidisciplinar que tenha um objetivo comum. Em outras palavras, trata-se de reunir profissionais de áreas diferentes, com especialidades distintas para que, juntos, alcancem um mesmo propósito.

Metodologia Scrum?

Uma **equipe de três a nove pessoas** em um **projeto** utilizando a **metodologia Scrum** desempenha papéis específicos e executa **atividades importantes** para o desenvolvimento do projeto de forma iterativa e incremental. Os **papéis principais dentro de uma equipe Scrum são:**

Scrum Master: É responsável por facilitar o uso do **framework (estrutura)** Scrum, **removendo impedimentos** que **atrapalham o progresso do time**, auxiliando na adoção das práticas ágeis e garantindo que a equipe siga as regras e valores do Scrum.

Product Owner (Proprietário do produto): É o **responsável** por representar os **stakeholders (partes interessadas)** clientes, usuários, patrocinadores e definir os **requisitos e prioridades** do produto no **Backlog** (Pendências) do Produto (Product Backlog).

Desenvolvedores: São os **membros da equipe** que realizam o trabalho necessário para **entregar as funcionalidades** do produto. Eles são autogerenciados e multidisciplinares, trabalhando juntos para concluir as tarefas do Sprint Backlog. Em um **projeto Scrum**, a equipe de **três a nove pessoas** trabalham em **ciclos chamados Sprints**, que geralmente têm **duração de 1 a 4 semanas**. Durante cada Sprint, a equipe realiza as seguintes atividades:

Sprint Planning (Planejamento da Sprint): A equipe **se reúne com o Product Owner** para selecionar **itens do Backlog** do Produto a serem trabalhados durante a Sprint. Eles estimam o esforço necessário para concluir as tarefas e definem o **Sprint Goal** (Metas e objetivo da Sprint).

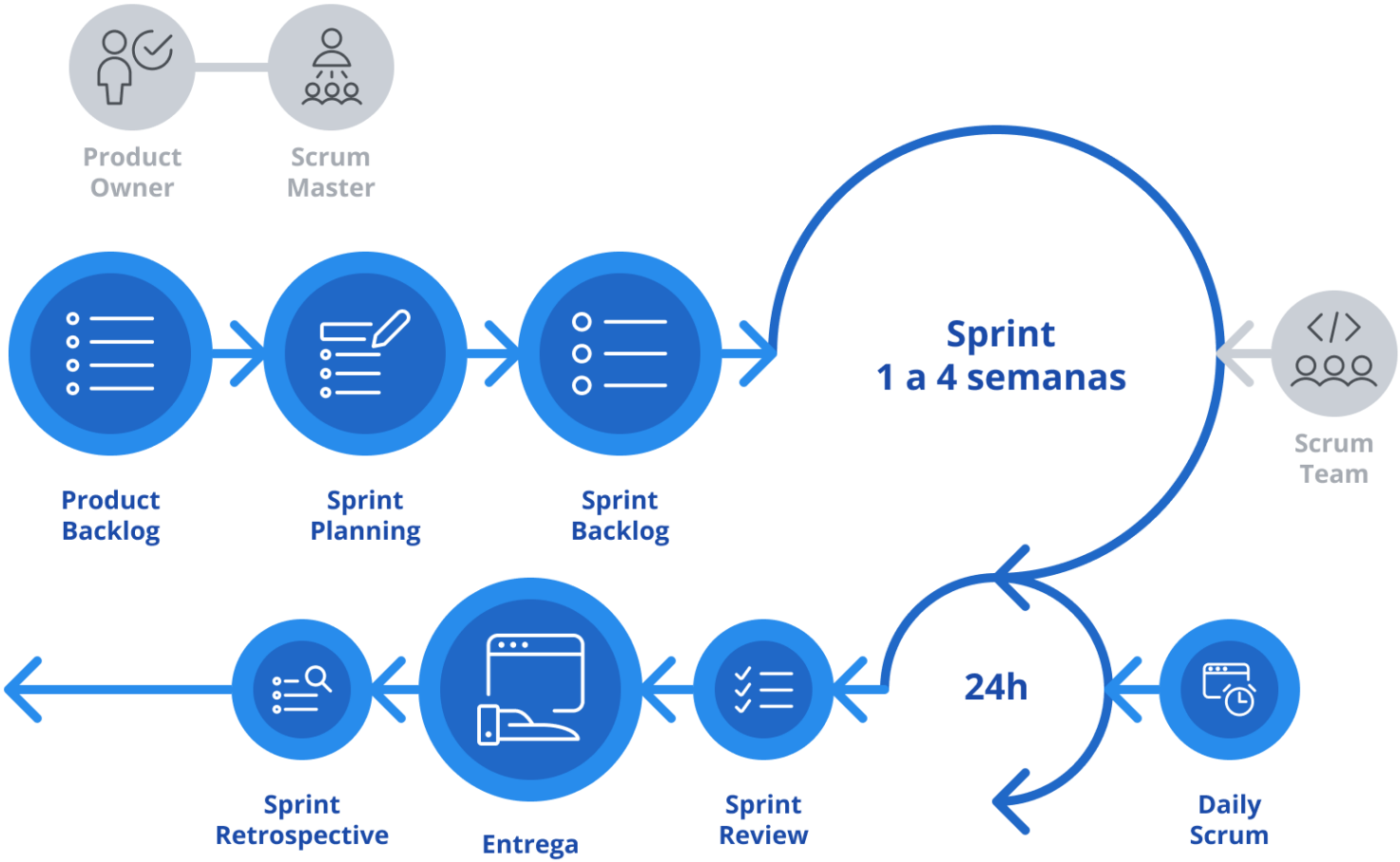
Daily Scrum (Scrum Diário): **Todos os dias**, a equipe realiza **uma breve reunião** (geralmente de **15 a 30 minutos**) para **sincronizar as atividades**, discutir progresso e **identificar quaisquer impedimentos.**

Desenvolvimento: Durante a Sprint, a **equipe** trabalha para **desenvolver e entregar** as funcionalidades **selecionadas** no **Backlog do Produto**. Eles **colaboram** para **garantir que o trabalho seja concluído** e revisam **constantemente o progresso em direção ao Sprint Goal.**

Revisão da Sprint (Sprint Review): Ao **final da Sprint**, a **equipe realiza uma reunião com os stakeholders** para demonstrar o que **foi concluído** durante a **Sprint e receber feedback**.

Retrospectiva da Sprint (Sprint Retrospective): Também ao **final da Sprint**, a e**quipe se reúne para revisar seu desempenho,** identificar **oportunidades de melhoria** e **planejar ações para aumentar a eficácia** do próximo Sprint. Essas são algumas das principais atividades **realizadas por uma equipe de 5 pessoas** em um projeto utilizando a metodologia Scrum. O **Scrum promove uma abordagem colaborativa e iterativa para o desenvolvimento de projetos**, permitindo que a **equipe se adapte às mudanças** e entregue **valor ao cliente de forma mais rápida e eficiente.**

Retrospectiva da Sprint (Sprint Retrospective): Também ao **final da Sprint**, a e**quipe se reúne para revisar seu desempenho,** identificar **oportunidades de melhoria** e **planejar ações para aumentar a eficácia** do próximo Sprint. Essas são algumas das principais atividades **realizadas por uma equipe de 5 pessoas** em um projeto utilizando a metodologia Scrum. O **Scrum promove uma abordagem colaborativa e iterativa para o desenvolvimento de projetos**, permitindo que a **equipe se adapte às mudanças** e entregue **valor ao cliente de forma mais rápida e eficiente.**



Acesse o exercício: https://forms.gle/WCAnfB32TMyDSx8c6

LinkedIn

É uma **plataforma** de **mídia social focada em negócios e emprego** que **funciona através de sites e aplicativos móveis.** Fundada em dezembro de 2002 e lançada em 5 de maio de 2003, **de propriedade da Microsoft.**

https://www.linkedin.com

Diagrama de Classes

Os diagramas de classes **mostram as diferentes classes** que **fazem parte de um sistema** e **como elas se relacionam**. Uma **classe é representada** em UML por um **retângulo**, com o **nome da classe**, e podem **também mostrar os atributos** e operações (métodos) da classe.

O que são atributos: São as particularidades, **qualidades e características que são próprias de alguém ou algo**. Por norma, **os atributos estão relacionados com aspectos positivos.** Os atributos costumam ser características exclusivas de determinada pessoa, grupo ou coisa.

As classes de um diagrama de classes podem possuir diferentes relacionamentos como:

Generalização: o relacionamento de generalização conecta classes generalizadas com outras mais especializadas. A generalização **é um relacionamento de itens gerais** (superclasses) e itens mais específicos (subclasses) e é considerado um como “é um tipo de”;

A **Figura A.1** apresenta um exemplo de **diagrama de classe.**

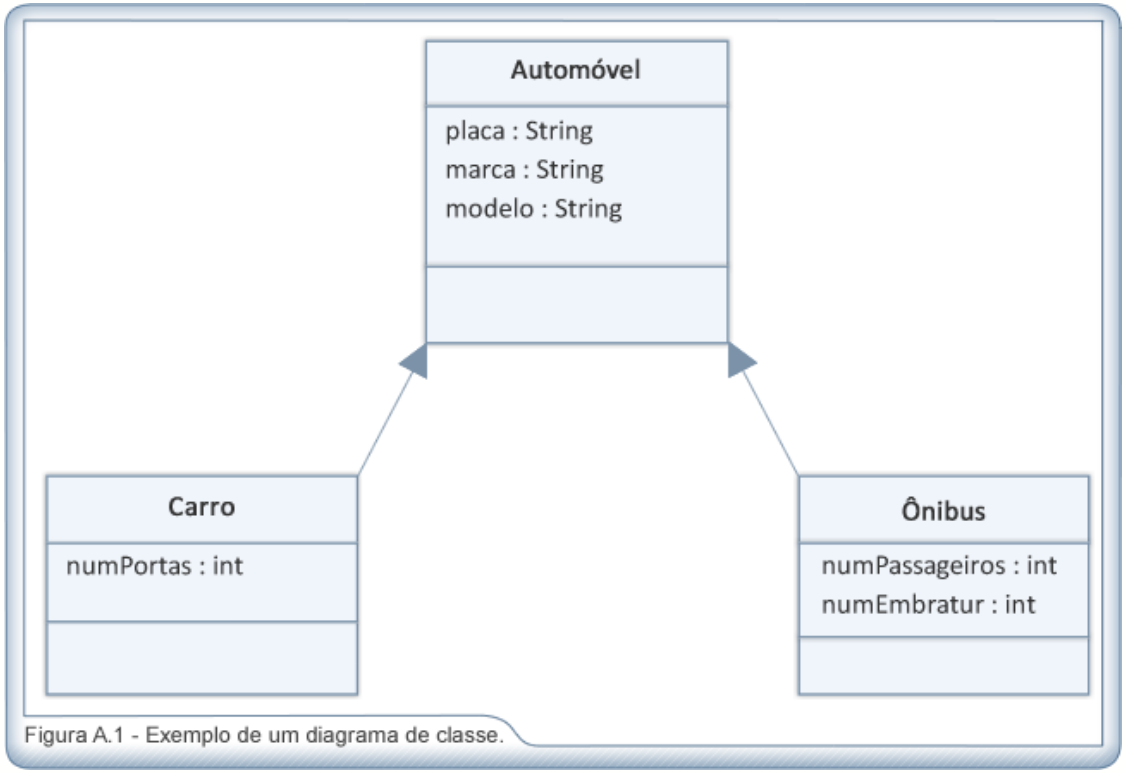


Figura A.1 - Exemplo de um diagrama de classe.

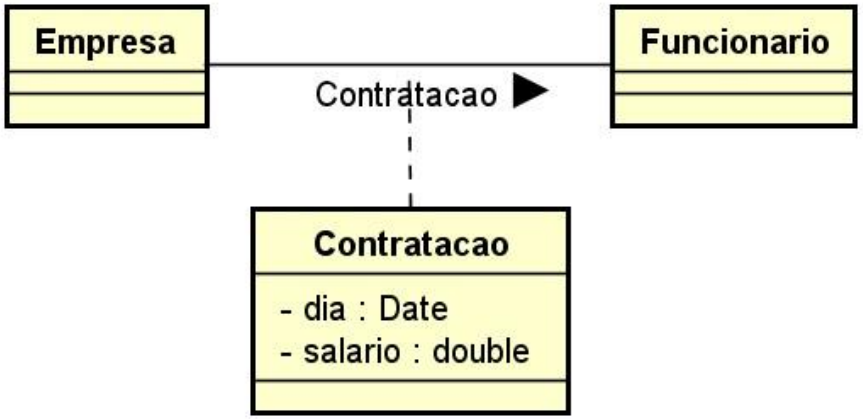
Na programação de computadores, uma cadeia de caracteres ou string **é uma sequência de caracteres, geralmente utilizada para representar palavras, frases ou textos de um programa.**

Int é a representação de **números inteiros.**

Associação: a associação é um **relacionamento estrutural entre instâncias** (especifica objetos de um item conectados a objetos de outro item). Uma **pura associação entre duas classes representa um relacionamento estrutural entre pares em que as classes estão em um mesmo nível e uma não é mais importante que outras.** Também é possível a partir de uma **associação conectando duas classes, navegar do objeto de uma classe até o objeto de outra classe e vice-versa. Em associação um objeto "usa um" outro objeto**;

Exemplo:

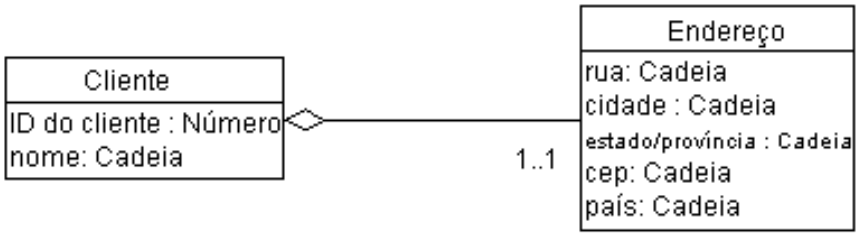
Suponhamos a situação de contratação de um funcionário por uma empresa. Existiriam duas classes, a classe "**Empresa**" e a classe "**Funcionário**". Entre essas **classes existiria um relacionamento chamado, por exemplo, "contrata".** Suponhamos também que seja interessante para o sistema guardar as informações do dia da contratação e do salário acordado.



Agregação: em **uma pura associação entre duas classes**, essas **classes estão em um mesmo nível e uma não é mais importante que outras**. Contudo, em certas modelagens, **é necessário representar uma classe**, que representa um item maior (o “todo”) que é formada por itens menores (as “partes”). Na agregação, um objeto é composto por (ou é parte de) outros objetos. **Em outras palavras, uma agregação é uma associação mais específica**;

Exemplo:

Neste exemplo, um **Cliente** tem um **Endereço**. Usamos agregação, pois as duas classes representam parte de um todo maior. Escolhemos também **modelar o Endereço como uma classe separada**, já que muitos outros tipos de itens também têm endereços também.



Composição: a composição é uma **forma de agregação**, uma **especialização da agregação**, **com propriedade bem definida e tempo coincidente como parte do todo**.

Na **agregação**, se a **instância** do todo **for removida**, **suas partes não serão necessariamente removidas**.

Já na **composição, se a instância do todo for removida, suas partes também deverão ser removidas.**

Agradeço atenção.

- Rafael Sacramento – rferfa@gmail.com
- Linkledin - https://www.linkedin.com/in/rafael-do-sacramento-bomfim-9150784b/
- Instagram - https://www.instagram.com/rafaelrfe/

“A GUARDA MORRE, MAS NÃO SE RENDE!”