



**UNIVERSIDADE PAULISTA**

**Especificação de Requisitos e Modelagem de um produto de  
software**

FELIPE SCHERER - RA D88HJE-1

JEHAN MARQUES MENDONÇA DIAS - RA N465CC-1

KELLY DENA - RA D912JB-8

VINÍCIUS GUIDI ROSSI - RA N386BH-8

TURMA CC7P12

ORIENTADOR: ELIO FILHO

CURSO: CIÊNCIA DA COMPUTAÇÃO - NOTURNO

CAMPINAS - SP

MAIO / 2022

<b>OBJETIVO</b>	<b>3</b>
<b>CONCEITOS GERAIS</b>	<b>5</b>
UML	5
RUP	8
<b>DOCUMENTO DE REQUISITOS</b>	<b>11</b>
Revisões Anteriores	11
1. INTRODUÇÃO	11
1.1. PROPÓSITO:	11
1.2. ESCOPO:	11
1.3. DEFINIÇÕES E SIGLAS:	11
1.4. REFERÊNCIAS:	12
1.5. VISÃO GERAL:	12
2. DESCRIÇÃO GERAL:	12
2.1. PERSPECTIVAS DO PRODUTO:	12
2.2. FUNÇÕES DO PRODUTO:	12
2.3. CARACTERÍSTICAS DO USUÁRIO:	13
2.4. RESTRIÇÕES:	13
2.5. SUPOSIÇÕES E DEPENDÊNCIAS	13
3. REQUISITOS ESPECÍFICOS:	13
3.1. REQUISITOS FUNCIONAIS:	13
3.2. REQUISITOS NÃO FUNCIONAIS:	14
<b>MODELAGEM DO SISTEMA</b>	<b>15</b>
Diagrama de caso de uso	15
Manter dados do cliente - Descrição de caso de uso	16
Manter dados de venda - Descrição de caso de uso	17
Manter dados do vendedor - Descrição de caso de uso	18
Relatório de venda dos produtos - Descrição de caso de uso	19
Diagrama de classes	21
<b>CONCLUSÃO</b>	<b>22</b>
<b>BIBLIOGRAFIA</b>	<b>23</b>
<b>FICHA DE ATIVIDADES PRÁTICAS SUPERVISIONADAS</b>	<b>25</b>

## OBJETIVO

---

Para desenvolvimento desse projeto temos como objetivo a modelagem de um site para ajudar microempreendedor que muita das vezes tem algo para vender, mas não sabe ao certo onde recorrer, a ideia do projeto é oferecer um site que posso ter grande facilidade de uso.

O site terá como seu principal objetivo a venda de produtos do ramo alimentício, pois muitas das vezes alguns produtos por falta de divulgação ou até mesmo esse suporte de venda é escasso, com o nosso site pretendemos abraçar toda essa comunidade com o objetivo de prover a venda dos produtos de uma forma facilitada e acessível para todos os tipos de público.

Para a construção dessa modelagem vamos fazer o uso de alguns conceitos que vão nos ajudar no futuro para termos um bom desenvolvimento do site, e para conseguirmos alcançar nosso objetivo com o maior sucesso possível, a ideia dessa modelagem é fazer com que tenhamos documentado o que deve ser feito, para que o processo de desenvolvimento seja o mais simples e linear possível, sem que tenha uma necessidade grande de a cada etapa ficamos reescrevendo os requisitos e tendo drásticas mudanças de percurso.

Para que seja possível atingir esse objetivo vamos fazer uma especificação de requisitos no padrão IEEE-830. Esse modelo é uma prática recomendada para especificações de exigências de software. Nesse documento descreve o conteúdo e a qualidade de uma boa especificação, além de apresentar exemplos de possíveis tipos de estruturas. Nele será possível encontrar dados sobre o público alvo e uma breve ideia do conceito que terá no site, terá alguns conceitos como os requisitos funcionais e não funcionais, características do usuário, as restrições do site, etc.

Além do modelo IEEE-830 vamos fazer o uso de diagramas de casos de uso com a descrição de caso de uso. Os diagramas servem para demonstrar possíveis formas do usuários usarem o sistema, para criar esses diagramas é usado uma linguagem única UML. Esses diagramas vão nos ajudar pois dão uma visão do relacionamento entre casos de uso, atores e sistemas. E para complementar os diagramas vamos também dar descrições dos casos de uso.

Para complementar o uso dos itens acima teremos também um diagrama de classes de entidade ele é uma abstração da vida real, que no caso é como os usuários vão se relacionar com o sistema no seu dia a dia. Nele é ilustrado graficamente a estrutura do software e como a estrutura está interligada

Com o uso desses conceitos pretendemos construir um software com maior agilidade, e uma mitigação de custos, pois uma construção de software sem uma caminho bem desenhado tende a cair em um limbo onde fica num ciclo infinito de criação de itens, ou modificação onde fica uma bagunça que no futuro pode trazer grandes prejuízos. Para termos a ideia fazer um requisito errado tende a ter um desperdício que custa 3 vezes mais que se tivéssemos feito certo da primeira vez.

Com o auxílio das técnicas citadas acima ficamos cada vez mais perto de obter sucesso no software. Pois assim como para se construir uma casa ou um prédio de qualidade, é essencial fazer um planejamento detalhado, com a finalidade de pensar sobre as formas de construção, fazer estimativas de tempo e material para a realização desse projeto. O desenvolvimento de um software de qualidade é semelhante a este processo, pois também se trata de uma questão de arquitetura e ferramentas.

Ao final de todo esse processo queremos ter um site com funcionalidades como por exemplo acesso diferenciado para vendedores e compradores, possível controle de estoque para o vendedor de produtos que estão à venda, controle de venda e emissão de relatório mensal de faturamento dentre outras funcionalidades que sejam realmente úteis, fáceis de usar e uma boa compatibilidade.

# CONCEITOS GERAIS

---

Para ter um bom software e conseguir atingir a conclusão do desenvolvimento do mesmo não é fácil, isso é claro, mas fazendo o uso de alguns conceitos podemos ter alguns caminhos que vão nos ajudar a chegar no produto final, para isso nesse projeto iremos explicar alguns conceitos que serão usados para termos mais agilidade na criação do mesmo.

Vale ressaltar que agilidade não é produzir software com pressa, é produzir software com velocidade, entregando valor no menor espaço de tempo possível, e melhorando isso continuamente, para ser ágil, é fundamental ser eficiente. Não é plausível falar em agilidade sem eficiência, com desperdício, em projetos de software, um dos maiores desafios é a boa comunicação e deixar claro o que se quer, o que será entregue, como será produzido etc. Isso não é simples quando produzimos software, devido à complexidade dessa atividade.

E neste contexto, fica claro que o uso racional de diagramas UML com o objetivo de transmitir ideias entre membros de um mesmo time, é uma ferramenta que favorece muito uma cultura ágil.

Além do conceito acima temos também o RUP é uma metodologia com práticas ágeis, assim como Scrum e o Extreme Programming (XP). Esses métodos possuem em comum a utilização de boas práticas que auxiliam na obtenção de uma rotina e técnicas produtivas.

## 1. UML

Seu nome significa '*Unified Modeling Language*', basicamente é uma linguagem de notação, que tem um jeito de escrever, ilustrar e comunicar, geralmente usado em projetos de sistemas.

Essa linguagem vem para nos ajudar no levantamento dos requisitos que irão construir um sistema, além de oferecer recursos de modelagem de estruturas que farão parte do mesmo. A UML tem grande aceitação no mercado também se deve a forte integração desça com conceitos de orientação a objetos. Já que muitos sistemas usam o conceito de orientação a objeto o uso das técnicas e abordagens do UML são realmente muito úteis. Para termos uma ideia sistemas

construídos nas mais variadas linguagens e plataformas como C#, JAVA, Delphi, etc. Podem se beneficiar das vantagens decorrentes do uso dessa linguagem.

O começo da UML como um dos principais meios para a documentação de sistemas aconteceu ainda no final dos anos 1990, graças ao trabalho conjunto de três especialistas da área de desenvolvimento de software: James Rumbaugh, Grady Booch e Ivar Jacobson. Diversos dos diagramas disponibilizados pela UML são resultado da evolução de representações propostas em momentos anteriores por estes especialistas, sendo que os mesmos procuram contemplar diferentes aspectos da construção de aplicações baseadas em técnicas da Orientação a Objetos.

É bastante comum que se encontrem literaturas a respeito de design patterns que fazem uso da UML, empregando esta última como um meio para a representação esquemática das ideias que estão sendo discutidas.

UML ajuda muito a deixar o escopo claro, pois centraliza numa única visão (o diagrama) um determinado conceito, utilizando uma linguagem que todos os envolvidos no projeto podem facilmente entender, mas ajuda desde que utilizada na medida certa, ou seja, apenas quando realmente é necessário. Pois, o seu uso de forma 'descontrolada' pode causar confusão ou um mau entendimento de requisitos.

Usar a UML de forma correta é essencial para seu bom desenvolvimento, pois será ela que fará parte da comunicação entre diversas pessoas envolvidas nesse processo como por exemplo analistas de negócio, product owner, scrum master, arquitetos, desenvolvedores, gerentes de projeto/produto e demais partes interessadas.

A UML tem diversas notações que podem ser utilizadas dependendo da situação. Algumas situações que podemos nos deparar é por exemplo:

- Como documentação que servirá de base para atividades de codificação das estruturas de um sistema, bem como elaboração de testes das funcionalidades implementadas;
- Para esboçar estruturas de um sistema em discussões a respeito do mesmo. Isto costuma acontecer de um modo informal, através do desenho de um componente ou processo da aplicação considerada, buscando assim um melhor entendimento daquilo que está analisando;

- Na documentação de estruturas já existentes de um sistema, ou seja, como uma ferramenta de engenharia reversa, a partir da qual serão documentadas funcionalidades e outras estruturas da aplicação em questão.

Além das diversas situações podemos dizer que temos também diversos tipo de diagramas

- **Diagramas estruturais:** priorizam a descrição fixa de estruturas de um sistema, como atributos, classes, operações além de prováveis relacionamento entre as estruturas. Dentro desta classe temos diversos diagramas que se enquadram nesta categoria:
  - Diagrama de classes: permite a visualização de um conjunto de classes, detalhando atributos e operações (métodos) presentes nesta última, assim como prováveis relacionamentos entre essas estruturas
  - Diagrama de componentes: mostrar diferentes componentes do sistema, além de possíveis furos. Pode referir-se a uma parte ou até mesmo um módulo da aplicação
  - Diagrama de pacotes: descreve as dependências entre diferentes pacotes que compõem uma aplicação, costumam conter classes, interface e outros elementos
  - Diagrama de objetos: apresenta o estado de instâncias de objetos dentro de um sistema, levando em conta um intervalo de tempo específico
  - Diagrama de estrutura composta: utilizado para demonstrar a estrutura interna de uma classe, incluindo referências que apontam para outras partes de um sistema.
  - Diagrama de instalação: demonstra estruturas de hardware adotada para a implantação em um ambiente
- **Diagramas comportamentais:** detalha o comportamento de partes do sistema ou processos de negócio relacionados a aplicação. Dentro desta classe temos diversos diagramas que se enquadram nesta categoria:
  - Diagrama de casos de uso: voltado à apresentação de funcionalidades e características de um sistema, assim como de que forma tais elementos se relacionam com usuários e entidades externas envolvidas num determinado processo.

- Diagrama de atividades: mostra as diversas tarefas desempenhadas na execução de uma atividade.
- Diagrama de transição de estado: detalha diferentes estados pelos quais um objeto pode passar
- **Diagrama de interação**: são considerados um subgrupo dos diagramas comportamentais, normalmente utilizados na representação de interação entre objetos e uma aplicação. Dentro desta classe temos diversos diagramas que se enquadram nesta categoria
  - Diagrama de sequência: demonstra interação entre diferentes objetos na execução de uma operação
  - Diagrama de interatividade: espécie de representação híbrida, com uma estrutura similar à de diagramas de atividade. O que diferencia este tipo de representação está justamente no fato do equivalente a uma atividade ser representada por outro diagrama, sendo o de sequência um exemplo de uso válido neste último caso.
  - Diagrama de tempo: corresponde a um tipo específico de diagrama de sequência, descrevendo mudanças de estado e interações entre objetos dentro de intervalos de tempo tomados como parâmetro.

Após os conceitos dos diagramas serem apresentados podemos concluir que a UML fornece ao nosso desenvolvimento diversas vantagens como a produção de software a ter maior eficiência e eficácia no dia a dia, possibilitando uma comunicação clara e objetiva sobre o que deve ser feito, e como deve ser feito.

Essa clareza possibilitada na comunicação pelo uso da UML diminui diretamente o desperdício tão comum na produção de software, desperdício este gerado pelo entendimento “torto” das coisas, e descoberta tardia deste desalinhamento.

## 2. RUP

O RUP também é um processo muito utilizado na engenharia de software, para apoiar o desenvolvimento orientado a objetos. Antes de falarmos sobre como ele funciona vamos saber primeiro o que significa o seu nome RUP vem de *‘Rational Unified Process’* que foi criado pela Rational Software Corporation.



O seu principal objetivo é atender as necessidades dos usuários garantindo uma produção de software de alta qualidade que cumpra o cronograma e um orçamento previsíveis. Nesse modelo é definido perfeitamente quem é responsável pelo que, como as coisas deverão funcionar e quando devem ser realizadas, descrevendo todas as metas de desenvolvimento.

O RUP organiza o desenvolvimento em quatro fases onde é tratado o planejamento, levantamento de requisitos, análise, implementação, teste e implantação de software. Cada uma dessas fases tem um papel importante para que o objetivo seja atingido. As quatro fases do RUP serão melhor definidas abaixo

- Fase de Concepção / Iniciação: nesta fase são feitas as tarefas de comunicação com o cliente e planejamento. É feito um plano avaliando riscos, estimativas de custos de prazo, estabelecendo as prioridades, levantamento de requisitos do sistema e preliminarmente analisá-lo. Assim terá uma aceitação das partes interessadas na definição do escopo do projeto, onde são examinados os objetivos para se decidir sobre a continuidade do desenvolvimento.
- Fase de Elaboração: nessa fase contém a modelagem do modelo genérico do processo, o objetivo é analisar de forma mais detalhada a análise domínio do problema, revisando os riscos que o projeto pode sofrer e a arquitetura do projeto começa a ter sua forma básica. Indagações como "O plano do projeto é confiável?", "Os custos são admissíveis?" são esclarecidas nesta etapa.
- Fase de Elaboração: nessa fase é desenvolvido ou adquirido os componentes do software, o principal objetivo é a construção do sistema com foco no desenvolvimento de componentes e outros recursos do sistema. É na fase de Construção que a maior parte de codificação ocorre.
- Fase de Transição: nessa fase é feita a entrega do software ao usuário e a fase de testes. O objetivo desta fase é disponibilizar o sistema, tornando-o disponível e compreendido pelo usuário final. As atividades desta fase incluem o treinamento dos usuários finais e também a realização de testes da versão beta do sistema visando garantir que o mesmo possua o nível adequado de qualidade.

Além das fases do RUP temos também algumas boas práticas a serem seguidas, essas práticas consistem em habilidades que precisam ser desenvolvidas para o incremento seja realizado da melhor maneira possível. O objetivo dessas práticas é gerenciar a qual a produção seja feita com qualidade, dentro dos prazos, com orçamentos

previsíveis e de forma satisfatória para as partes interessadas. Abaixo podemos ver algumas dessas boas práticas:

- Desenvolver o software iterativamente
- Gerenciar requisitos
- Usar arquiteturas baseadas em componentes
- Modelar software visualmente
- Verificar a qualidade do software, e
- Controlar as mudanças do software.

Com o uso das práticas citadas acima como UML e RUP podemos ter uma boa gerência e dos projetos garantindo assim qualidade no software produtividade no desenvolvimento, operação e manutenção de software, além de permitir controle sobre prazos e custos de desenvolvimento para atingir o nível de qualidade desejado.

# DOCUMENTO DE REQUISITOS

---

## Revisões Anteriores

Revisão	Comentário	Data
1	Versão inicial do documento de requisitos	30/04/2022
2	Adição dos requisitos específicos	07/05/2022

---

## 1. INTRODUÇÃO

### 1.1. PROPÓSITO:

O presente documento tem o objetivo de especificar e estabelecer os requisitos para o desenvolvimento de um aplicativo de vendas, empregando o gerenciamento de demandas.

### 1.2. ESCOPO:

Alguns microempreendedores estão com dificuldades de alcançar novos clientes, gerenciar controle de dados dos clientes e validação de compras. Com isso solicitou o desenvolvimento de um aplicativo de controle e vendas, bem como a melhoria na sua efetividade e organização de seus dados.

Este aplicativo deverá permitir o cadastro de clientes, vendedores, produtos e gerar vendas. Deverá emitir relatórios de dias com mais vendas e também gerar valor mensal por vendedor.

### 1.3. DEFINIÇÕES E SIGLAS:

**Clientes:** grupo de pessoas que vão acessar o aplicativo com intuito de comprar ou olhar os produtos que serão oferecidos.

**Vendedores:** grupo de pessoas responsável por cadastrar os produtos e fazer o acompanhamento da venda caso necessário, um vendedor pode vender um ou mais produtos.

**Produtos:** itens disponíveis para a venda no sistema que será disponibilizado, podendo conter apenas seu cadastro sem nenhum estoque ou estoque com vários produtos iguais.

**Vendas:** ato de um comprador acessar o aplicativo e comprar um produto disponibilizado

#### **1.4. REFERÊNCIAS:**

Não Aplicável.

#### **1.5. VISÃO GERAL:**

Não Aplicável.

## **2. DESCRIÇÃO GERAL:**

### **2.1. PERSPECTIVAS DO PRODUTO:**

O Delivery Food tem como objetivo ajudar microempreendedores a gerenciar seus clientes e assim gerar uma melhoria na organização da empresa e aumentar seus lucros.

O aplicativo deverá automatizar o cadastro de produtos e controlar a quantidade do mesmo, além de permitir o controle de clientes. Deverá também emitir relatórios de vendas de produtos mais vendidos e de faturamento mensal por vendedor. O Delivery Food é independente, e será utilizado em uma estrutura cliente/servidor.

Requisito de Software: o aplicativo será desenvolvido utilizando a Ferramenta de desenvolvimento JavaScript e o SGBD MySQL.

### **2.2.FUNÇÕES DO PRODUTO:**

Manter os dados dos produtos (Incluir, Alterar, Consultar)

Manter os dados dos vendedores (Incluir, Alterar, Consultar)

Manter os dados dos clientes(Incluir, Alterar, Consultar)

Efetuar venda de produtos (Incluir, Alterar, Consultar)

Efetuar controle de estoque dos produtos (Incluir, Alterar, Consultar)

Avaliar Produto e Vendedor

Emitir relatórios de venda de produtos, ordenado pelo mais vendido

Emitir relatórios de venda mensal total

### **2.3. CARACTERÍSTICAS DO USUÁRIO:**

Os usuários desse aplicativo serão os vendedores e devem possuir conhecimentos básicos de informática e avançado sobre os produtos disponíveis. E temos também os clientes que devem possuir conhecimento básico de informática não será necessário treinamento para o uso do sistema.

### **2.4. RESTRIÇÕES:**

O aplicativo deverá executar numa rede (cliente/servidor), nas seguintes versões dos navegadores; Google Chrome 85, FireFox 95; Ópera 80

### **2.5. SUPOSIÇÕES E DEPENDÊNCIAS**

Não Aplicável.

## **3. REQUISITOS ESPECÍFICOS:**

### **3.1. REQUISITOS FUNCIONAIS:**

Requisito funcional 1: O site deverá permitir cadastro de vendedor, contendo os campos nome, sobrenome, email, telefone, nome da loja, descrição da loja, endereço, senha.

Requisito funcional 2: O site deverá permitir cadastro de cliente contendo os campos nome, sobrenome, email, senha

Requisito funcional 3: O site deverá permitir que vendedores, possam fazer cadastro de produtos, contendo os seguintes itens, nome, descrição, foto, valor, possui controle de estoque (S/N), quantidade disponível

Requisito funcional 4: O site deverá permitir que o vendedor remova produto, deverá ser excluído um de cada vez

Requisito funcional 5: O site deverá ter uma tela inicial contendo informações de todos os vendedores ativos e com produtos em estoque, contendo o nome da loja pela proximidade da localização do cliente

Requisito funcional 6: O site deverá permitir uma visão geral do cardápio por vendedor, ao entrar no perfil do vendedor deve exibir uma lista dos produtos disponíveis mostrando o nome, foto (caso tenha), valor e descrição

Requisito funcional 7: O site deverá ter uma área exclusiva para vendedor, lá terá algumas funcionalidades extra como o controle de estoque, contendo nome do item e quantidade

Requisito funcional 8: O site deverá permitir a compra de produtos, o usuário deverá adicionar no carrinho e seguir para o pagamento onde deve ser confirmado, Endereço de entrega, tipo de pagamento(pix ou dinheiro) e exibir os produtos que estão sendo comprados

Requisito funcional 9: o site deverá ter uma tela inicial contendo os produtos disponíveis, nessa mesma tela deverá ter um botão de filtro, onde será possível buscar pelos tipos, por exemplo lanche, sushi, doce, salgado, etc.

Requisito funcional 10: O site deverá ter um botão de chat com o vendedor, onde será possível tirar as dúvidas, ou até mesmo fazer encomenda em larga escala.

Requisito funcional 11: O site deverá ter uma aba de perfil, onde será possível fazer atualização de dados colocados no cadastro.

Requisito funcional 12: O site deverá permitir que usuários que conversaram por chat com algum vendedor ou que compraram um produto possam fazer uma avaliação.

Requisito funcional 13: O site deverá ter uma aba de perfil exclusiva de vendedor, onde será possível fazer atualização de dados colocados no cadastro.

Requisito funcional 14: O site deverá permitir alteração de senha em caso de esquecimento, será necessário confirmar um código enviado para o email cadastrado

Requisito funcional 15: O site deverá ter autenticação em dois fatores, sendo assim no momento do login deverá enviar um código para o email do usuário

### **3.2. REQUISITOS NÃO FUNCIONAIS:**

Requisito não funcional 1: O site deve ser simples e acessível a todos os públicos;

Requisito não funcional 2: O site deverá apresentar resposta ao usuário em menos de 10 segundos;

Requisito não funcional 3: O site deverá executar em diferentes browsers;

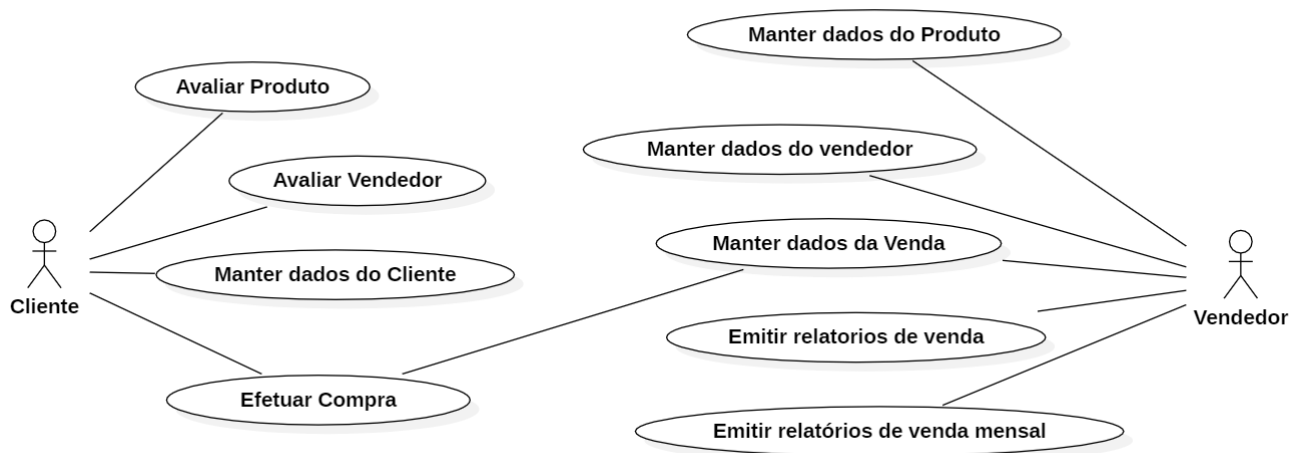
Requisito não funcional 4: O site deverá usar a biblioteca socket.io para o chat

Requisito não funcional 5: O site deverá tratar falhas sem perder informações, retornando de onde o usuário tinha parado. Por exemplo, caso dê falha ao fazer uma compra deve exibir uma mensagem de erro e possibilitar uma nova tentativa.

# MODELAGEM DO SISTEMA

---

## Diagrama de caso de uso



## Manter dados do cliente - Descrição de caso de uso

<b>Nome do caso de uso:</b>	Manter Dados do Cliente
<b>Ator principal:</b>	Cliente
<b>Ator Secundário:</b>	NA
<b>Descrição:</b>	Este caso de uso permite a manutenção dos dados (Inclusão, alteração e consulta) dos clientes do sistema
<b>Pré-Condições:</b>	Clientes realizar cadastro
<b>Pós-Condições:</b>	Dados dos clientes mantidos no sistema
<b>Fluxo Principal</b>	
<b>Ações do Ator:</b> 1. OSD permitir ao cliente iniciar o processo de manutenção de seus dados;  4. OSD permitir ao cliente escolher a opção desejada;	<b>Ações do Sistema:</b> 2. OSD apresentar ao cliente uma janela com seus dados atuais (Os obrigatórios na hora do cadastro e os não-obrigatórios)(E1, A1);  3. OSD apresentar ao cliente a opção de editar esses dados (A1);  5. O sistema executa o sub-fluxo correspondente ao tipo de operação recebida (Incluir, alterar, excluir ou consultar) (E2)
<b>Sub-Fluxo Consultar</b>	
<b>Ações do Ator:</b> 6. OSD permitir ao cliente acessar seu cadastro;	<b>Ações do sistema:</b> 7. OSD apresentar os dados do cliente em formato de formulário(E1, A1);
<b>Sub-Fluxo Incluir</b>	
<b>Ações do Ator:</b> 6. OSD permitir ao cliente realizar seu cadastro na plataforma  8. OSD permitir ao cliente confirmar se os dados estão corretos 9. OSD permitir ao cliente selecionar a opção Salvar.	<b>Ações do sistema:</b> 7. OSD apresentar os campos: nome, telefone, CPF, endereço, senha e email habilitados e vazios.(E1, A1, RN1, V1, V2, V3)  10. OSD incluir os dados do cliente. (E2) 11. OSD apresentar uma mensagem ao cliente confirmando a realização do cadastro. (A1)
<b>Sub-Fluxo Alterar</b>	
<b>Ações do Ator:</b> 6. OSD permitir ao cliente selecionar um campo de dado. 7. OSD permitir ao cliente selecionar a opção Alterar.  10.OSD permitir ao cliente alterar este dado 11. OSD permitir ao cliente selecionar a opção Salvar.	<b>Ações do sistema:</b>  8. OSD recuperar o dado selecionado do cliente.(E1, A1) 9. OSD apresentar o dado do cliente. (A1)  12. OSD alterar o dado do cliente.(E2)  13. OSD apresentar uma mensagem ao cliente confirmando a realização da operação. (A1);



<b>Fluxos de Exceção:</b>	E1. O sistema não consegue recuperar os dados de um cliente. OSD exibe mensagem de erro. E2. O sistema não consegue armazenar os dados de um cliente. OSD exibir mensagem de erro.
<b>Fluxos alternativos:</b>	A1. OSD permitir ao usuário cancelar a operação sem executar nenhuma ação.
<b>Regras de negócios:</b>	RN1: Os clientes não podem apresentar o mesmo CPF ou email.
<b>Validações:</b>	V1: O campo nome pode ter no máximo 75 caracteres, não podendo ser vazio V2: O campo endereço pode ter no máximo 150 caracteres, não podendo ser vazio. V3: Validação do formato do CPF

## Manter dados de venda - Descrição de caso de uso

<b>Nome do caso de uso:</b>	Manter Dados da Venda
<b>Ator principal:</b>	Vendedor
<b>Ator secundário:</b>	Cliente
<b>Descrição:</b>	Este caso de uso permite a manutenção da venda de um produto
<b>Pré-Condições:</b>	Cliente realizar uma compra
<b>Pós-Condições:</b>	Dados da venda mantidos no sistema, permitindo alterar, cancelar, editar e consultar a venda
<b>Fluxo Principal</b>	
<b>Ações do Ator:</b> 1. OSD permitir ao vendedor iniciar o processo de manutenção de venda.  3. OSD permitir ao vendedor escolher a opção desejada.	<b>Ações do Sistema:</b> 2. OSD apresentar ao vendedor uma janela de pesquisa de vendas (A1).  4. OSD apresenta a listagem dos dados da venda  5. O sistema executa o sub-fluxo correspondente ao tipo de operação recebida (Incluir, alterar ou consultar)
<b>Sub-Fluxo Consultar</b>	
<b>Ações do Ator:</b> 6. OSD permitir ao vendedor selecionar uma venda  7. OSD permitir ao vendedor selecionar a opção consultar.	<b>Ações do sistema:</b> 8. OSD recuperar os dados da venda selecionado: Nome, CPF e email do Cliente; Quantidade dos produtos comprados; Endereço de entrega; Verificação se o pagamento foi realizado (A1,E2).  9. OSD apresentar os dados da venda preenchidos e desabilitados ao usuário (A1).
<b>Sub-Fluxo excluir</b>	
<b>Ações do Ator:</b> 6. OSD permitir ao vendedor selecionar uma venda 7. OSD permitir ao vendedor selecionar a opção Excluir e realizar estorno.	<b>Ações do sistema:</b>  8. OSD recuperar os dados da venda selecionado: Nome, CPF e email do Cliente; Quantidade dos produtos

	comprados; Endereço de entrega; Verificação se o pagamento foi realizado (A1,E2). 9. OSD apresentar a opção de estorno e exclusão da venda 10. OSD deve apresentar a confirmação da exclusão de venda e estorno. (E3)
<b>Fluxos de Exceção:</b>	E1. O sistema não consegue recuperar os dados das vendas . OSD exibir mensagem de erro. E2. O sistema não consegue recuperar os dados de uma venda. OSD exibir mensagem de erro. E3. O sistema não consegue excluir os dados de uma venda. OSD exibir mensagem de erro.
<b>Fluxos alternativos:</b>	A1. OSD permitir ao usuário cancelar a operação sem executar nenhuma ação.
<b>Regras de negócio:</b>	NA
<b>Validações:</b>	NA

## Manter dados do vendedor - Descrição de caso de uso

<b>Nome do caso de uso:</b>	Manter dados vendedor
<b>Ator principal:</b>	Vendedor
<b>Ator Secundário:</b>	Cliente
<b>Descrição:</b>	Este caso permite que o vendedor altere suas informações
<b>Pré-Condições:</b>	NA
<b>Pós-Condições:</b>	Dados do vendedor mantidos no sistema
<b>Fluxo Principal</b>	
<b>Ações do Ator:</b> 1. OSD permite ao vendedor a opção de incluir, alterar e consultar.  3. OSD permite ao vendedor escolher a opção que deseja.	<b>Ações do Sistema:</b>  2. OSD apresentar ao vendedor uma janela com as opções que ele pode escolher.  4. O sistema executa o subfluxo correspondente ao tipo de operação recebida (Incluir, Alterar, Consultar)
<b>Sub-Fluxo Consultar</b>	
<b>Ações do Ator:</b> 5. OSD permitir ao vendedor acessar seu cadastro;	<b>Ações do sistema:</b> 6. OSD apresentar os dados do vendedor em formato de formulário(E1, A1);
<b>Sub-Fluxo Incluir</b>	
<b>Ações do Ator:</b> 7. Permite ao vendedor selecionar a opção adicionar 8. ODS permite ao vendedor informar os dados 9. ODS permite ao vendedor salvar as alterações	<b>Ações do sistema:</b>  10. ODS exibe os campos nome, CNPJ, endereço (A1) 11. ODS salva os dados do vendedor (E2, RN1, V1,V2,V3, V4)

	12. ODS exibe ao vendedor uma mensagem de confirmação
<b>Sub-Fluxo Alterar</b>	
<b>Ações do Ator:</b> 5. OSD permite ao vendedor selecionar um campo. 6. OSD permite que o vendedor selecione a opção alterar  8. OSD Permite ao vendedor alterar os dados 9. OSD Permite ao vendedor salvar as alterações	<b>Ações do sistema:</b>  7. Retorna as informações do vendedor: nome, CNPJ, endereço (A1, E1)  10. OSD altera os dados do vendedor (E2, RN1, V1,V2, V3, V4) 11. Exibe ao vendedor uma mensagem de confirmação.
<b>Fluxos de Exceção:</b>	E1. O sistema não consegue retornar os dados do vendedor. OSD exibir mensagem de erro. E2. O sistema não consegue armazenar os dados de um vendedor. OSD exibe uma mensagem de erro.
<b>Fluxos alternativos:</b>	A1. OSD permitir ao usuário cancelar a operação sem executar nenhuma ação.
<b>Regras de negócios:</b>	RN1: Os vendedores não podem apresentar o mesmo CNPJ.
<b>Validações:</b>	V1: O campo nome pode ter no máximo 75 caracteres, não podendo ser vazio. V2: O campo endereço pode ter no máximo 150 caracteres, não podendo ser vazio. V3: Validação do formato do CNPJ

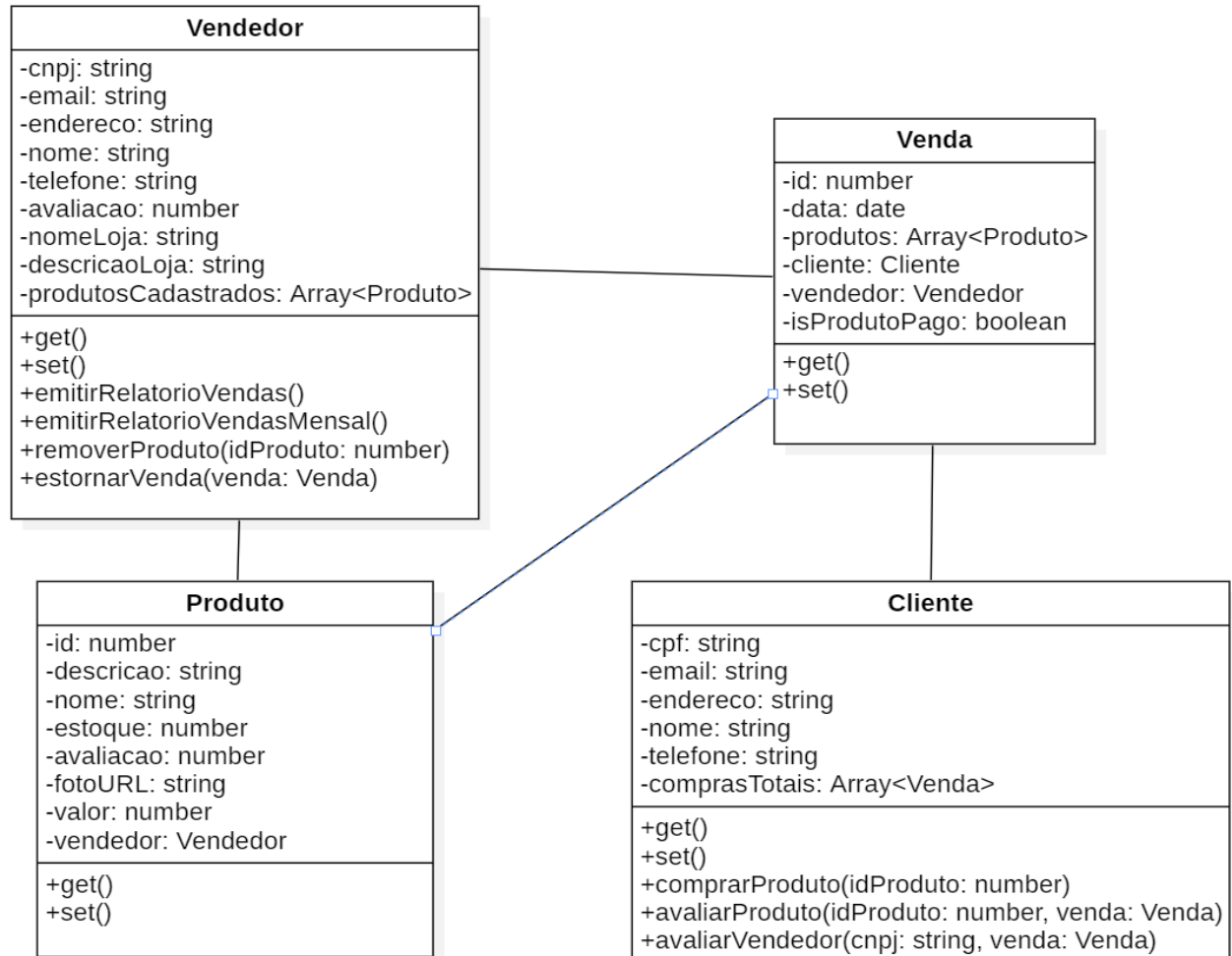
## Relatório de venda dos produtos - Descrição de caso de uso

<b>Nome do caso de uso:</b>	Relatório de venda do produto
<b>Ator principal:</b>	Vendedor
<b>Ator Secundário:</b>	Cliente
<b>Descrição:</b>	Este caso de uso permite que o vendedor possa elaborar relatório e visualizar relatório de venda
<b>Pré-Condições:</b>	NA
<b>Pós-Condições:</b>	Dados de vendas devem estar contidos no sistema e deve ser feito uma associação com a data das vendas.
<b>Fluxo Principal</b>	
<b>Ações do Ator:</b> 1. OSD permite que o vendedor inicie o processo de visualização ou elaboração de relatório  3. OSD permite que o vendedor escolhas as opções de visualizar ou criar relatório	<b>Ações do Sistema:</b>  2. Exibe ao vendedor uma janela com a opção de visualizar relatórios existentes ou criar relatórios 4. OSD executa o subfluxo correspondente à opção escolhida
<b>Sub-Fluxo visualizar</b>	
<b>Ações do Ator:</b> 5. OSD exibe relatórios existentes. 6. OSD permite que o usuário escolha uma opção	<b>Ações do sistema:</b>  7. OSD exibe o relatório que o cliente solicitou (E1, RN1)
<b>Sub-Fluxo criar</b>	
<b>Ações do Ator:</b>	<b>Ações do sistema:</b>

<p>5. OSD permite que o vendedor inicie o processo de criação de relatório</p> <p>7. OSD permite que o vendedor selecione um período de vendas</p> <p>8. OSD permite que o vendedor crie o relatório</p>	<p>6. OSD apresenta ao vendedor em uma janela vendas e as datas que foram realizadas (A1)</p> <p>9. OSD elabora o relatório de acordo com a data selecionada pelo vendedor (E2, RN1, V2)</p> <p>10. OSD salva o relatório.</p>
<b>Fluxos de Exceção:</b>	<p>E1. O sistema não consegue trazer as informações solicitadas e exibe uma mensagem de erro.</p> <p>E2. O sistema não consegue encontrar as datas selecionadas pelo vendedor e exibe uma mensagem de erro.</p>
<b>Fluxos alternativos:</b>	<p>A1. OSD permitir ao usuário cancelar a operação sem executar nenhuma ação.</p>
<b>Regras de negócios:</b>	<p>RN1: Os vendedores não podem adicionar uma data inexistente</p>
<b>Validações:</b>	<p>V2: O campo data deve ser no formato (dd/mm/aaaa)</p>

## Diagrama de classes

---



## CONCLUSÃO

---

Ao final desse trabalho podemos perceber que construir um software pode ser muito mais difícil do que parece. Para construir um bom software não basta apenas ter a ideia e sair desenvolvendo, muito pelo contrário requer muito planejamento e dedicação.

Com os métodos apresentados durante esse trabalho podemos ter uma ideia dos passos que podem ser seguidos durante o processo de criação, planejamento e desenvolvimento. Percebemos que cada etapa que formos passar requer muita atenção e cuidado aos detalhes para que tenhamos uma entrega final dentro do prazo e custos previamente estipulados. Pois a maioria dos softwares atualmente dificilmente conseguem atingir as metas definidas no começo.

Percebemos que para atingir as tão sonhadas metas do começo precisamos fazer o uso de metodologias e conceitos que no início podem até mesmo parecer perca tempo, porém com o tempo vão se mostrando totalmente necessário, por exemplo o RUP a utilização desse processo é muito benéfico para o desenvolvimento do projeto pois garante a qualidade do software, produtividade no desenvolvimento, operação e manutenção, além de permitir controle sobre o desenvolvimento dentro de custos, prazos e níveis de qualidade. Sem esse processo com certeza não teríamos tantas vantagens e com certeza o desenvolvimento do software tende a cair em um limbo de manutenções infinitas, prazos sempre atrasados e custos cada vez maiores.

Além do RUP temos também os diagramas UML que no caso do projeto, que no caso do nosso projeto foram feitos o do caso de uso e de classes. Os diagramas UML acabam dividindo a indústria de engenharia de software, pois apesar de ter inúmeras desvantagens tem também suas vantagens que quando colocamos na balança pode ser que dependendo do projeto a necessidade do uso de UML acaba ficando inquestionável. Dentre as suas vantagens temos comunicação mais eficaz, menos trabalho na implementação, pois com as especificações documentadas a solução já está pronta, o tempo gasto com o desenvolvimento, testes de unidade e automatizados são menores, etc.

Com todos esses fatos apresentados ao longo do projeto podemos perceber que para um projeto médio a longo o uso de metodologias é indispensável para que ao final um bom software possa ser entregue.

## BIBLIOGRAFIA

---

**Lucidchart. Diagrama de caso de uso UML: O que é, como fazer e exemplos.**

Lucidchart. Disponível em: <https://www.lucidchart.com/pages/pt/diagrama-de-caso-de-uso-uml>. Acesso em: 10/04/2022.

**Ventura, Plínio. Entendendo o Diagrama de Classes da UML.** Até o momento.

Disponível em: <https://www.ateomomento.com.br/uml-diagrama-de-classes/>. Acesso em: 10/04/2022.

**Camarini, Evandro. A importância do Modelagem de Objetos no Desenvolvimento de Sistemas.** Disponível em:

<http://www.linhadecodigo.com.br/artigo/1293/a-importancia-do-modelagem-de-objetos-no-desenvolvimento-de-sistemas.aspx>. Acesso em: 16/04/2022.

**Emanoele, Alícia. Como funciona o processo chamado RUP e quais são as 4 fases desse método.** Voitto. Disponível em: <https://www.voitto.com.br/blog/artigo/o-que-e-rup>. Acesso em: 16/04/2022.

**Emanoele, Alícia. Como funciona o processo chamado RUP e quais são as 4 fases desse método.** Voitto. Disponível em: <https://www.voitto.com.br/blog/artigo/o-que-e-rup>. Acesso em: 16/04/2022.

**Devmedia. Gestão de projetos com RUP.** Devmedia. Disponível em: <https://www.devmedia.com.br/gestao-de-projetos-com-rup/39332>. Acesso em: 16/04/2022.

**Guedes, Marylene. O que é RUP - Rational Unified Process?.** Treinaweb. Disponível em: <https://www.treinaweb.com.br/blog/o-que-e-rup-rational-unified-process>. Acesso em: 29/04/2022.

**Devmedia. Modelagem de sistemas através de UML: uma visão geral.** Devmedia. Disponível em:

<https://www.devmedia.com.br/modelagem-de-sistemas-atraves-de-uml-uma-visao-geral/27913>. Acesso em: 29/04/2022.

**Creately. Tutorial do diagrama de caso de uso.** Creately. Disponível em: <https://creately.com/blog/pt/diagrama/tutorial-de-diagrama-de-caso-de-uso/>. Acesso em: 29/04/2022.



FELIPE SCHERER  
RA: D88HJE-1

UNIP

UNIVERSIDADE PAULISTA

FICHA DAS ATIVIDADES PRÁTICAS SUPERVISIONADAS - APS

NOME: Felipe Scheer

TURMA: CC7P12

RA: D88HJE-1

CURSO: Ciência da Computação

CAMPUS: Self-Complex

SEMESTRE: 7º

TURNO: Noturno

CÓDIGO DA ATIVIDADE: 77B4

SEMESTRE: 7º

ANO GRADE: 4º

DATA DA ATIVIDADE	DESCRIÇÃO DA ATIVIDADE	TOTAL DE HORAS	ASSINATURA DO ALUNO	HORAS ATRIBUÍDAS (1)	ASSINATURA DO PROFESSOR
09/04	Reunião de grupo	7	Felipe	7	
10/04	Resposta	8	Felipe	8	
16/04	Índio do desenvolvimento da APS	10	Felipe	10	
17/04	Pesquisa sobre dia gramas	6	Felipe	6	
23/04	Desenvolvimento ITCE 830	5	Felipe	5	
24/04	Continuação do documento ITCE 830	5	Felipe	5	
29/04	Pesquisa o tema	3	Felipe	3	
30/04	Aplicação da pesquisa sobre o tema	7	Felipe	7	
01/05	Descrição de caso de uso	9	Felipe	9	
04/05	Diagrama de caso de uso	3	Felipe	3	
07/05	Continuação do diagrama de caso de uso	6	Felipe	6	
13/05	Desenvolvimento dos conceitos gerais	2	Felipe	2	
14/05	Revisão dos diagramas	2	Felipe	2	
15/05	Finalização da APS	2	Felipe	2	
(1) Horas atribuídas de acordo com o regulamento das Atividades Práticas Supervisionadas do curso.		TOTAL DE HORAS ATRIBUÍDAS:		<u>75</u>	

AVALIAÇÃO: \_\_\_\_\_

Aprovado ou Reprovado


NOTA: \_\_\_\_\_

DATA: \_\_\_\_/\_\_\_\_/\_\_\_\_

CARIMBO E ASSINATURA DO COORDENADOR DO CURSO

JEHAN MARQUES MENDONÇA DIAS

RA: N465CC-1

		FICHA DAS ATIVIDADES PRÁTICAS SUPERVISIONADAS - APS			
UNIVERSIDADE PAULISTA					
NOME: <u>Jeihan Marques Mendonça Dias</u>		TURMA: <u>CC7P12</u>			
CURSO: <u>Ciência da Computação</u>		RA: <u>N465CC-1</u>			
CAMPUS: <u>Swift - Campinas</u>		SEMESTRE: <u>7º</u>			
CÓDIGO DA ATIVIDADE: <u>77B4</u>		ANO GRADE: <u>4º</u>			
SEMESTRE: <u>7º</u>		TURNO: <u>Noturno</u>			
DATA DA ATIVIDADE	DESCRIÇÃO DA ATIVIDADE	TOTAL DE HORAS	ASSINATURA DO ALUNO	HORAS ATRIBUÍDAS (1)	ASSINATURA DO PROFESSOR
09/04	Reunião em grupo	7	Jeihan	7	
10/04	Pesquisa	8	Jeihan	8	
16/04	Início do desenvolvimento da APS	10	Jeihan	10	
17/04	Pesquisa sobre diagramas	6	Jeihan	6	
23/04	Documento IEEE 830	5	Jeihan	5	
24/04	Continuação do documento IEEE 830	5	Jeihan	5	
29/04	Pesquisa do tema	3	Jeihan	3	
30/04	Aplicação da pesquisa sobre o Tema	7	Jeihan	7	
01/05	Descrição de casos de uso	9	Jeihan	9	
04/05	Diagrama de casos de uso	3	Jeihan	3	
07/05	Continuação do diagrama de casos de uso	6	Jeihan	6	
13/05	Desenvolvimento dos conceitos gerais	2	Jeihan	2	
14/05	Revisão dos diagramas	2	Jeihan	2	
15/05	Finalização da APS	2	Jeihan	2	
(1) Horas atribuídas de acordo com o regulamento das Atividades Práticas Supervisionadas do curso.		TOTAL DE HORAS ATRIBUÍDAS: <u>75</u>			
AVALIAÇÃO: _____		Aprovado ou Reprovado			
NOTA: _____					
DATA: ____/____/____					
CARIMBO E ASSINATURA DO COORDENADOR DO CURSO					



KELLY DENA  
RA: D912JB-8

<b>UNIP</b> UNIVERSIDADE PAULISTA		FICHA DAS ATIVIDADES PRÁTICAS SUPERVISIONADAS - APS			
NOME: <u>Kelly Regina Dena da Silva</u>		TURMA: <u>CC3P1A</u>		RA: <u>D912JB-8</u>	
CURSO: <u>Ciência da Computação</u>		CAMPUS: <u>Amilinas - Suíte 7</u>		SEMENTRE: <u>7º</u> TURNO: <u>Noturno</u>	
CÓDIGO DA ATIVIDADE: <u>7784</u>		SEMENTRE: <u>7º</u>		ANO GRADE: <u>2022</u>	
DATA DA ATIVIDADE	DESCRIÇÃO DA ATIVIDADE	TOTAL DE HORAS	ASSINATURA DO ALUNO	HORAS ATRIBUÍDAS (1)	ASSINATURA DO PROFESSOR
09/04/2022	Discussão sobre o tema com o grupo	7	Kelly Dena	7	
10/04/2022	Pesquisa sobre o tema	8	Kelly Dena	8	
16/04/2022	Trabalho de desenvolvimento de APP	10	Kelly Dena	10	
17/04/2022	Pesquisa sobre diagrama	6	Kelly Dena	6	
23/04/2022	Documento IEEE-830	5	Kelly Dena	5	
24/04/2022	Continuação do documento IEEE-830	5	Kelly Dena	5	
29/04/2022	Pesquisa sobre o tema	3	Kelly Dena	3	
30/04/2022	Aplicação a pesquisa sobre o tema	7	Kelly Dena	7	
01/05/2022	Diagrama de caso de uso	9	Kelly Dena	9	
04/05/2022	Diagrama de caso de uso	3	Kelly Dena	3	
07/05/2022	Continuação dos diagramas	6	Kelly Dena	6	
13/05/2022	Pesquisando os conceitos gerais	2	Kelly Dena	2	
14/05/2022	Pesquisa dos diagramas	2	Kelly Dena	2	
15/05/2022	Finalização	2	Kelly Dena	2	
(1) Horas atribuídas de acordo com o regulamento das Atividades Práticas Supervisionadas do curso.		TOTAL DE HORAS ATRIBUÍDAS: <u>75</u>			
AVALIAÇÃO: _____		Aprovado ou Reprovado			
NOTA: _____					
DATA: ____/____/____					
CARIMBO E ASSINATURA DO COORDENADOR DO CURSO					



UNIVERSIDADE PAULISTA

## FICHA DAS ATIVIDADES PRÁTICAS SUPERVISIONADAS - APS

NOME: Vinícius Guidi RossiTURMA: CC7P12RA: N386BH-8

CURSO: \_\_\_\_\_

CAMPUS: SumaréSEMESTRE: 7ºTURNO: matutinoCÓDIGO DA ATIVIDADE: 7788SEMESTRE: 7º

ANO GRADE: \_\_\_\_\_

4º

DATA DA ATIVIDADE	DESCRIÇÃO DA ATIVIDADE	TOTAL DE HORAS	ASSINATURA DO ALUNO	HORAS ATRIBUÍDAS (1)	ASSINATURA DO PROFESSOR
09/04	Reunião de grupo	7	Vinícius Rossi	7	
10/04	Requisitos	8	Vinícius Rossi	8	
16/04	Desenvolvimento da APS	10	Vinícius Rossi	10	
17/04	Requisitos sobre caso de lupa	6	Vinícius Rossi	6	
23/04	Desenvolvimento IEE830	5	Vinícius Rossi	5	
24/04	Desenvolvimento de documento IEE830	5	Vinícius Rossi	5	
29/04	Requisitos sobre diagrama de fluxo	3	Vinícius Rossi	3	
30/04	Desenvolvimento do diagrama de fluxo	7	Vinícius Rossi	7	
01/05	Desenvolvimento do caso de uso	9	Vinícius Rossi	9	
04/05	Desenvolvimento de caso de uso	3	Vinícius Rossi	3	
07/05	Desenvolvimento de caso de uso	6	Vinícius Rossi	6	
13/05	Desenvolvimento de caso de uso	2	Vinícius Rossi	2	
14/05	Revisão das diagramas	2	Vinícius Rossi	2	
15/05	Análise das APS	2	Vinícius Rossi	2	
(1) Horas atribuídas de acordo com o regulamento das Atividades Práticas Supervisionadas do curso.					
TOTAL DE HORAS ATRIBUÍDAS:				<u>75</u>	

AVALIAÇÃO: \_\_\_\_\_

Aprovado ou Reprovado

NOTA: \_\_\_\_\_

DATA: \_\_\_\_/\_\_\_\_/\_\_\_\_

CARIMBO E ASSINATURA DO COORDENADOR DO CURSO \_\_\_\_\_

VINÍCIUS GUIDI ROSSI

RA: N386BH-8