Análise e Desenvolvimento de Sistemas

ENGENHARIA DE SOFTWARE ÁGIL

RELATÓRIO DE AULAS PRÁTICAS

Nome:Vitória Freitas

RA:2523279

Polo de matrícula:Carapicuíba - Vila Dirce

Local da realização da Aula Prática:Santana de Parnáiba

Ano da postagem:2025

Instituto de Ciências Exatas e Tecnologia	**Disciplina:** Análise e Projeto de Sistemas **Título da Aula:** Engenharia de Software	**Relatório 1**

1. Resumo Teórico

O que é engenharia de Software? É uma área que trabalha com a tecnologia, utilizando seus conhecimentos de engenharia, arquitetura e desenvolvimento para criação, projeção e gerenciamento de sistemas de diversos tipos para diversas áreas do mercado de trabalho.

Diferença de análise de software e de um sistema de software. Análise de software é buscar e entender as necessidades de um cliente em relação a construção de um sistema, o que ele quer que o sistema faça e a performance dele. Um sistema de software é um programa, feito para executar o hardware e os aplicativos do computador, é como se fosse a interface entre o hardware e os aplicativos do usuário, fornecendo uma plataforma para que o software possa ser executado. Um exemplo de sistema de software é o Sistema Operacional, como Windows ou Linux.

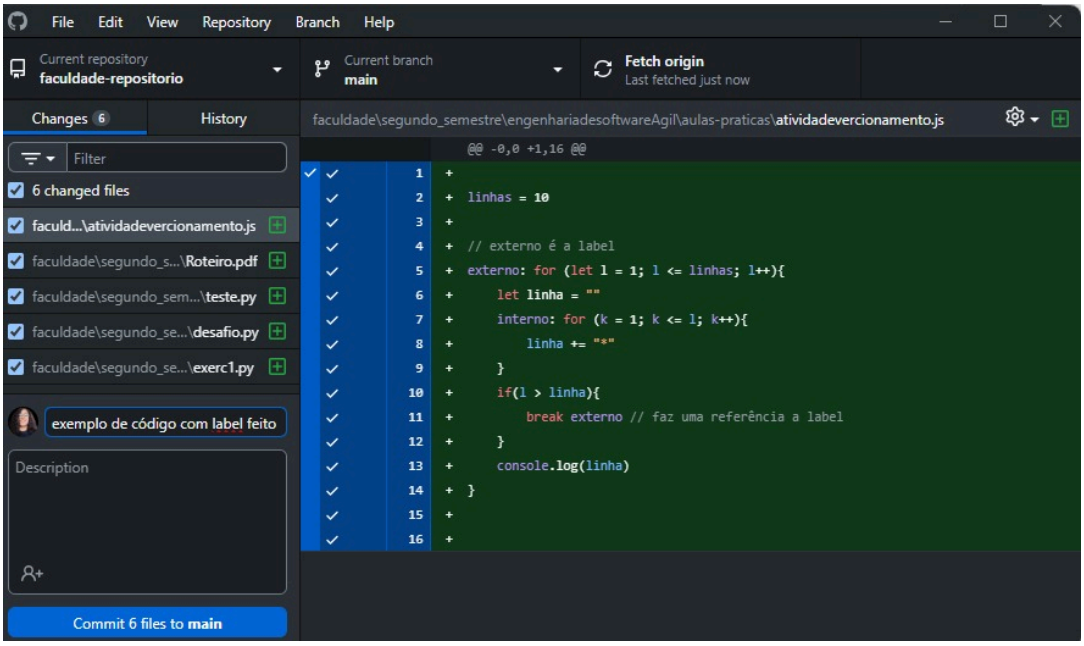
2. Estudo de Caso

O Linux, é um Sistema Operacional criado por Linus Torvalds em 1991. Hoje está presente em diversos tipos de sistemas e dispositivos. Ele é open source (dar o acesso para pessoas fazerem mudanças e estudar o seu código-fonte). Foi desenvolvido para ser parecido com o Unix. Todo Linux possui o seu Kernel, que tem como função fazer o gerenciamento dos recursos de hardware.

Mudanças que ocorreram durante sua história foram a criação de 321 distribuições, que hoje são monitoradas pelo DistroWatch, como Mint, Ubuntu e Fedora.

3. Diagrama do versionamento

Utilizei o GitHub Desktop que é um aplicativo de código aberto, que facilita o trabalho com códigos hospedados no GitHub, e outros serviços de Hospedagem Git. Abaixo está uma imagem de um commit dentro do GitHub Desktop



4. Reflexões Finais

A importância da engenharia de software hoje, dentro da minha perspectiva, está na construção de diversos tipos sistemas, tendo seu funcionamento funcional e com boas performances, sendo muito bem planejados. As principais reclamações de usuários podem estar relacionadas a sua velocidade de resposta, por exemplo quando um site demora muito para mostrar os produtos em seu catálogo, e quando o sistema não possui uma interface intuitiva, deixando o usuário perdido na hora de utilizar os serviços oferecidos pelo sistema.

Nessa aula aprendemos sobre a engenharia de software, sobre sistemas de software, análises que têm que serem feitas antes da construção de um sistema.

5.Referências

https://faculdade.grancursosonline.com.br/blog/engenharia-de-software/#o-que

https://blog.casadodesenvolvedor.com.br/analise-de-requisitos-de-software/

https://www.redhat.com/pt-br/topics/linux/what-is-linux

https://www.techtudo.com.br/noticias/2012/05/a-evolucao-do-linux.ghtm

https://docs.github.com/pt/desktop/overview/getting-started-with-github-desktop

Instituto de Ciências Exatas e Tecnologia	**Disciplina:** Análise e Projeto de Sistemas **Título da Aula:** Engenharia de Requisitos	**Relatório 2**

1. Resumo Teórico

Engenharia de requisitos, disciplina que vai definir, documentar e manter os requisitos de um sistema de software. Entender as necessidades do cliente para a construção de um sistema com uma boa qualidade.

RS engloba todos os requisitos do sistema, RU são os requisitos do usuário, os desejos dele, como sistema deve ser, o que ele irá fazer, suas funcionalidades, design. Depois vão virar RF e RNF. RF são os requisitos funcionais, o que o sistema deve fazer, por exemplo o sistema deve ter um painel administrativo, deve mostrar o estoque. RNF são critérios que qualificam os requisitos funcionais, como performance, usabilidades, robustez, segurança.

2. Estudo de Caso

Sistema de software ERP, um sistema que tem como objetivo a otimização dos principais processos de negócio de uma empresa, como finanças, RH, produção, cadeia de suprimentos e vendas.

Nas reuniões com os clientes e desenvolvedores foram definidos os requisitos funcionais e não funcionais, abaixo estão eles.

Requisitos funcionais: o sistema deve ter uma área exclusiva para finanças, outra para o RH, uma para mostrar o estoque disponível e as vendas da empresa.

Requisitos não funcionais: o sistema terá criptografia para segurança dos dados, tem que ter uma boa velocidade para mostrar os produtos, no máximo 1 segundo de espera.

Ferramenta para o desenvolvimento, backend vai ser desenvolvido utilizando a linguagem de programação Java, por sua robustez e segurança, framework Spring Boot. Front end irá usar React Js, para facilitar a navegação entre páginas, banco de dados o MySql.

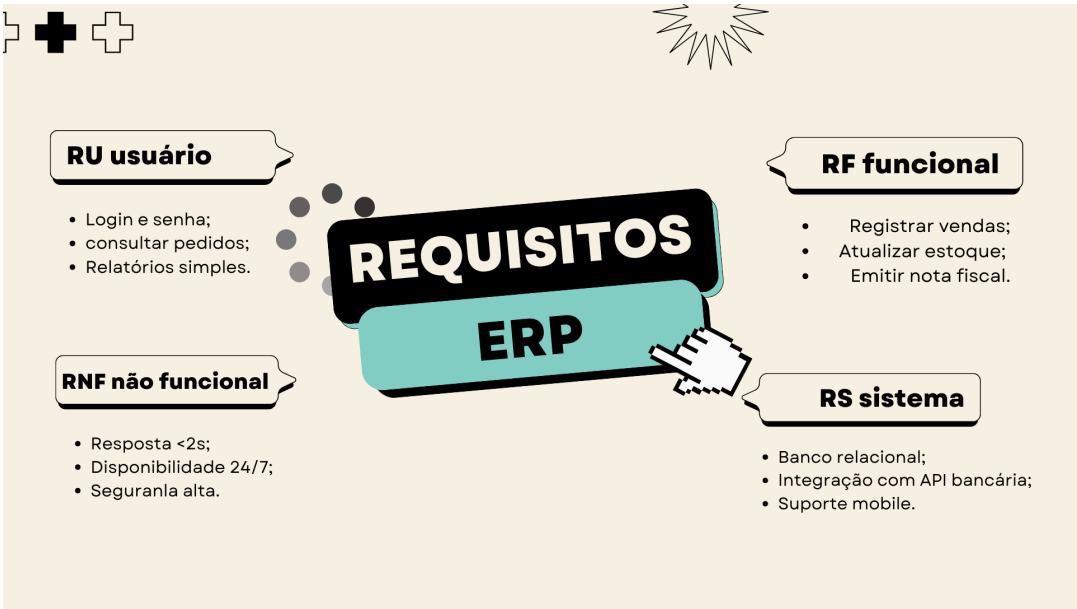
3. Diagrama de casos de uso

Logo abaixo estão diagrama de caso de uso e uma tabela dos requisitos que foram feitos em grupo.

(Créditos: autoria própria)

Código	Tipo	Descrição	Prioridade	Status
RU01	Usuário	usuário deve acessar o sistema com login e senha	Alta	Em análise
RU02	Funcional	sistema deve registrar vendas e atualizar o estoque automaticamente	Alta	Validado
RU03	Não Funcional	O sistema deve processar consultas em até 2 segundos	Média	Validado
RU04	Sistema	O sistema deve armazenar dados em banco de dados relacional (MySQL)	Alta	Em Análise

4. Diagrama dos requisitos com mapa mental



(Créditos: autoria própria)

5. Reflexões finais

A importância de casos de uso e mapa mental, está na melhora do entendimento do que é para ser feito no desenvolvimento do sistema, assim deixando claro o que o sistema deve fazer e como deve se comportar.

Validação dos requisitos deve ser feita, se aquele requisito é viável para o desenvolvimento daquele sistema, por exemplo, se ele se encaixa no orçamento do sistema que está sendo feito.

6. Referências

https://www.devmedia.com.br/introducao-a-requisitos-de-software/29580

https://www.sap.com/brazil/products/erp/what-is-erp.html

Instituto de Ciências Exatas e Tecnologia	**Disciplina:** Análise e Projeto de Sistemas **Título da Aula:** Processos de Software	**Relatório 3**

1. Resumo Teórico

Processo de software, conjunto de atividades que vão levar a produção de um produto de software, são procedimentos com um propósito de arquitetar, desenvolver, testar e implantar o software.

Modelos de processos de software. Modelos Prescritivos, foram feitos para trazer uma estrutura inicial aos processos e a maneira como eles se inter-relacionam, dentro dele há modelo cascata, Modelo Incremental, Modelo Evolucionário e Modelos Concorrentes.

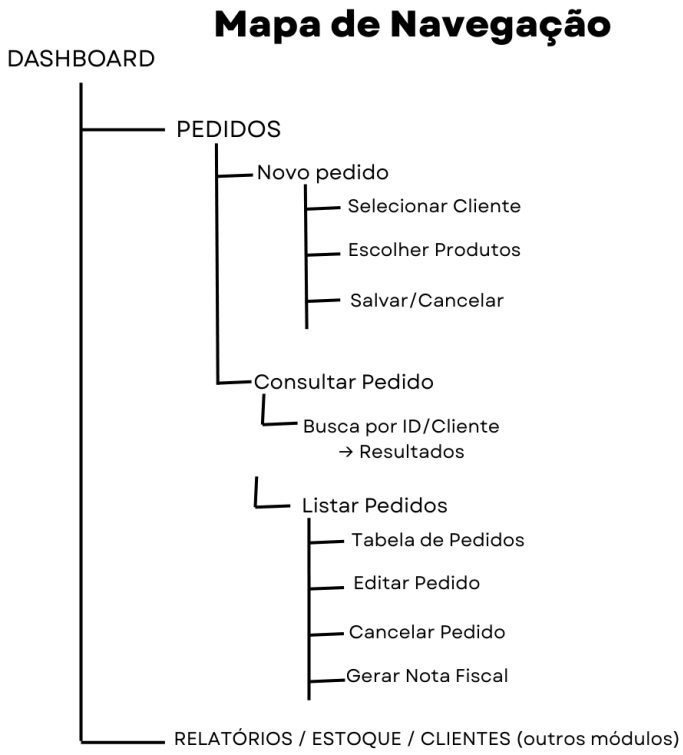
Dentro do sistema de software ERP, as principais operações do usuário seriam: o acesso a finanças da empresa, acesso ao RH e ao estoque de produtos.

2. Estudo de caso

Funcionalidade escolhida: Estoque de produtos do sistema ERP. Modelo escolhido: Incremental

Para criar as funcionalidades financeiras, iremos deixar claro os requisitos iniciais bem definidos, o que o cliente quer que tenha na parte estoque, isso inclui suas ações e como o sistema irá agir em relação a sua performance.

3. Mapa de navegação de funcionalidades



4. Diagrama de atividades

```
[Início]
   ↓
(Planejar objetivos e requisitos)
   ↓
(Analisar riscos e alternativas)
   ↓
[Decisão: riscos aceitáveis?]
   ├── Não → (Revisar requisitos/estratégia) → volta para "Planejar"
   └── Sim → segue
   ↓
(Desenvolver protótipo / incremento)
```

5. Reflexões finais

Para construção de uma funcionalidade precisa entender o que o cliente precisa em seu sistema, suas ações e como ele deve agir.

Mapa de navegação importante para um melhor entendimento de como o sistema irá funcionar.

O que determinou a escolha de um modelo de processo, foi para ter um melhor entendimento de como irá funcionar o desenvolvimento do sistema, assim deixando bem documentado desde do início.

6. Referências

https://portal.ifto.edu.br/ifto/reitoria/diretoria-sistemica/tecnologia-da-informacao/documentos/processos/processo-de-gestao-de-software

https://www.devmedia.com.br/introducao-aos-processos-de-software-e-o-modelo-incremental-e-evolucionario/29839

Instituto de Ciências Exatas e Tecnologia	**Disciplina:** Análise e Projeto de Sistemas **Título da Aula:** Planejamento do Processo de Software	**Relatório 4**

1. Resumo Teórico

Escreva aqui um texto de 8 a 10 linhas abordando:
- O que é componentização do software.
- Principais diagramas a serem utilizados.
- Como funciona um sistema distribuído.
- Principais operações dos usuários.

2. Estudo de Caso

Descrição dos componentes de um sistema de software de um portal web.
Descrição:
- Breve resumo sobre sistemas distribuídos.

- Definir o que são módulos, componentes, estereótipos e nós. Quantos nós tem o sistema escolhido?

3. Requisitos do sistema (RS)

- Breve descritivo (de 3 a 5 linhas) sobre o portal web adotado.

(Inserir aqui a tabela de requisitos do sistema (pelo menos 12)).

4. Diagrama de componentes

- Breve descritivo (de 3 a 5 linhas) sobre os componentes do sistema.

(Inserir aqui a imagem de um repertório de componentes pela construção de um diagrama de componentes), criada na ferramenta escolhida – Astah, Draw.io ou Lucidchart).

5. Diagrama de implantação

- Breve descritivo (de 3 a 5 linhas) sobre os módulos de implantação do sistema.

(Inserir aqui a imagem da arquitetura da infraestrutura de TI, que deverá atender a um sistema distribuído, típico dos ambientes web, formados pela ligação de computadores servidores/clientes, sistemas operacionais, linguagens de programação e estereótipos das conexões (normalmente protocolos de rede). Esses componentes, estereótipos, módulos e nós de ligação deverão estar especificados na tabela de requisitos do sistema (RS), pela construção de um diagrama de implantação, criada na ferramenta escolhida – Astah, Draw.io ou Lucidchart).

6. Reflexões Finais

- Como se procede na componentização de um sistema de software?
- A forma de se fazer, especificar e modelar a arquitetura de um sistema de software.
- O que determina um componente do sistema?
- A melhoria da visualização, acompanhamento e suporte que oferece a arquitetura da infraestrutura de TI pelo emprego dos diagramas de componentes e de implantação da UML.

7. Referências

(Inclua pelo menos 1 referência do livro-texto e outras fontes utilizadas.)

Instituto de Ciências Exatas e Tecnologia	**Disciplina:** Análise e Projeto de Sistemas **Título da Aula:** Fusão do Produto e do Processo de Software	**Relatório 5**

1. Resumo Teórico

Escreva aqui um texto de 8 a 10 linhas abordando:
- O que é uma matriz de responsabilidades.
- Principais características da MR RACI.
- Como funciona o alinhamento do processo, produto e pessoas.
- Principais medidas a serem feitas no controle e esforços de serviços pelos stakeholders.

2. Estudo de Caso

Descrição dos principais perfis que compõem uma equipe de desenvolvimento de software.
Descrição:
- Breve resumo sobre o modelo de processo adotado (Incremental, RAD ou Espiral), JAD e distribuição das tarefas pela matriz de responsabilidades.

- Definir os artefatos de software que serão produzidos.

3. Modelo de processo de software

- Breve descritivo (de 3 a 5 linhas) sobre o modelo de processo de software adotado.

(Inserir aqui a imagem do modelo.)

4. Definição dos cargos e funções dos stakeholders

- Breve descritivo (de 3 a 5 linhas) sobre stakeholders.

(Inserir aqui a lista dos cargos e funções de cada participante da equipe.)

5. Matriz de responsabilidades

- Breve descritivo (de 3 a 5 linhas) sobre a MR RACI.

(Inserir aqui a matriz de responsabilidades.)

- Determinar o esforço de cada participante da equipe.

6. Diagrama de atividades

- Breve descritivo (de 3 a 5 linhas) da ferramenta escolhida para criar o diagrama de atividades.

(Inserir aqui a imagem do diagrama de atividades que mostre as atividades do desenvolvimento do software), criado pela ferramenta escolhida – Astah, Draw.io ou Lucidchart).

7. Reflexões Finais

- Como se determina a divisão de tarefas pela equipe.
- Como construir uma matriz de responsabilidades de acordo com os padrões do guia de projetos PMBOK®.
- Quais as melhorias que podem ser adotadas na distribuição de tarefas dos desenvolvedores.

8. Referências

(Inclua pelo menos 1 referência do livro-texto e outras fontes utilizadas.)

Instituto de Ciências Exatas e Tecnologia	**Disciplina:** Análise e Projeto de Sistemas **Título da Aula:** Processo Unificado	**Relatório 6**

1. Resumo Teórico

Escreva aqui um texto de 8 a 10 linhas abordando:
- O que é o processo unificado, método PERT e Kanban.
- Principais características do framework RUP.
- Como funciona a integração das atividades do RUP com a distribuição de tempos e dependências entre as tarefas, com o método PERT.
- Principais medidas e acompanhamentos a serem feitos no cronograma e rede de cartões.

2. RUP

Descrição das estruturas estáticas e dinâmicas, bem como as fases do processo unificado.

Descrição:
- Breve descritivo sobre as estruturas estáticas e dinâmicas.

- (Inserir aqui a imagem do framework RUP.)

- Definir as tarefas, dependências entre as tarefas e tempos distribuídos em um ciclo de 30 dias do processo unificado.

3. Tabela PERT

Descrição sobre a aplicação do método PERT.

Descrição:
- Breve descritivo sobre o PERT.

- (Inserir aqui a imagem do framework RUP.)

- Construir uma tabela com as tarefas, dependências entre as tarefas e tempos distribuídos em um ciclo de 30 dias do processo unificado.

4. Construção do cronograma e rede de cartões (Kanban)

- Breve descritivo (de 3 a 5 linhas) sobre cronograma e kanban.

- Fazer o input dos dados da tabela PERT por meio da aplicação para construção do cronograma e rede de cartões (GanttProject, Trello).

(Inserir aqui duas imagens: 1. Cronograma do projeto “Gantt”; e 2. Mapa da rede kanban pelo “Gráfico PERT”.)

5. Reflexões Finais

- Como funciona a distribuição de atividades e períodos do desenvolvimento de software com o RUP?
- Como funciona a construção da tabela PERT?
- A necessidade do cronograma e rede de cartões no acompanhamento das atividades do desenvolvimento.
- As principais funcionalidades de uma aplicação específica (p. ex.: GantProject) para a construção de cronogramas e rede de cartões.
- Quais as melhorias que podem ser adotadas na distribuição de tarefas e períodos para sua realização?

6. Referências

(Inclua pelo menos 1 referência do livro-texto e outras fontes utilizadas.)

Instituto de Ciências Exatas e Tecnologia	**Disciplina:** Análise e Projeto de Sistemas **Título da Aula:** Metodologias Ágeis I	**Relatório 7**

1. Resumo Teórico

Escreva aqui um texto de 8 a 10 linhas abordando:
- O que são as metodologias ágeis como o XP, Scrum e FDD?
- Principais características da metodologia ágil XP.
- Como funciona a integração das atividades do XP em um quadro kanban.
- Principais medidas e acompanhamentos a serem feitos no quadro kanban.

2. Metodologia ágil XP

Descrição do framework XP.
- Breve descritivo sobre as atividades em pares e a distribuição das atividades-chave entre os programadores e (ou) analistas da XP.
- Breve descritivo sobre as atividades-chave da XP.

3. Quadro kanban

Descrição sobre a aplicação do quadro kanban.
Descrição:

- Definir o modelo de cartão, que deverá ter as informações da tarefa, datas de início e fim da tarefa e se possível o(s) responsável (ou responsáveis) pela tarefa.
- Construir um quadro kanban (ferramentas: Draw.io, GanttProject, Trello) dimensionado para entrega da funcionalidade em 15 dias.

4. Reflexões Finais

- Como funciona a metodologia ágil XP alinhada a um quadro kanban?
- O porque existe um ganho no desempenho da equipe com o uso de um quadro kanban?
- Quais as melhorias que podem ser adotadas com o uso da rede de cartões e o dimensionamento do caminho crítico (análise da rede) para à execução de uma atividade?

5. Referências

(Inclua pelo menos 1 referência do livro-texto e outras fontes utilizadas.)

Instituto de Ciências Exatas e Tecnologia	**Disciplina:** Análise e Projeto de Sistemas **Título da Aula:** Metodologias Ágeis II	**Relatório 8**

1. Resumo Teórico

(Escreva aqui um texto de 8 a 10 linhas abordando:)
- O que é a ferramenta Trello?
- Principais características da metodologia ágil Scrum.
- Como funciona a integração das atividades do Scrum em um quadro kanban.
- Principais medidas e acompanhamentos a serem feitas com o uso da ferramenta Trello.

2. Metodologia ágil Scrum

Descrição do framework XP.
- Breve descritivo sobre as atividades em pares e a distribuição das atividades-chave entre os programadores e (ou) analistas da XP.
- Breve descritivo sobre as atividades-chave da XP.

3. Quadro kanban com a ferramenta Trello

Descrição sobre a aplicação Trello e construção do quadro kanban.
Descrição:

- Definir o modelo de cartão, que deverá ter as informações de descrição do backlog e das sprints.

- Construir um quadro kanban (ferramentas: Trello) dimensionado para entrega da funcionalidade em 15 dias.

- (Inserir aqui imagens de tela do Trello do quadro de cartões e do painel de acompanhamento.)

4. Reflexões Finais

- Como funciona a metodologia ágil Scrum e os parâmetros associados da ferramenta Trello?
- Por que existe uma flexibilidade maior na gestão da equipe de desenvolvimento com o uso da ferramenta Trello em relação a outras ferramentas?
- Quais as melhorias associadas ao Trello que podem ser adotadas com o uso da função Jira?

5. Referências

(Inclua pelo menos 1 referência do livro-texto e outras fontes utilizadas.)