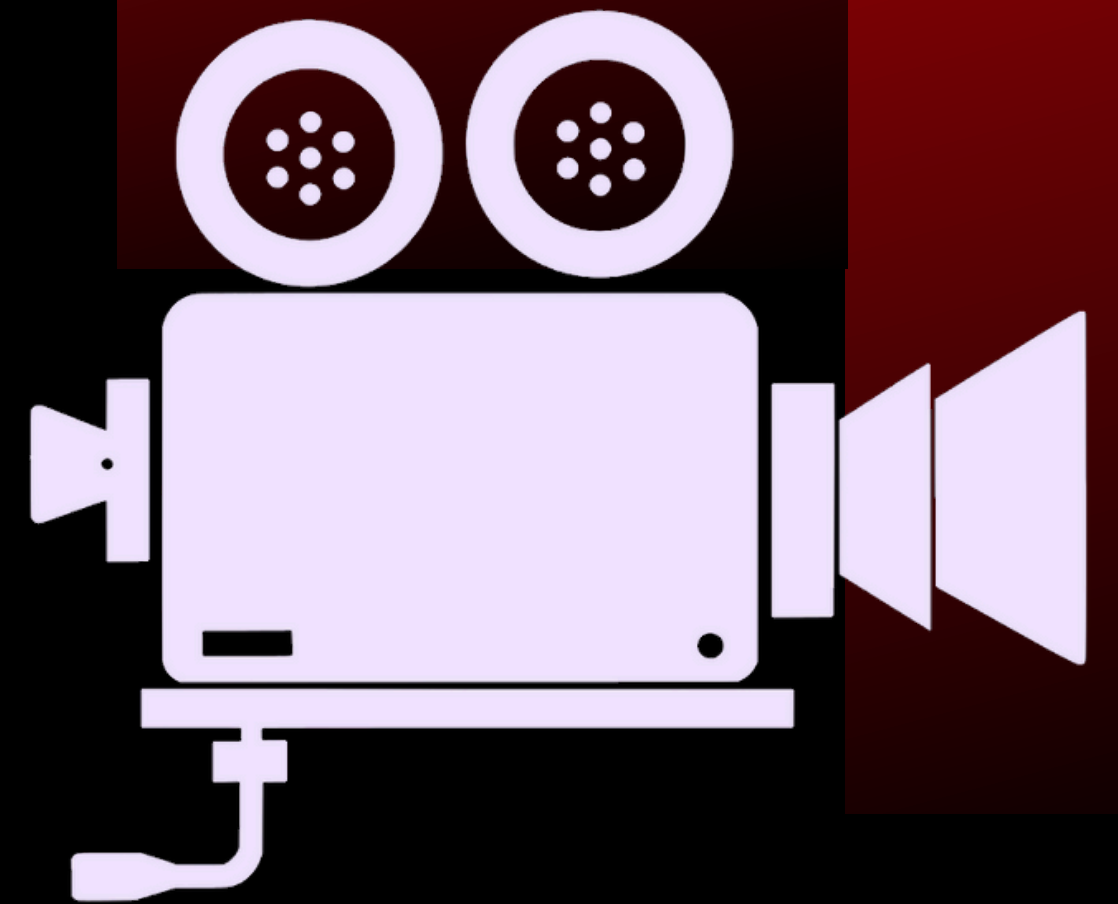


Cineticick

Trabalho final - Engenharia de Software



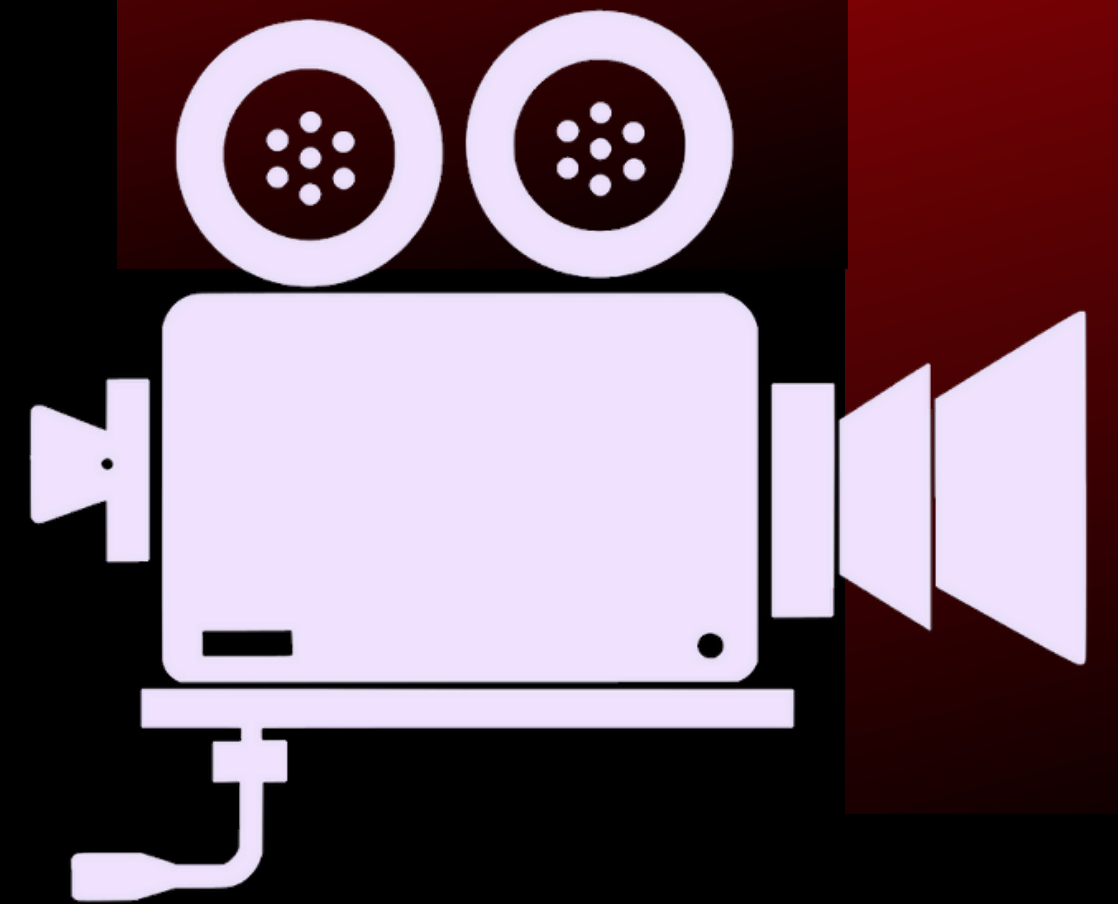
Universidade Federal do Tocantins

Curso: ciencias da computação

Professor: Edmilson Milhomen

Time: Lean Albuquerque, André Victor

Parte 1: Experiencia na disciplina



Universidade Federal do Tocantins

Curso: ciencias da computação

Professor: Edmilson Milhomen

Time: Lean Albuquerque, André Victor

Experiencia de aprendizado

Ter uma bom planejamento inicial e definições claras de tarefas.

adaptação a novas tecnologias e metodologias de trabalho

Importância da comunicação entre os membros da equipe

Desafios de soluções

Quanto aos integrantes:

- Desafio: Quantidade e funções incertas durante boa parte do desenvolvimento.
- Solução: Cobrar de forma assertiva e dividir as funções de forma clara e direta..

Erros em implementações:

- Desafio: Contato com novas tecnologias e pessoas geraram dificuldades a equipe
- Solução: Todos precisaram entender o que estavam fazendo antes de escrever código, em hipótese alguma ignorar ou postergar erros.

Pontos positivos

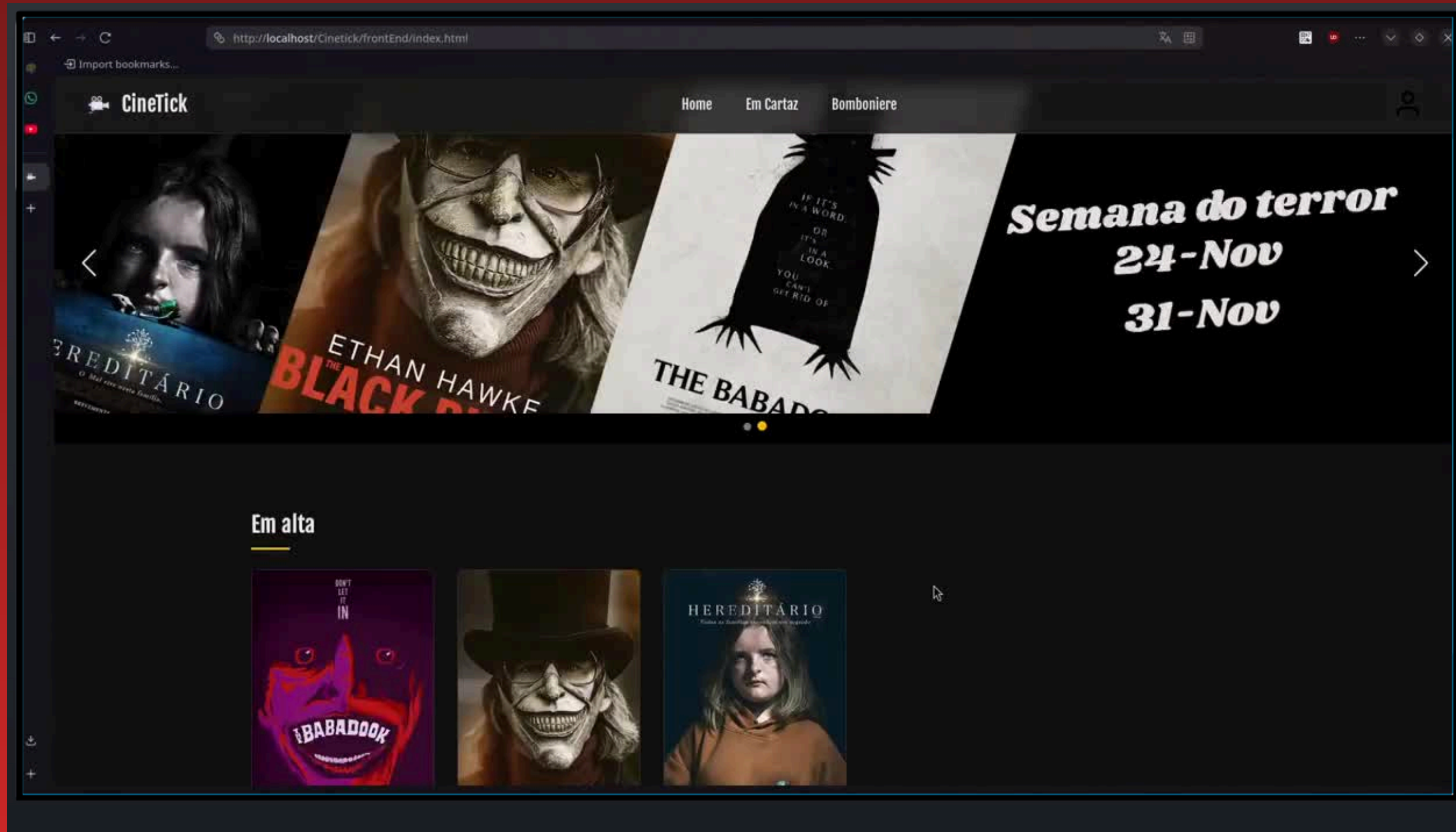
- Exposição direta ao ciclo de produção profissional, permitindo ao aluno começar a entender as dificuldades e cenários que deve encontrar no decorrer da sua carreira.
- Consolidação de fundamentos: Forçar a não utilização de frameworks coíbe o aluno a pular passos e o obriga a entender pelo menos em partes o que está sendo feito.
- Teoria aplicada: As metodologias foram expostas em sala de aula antes de serem colocadas em prática, o que evita que o aluno fique perdido.

Pontos negativos

- Falta de tempo: A matéria na prática exige do aluno atenção em dois períodos, um em sala de aula assistindo o conteúdo e outro em casa desenvolvendo o que foi aplicado, isso misturado com outros fatores acaba tornando a disciplina extremamente desgastante a longo prazo.
- Distribuição de grupos: Talvez fosse interessante ter um projeto pré estabelecido pelo professor no início das aulas e os grupos se dividirem em features. Isso evitaria discussões entre grupos e facilitaria a padronização da qualidade entre eles, visto que todos estariam no mesmo projeto.
- A soma dos fatores: A necessidade de entrega constante somada com onde a matéria se encontra na ementa do curso causa desconforto, muito alunos desistem por ainda não terem desenvolvido as atividades técnicas necessárias.

**Nota da experiecia
na Disciplina**

8/10



Parte 2:

Prova de
conhecimento

Do problema

- Reunião de alinhamento com o cliente para o levantamento de ideias e definição de requisitos, visando a compreensão completa do objetivo final. Em seguida, será elaborada a documentação preliminar do projeto, incluindo um fluxograma para mapear os principais eventos e ações
- Criar documentação inicial do projeto com um fluxograma para definir eventos e ações principais do projeto.

Dos requisitos

- Com aporte da Documentação e entendimento das necessidades do cliente em mãos podemos definir as prioridades do projeto.
- A entrega das funcionalidades tidas como prioritárias devem por óbvio ter maior ênfase no desenvolvimento, podendo inclusive, em caso de necessidade “engolir” ciclos inteiros para que estejam implementadas da melhor forma possível.

Validação de requisitos

- Confirmar junto ao cliente, de que o requisito descrito resolve o problema de negócio original.
- Verificar se não há conflitos entre diferentes requisitos e se a linguagem é clara tanto para o cliente quanto para os desenvolvedores (sem dupla interpretação).
- Analisar com o resto da equipe de engenharia para garantir que o requisito pode ser implementado dentro do prazo e orçamento estipulados.

Definição de tecnologias

- As ferramentas escolhidas devem suportar nativamente as regras de negócio e o volume de dados esperado.
- A arquitetura deve estar pronta para crescer junto com o negócio do cliente, sem gargalos de desempenho.
- Prioridade para tecnologias maduras, com suporte ativo da comunidade e atualizações de segurança constantes.
- Escolha de linguagens e padrões de mercado que facilitem a entrada de novos desenvolvedores e a manutenção do código a longo prazo.

Planejamento de Iterações

- Cada Iteração precisa ser planejada com base nas necessidades do projeto e visando sempre agregar o máximo de valor possível a cada encerramento de ciclo.
- É um momento importante para dar e receber feedback para o cliente, permitindo correção de fluxo caso se mostre necessário.
- Um ciclo mal planejado/executado pode acabar sendo fatal pro projeto.

Valor de Iterações

- Ter iterações no projeto é uma forma de evitar erros a longo prazo e interagir de forma eficiente com o cliente e com o time de desenvolvimento.
- Um projeto feito de forma direta do início ao fim sem iterações corre o risco de ignorar necessidades estabelecidas ou aproveitar oportunidades que surjam no decorrer do processo.
- É motivador ver o projeto crescer.

Padrão MVC

- O padrão MVC estabelece uma estrutura que promove organização e eficiência no desenvolvimento de software, levando a sistemas mais robustos, que são mais fáceis de gerenciar e portanto mais fáceis de escalar.
- Além disso, oferecem vantagens substanciais em termos de facilidade na manutenção, aproveitamento de código e adesão a práticas recomendadas pela indústria.

APIs

- As APIs REST possibilitam o desenvolvimento de sistemas flexíveis, escaláveis e integráveis, facilitando a comunicação entre aplicações e favorecendo a reutilização de recursos.
- Simplifica significativamente a manutenção e garante suporte a diversas plataformas, seguindo padrões amplamente reconhecidos, como o protocolo HTTP.

Testes Unitarios

- Os testes unitários são de suma importância para o projeto, identificando problemas e bugs que poderiam ser descobertos somente a longo prazo, colaborando para que o código do projeto seja confiável e possa sofrer novas alterações sem o risco de quebra

Controle de versão do código

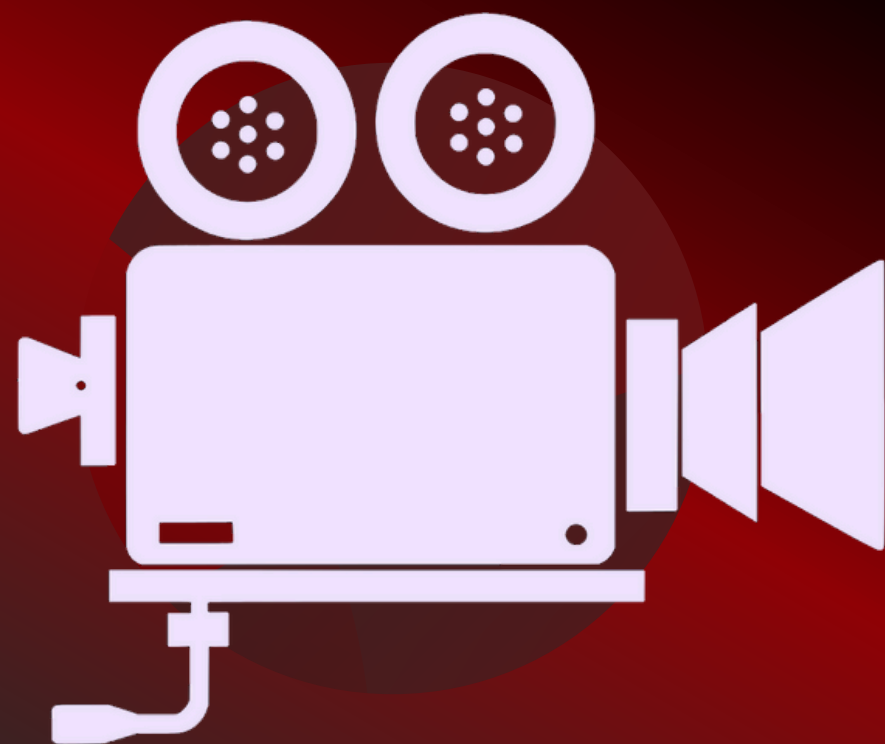
- A importância de manter o rastreamento das alterações feitas por cada colaborador, o controle das branches em paralelo sem alteração da versão principal, a facilidade em recuperar versões anteriores do projeto, proporcionando uma segurança a mais.

Fluxo de trabalho organizado pelo GitFlow

- O GitFlow beneficia na organização das tarefas para cada membro da equipe utilizando branches específicas para cada ramo do projeto (master, develop, feature, release, hotfix), o que torna o projeto mais fácil de compreender, ajudando na adição de novas funções, correções de erros e na entrega das versões do produto.

Release do produto

- Uma release é um ponto crucial no desenvolvimento de software, marcando uma versão específica que atingiu estabilidade para ser distribuída aos usuários finais, incluindo novas funcionalidades e correções de bugs planejadas para essa versão.



Obrigado
pela atenção

