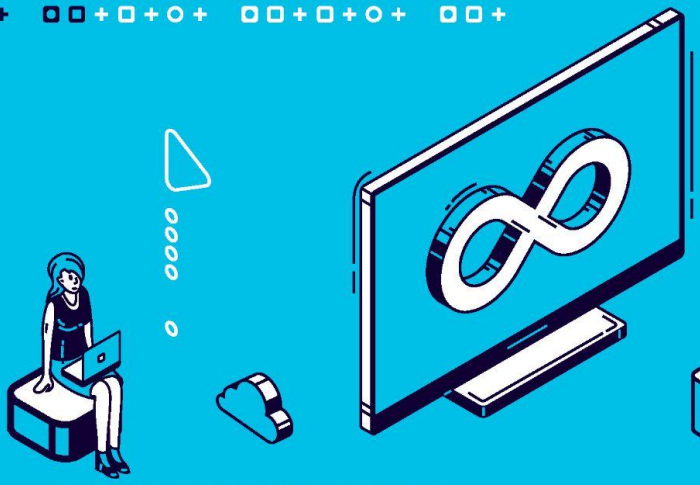


COMO MELHORAR O DESEMPENHO DA SUA

EQUIPE DE DEVS

SOFTWARE HOUSE EXPONENCIAL

#FAÇAPARTE DAHISTÓRIA

COMO MELHORAR O DESEMPENHO DA SUA EQUIPE DE DEVS

Um manual para você organizar a sua equipe de desenvolvedores

SOFTWARE HOUSE EXPONENCIAL

COMO MELHORAR O DESEMPENHO DA SUA EQUIPE DE DEVS

Um manual para você organizar a sua equipe de desenvolvedores

1ª
Edição
São
Paulo
2023

Autor: Equipe Software House Exponencial
Revisão: Equipe Software House Exponencial
Diagramação: Exponencial MKT
Imagens arquivo pessoal (Exponencial Holding)

DEDICATÓRIA

Este livro é dedicado a todos os empresários de software do Brasil, que trabalham incansavelmente para oferecer soluções inovadoras e de qualidade aos seus clientes. Sabemos que não é fácil liderar equipes de desenvolvimento, lidar com a rápida evolução da tecnologia e superar os desafios diários do mercado. Por isso, esperamos que este livro possa contribuir para o aprimoramento das equipes de desenvolvimento, para o aumento da produtividade e para a realização de projetos cada vez mais bem-sucedidos. A todos os empresários de software do Brasil, nosso muito obrigado pelo trabalho e dedicação à tecnologia e ao desenvolvimento do país.

___ Equipe Software House Exponencial

AGRADECIMENTOS

Agradeço a todos os colaboradores das empresas do Grupo Exponencial que tem contribuído diariamente para fazermos juntos parte positivamente da história do mercado de software do Brasil.

PREFÁCIO

Caro leitor,

Se você é um líder ou gestor de uma equipe de desenvolvimento de software, sabe que a tecnologia avança rapidamente e que é fundamental manter sua equipe motivada e produtiva para que sua Software house se mantenha competitiva. No entanto, nem sempre é fácil manter a equipe motivada e engajada. Problemas de organização, comunicação e clareza nos objetivos podem afetar o desempenho da equipe e gerar atrasos e retrabalho.

Nesse contexto, surge a necessidade de buscar maneiras de melhorar o desempenho da sua equipe de desenvolvedores. Para isso, é importante ter uma visão clara dos objetivos da equipe e dos projetos em que ela está trabalhando. Essa visão deve ser compartilhada com todos os membros da equipe, para que todos trabalhem em harmonia e saibam exatamente o que é esperado deles.

Este livro apresenta estratégias e técnicas para melhorar o desempenho da sua equipe de desenvolvedores. Abordaremos desde a definição de objetivos claros e mensuráveis, passando pela seleção de ferramentas de gestão de projetos, até a implementação de indicadores de desempenho e a análise e otimização dos processos da equipe de desenvolvimento.

Para alcançar esse objetivo, convidamos diversos especialistas em gestão de equipes de desenvolvimento de software para contribuir com suas experiências e conhecimentos. Cada capítulo é escrito por um especialista na área, trazendo uma visão única e valiosa sobre o tema.

Este livro é voltado para líderes e gestores de equipes de desenvolvimento de software, mas também pode ser útil para desenvolvedores que desejam aprimorar suas habilidades e contribuir para o sucesso da equipe. Os conceitos e técnicas apresentados neste livro são baseados em exemplos práticos e experiências reais, tornando-se uma ferramenta útil e prática para quem visa aprimorar o desempenho de sua equipe de desenvolvimento de software.

Esperamos que este livro seja útil para você e sua equipe, que possa contribuir para a realização de projetos cada vez mais bem-sucedidos e para o crescimento da sua Software house. Boa leitura e sucesso em seus projetos!

Atenciosamente,
Equipe Software House Exponencial

SUMÁRIO

INTRODUÇÃO

Como líder ou gestor de uma equipe de desenvolvimento de software, você sabe que a tecnologia avança rapidamente e que é fundamental manter sua equipe motivada e produtiva para que sua Software house se mantenha competitiva. Nesse contexto, é importante ter em mente que o sucesso de um projeto de software depende não só das habilidades técnicas dos desenvolvedores, mas também da capacidade de gestão e liderança dos gestores e líderes da equipe.

Henry Ford, o lendário empresário e fundador da Ford Motor Company, uma vez disse: "Juntos podemos fazer qualquer coisa. Separados, somos fracos". Esta frase exemplifica a importância da colaboração e do trabalho em equipe para alcançar objetivos comuns. Na área de desenvolvimento de software, o trabalho em equipe é ainda mais importante, já que cada integrante tem um papel fundamental no processo.

Pensando nisso, reunimos neste livro diversos especialistas em gestão de equipes de desenvolvimento de software, para apresentar estratégias e técnicas que podem ajudá-lo a aprimorar o desempenho da sua equipe. Cada capítulo aborda um tema específico, desde a definição de objetivos claros e mensuráveis até a implementação de indicadores de desempenho e a análise e otimização dos processos da equipe de desenvolvimento.

Como disse o escritor e filósofo britânico, G.K. Chesterton, “a única maneira de fazer um bom trabalho é amando o que você faz”. Acreditamos que este livro pode ajudá-lo a criar uma equipe de desenvolvimento de software apaixonada pelo que faz e focada em alcançar seus objetivos de forma eficiente e eficaz. Esperamos que este livro seja útil para você e sua equipe, contribuindo para o sucesso dos seus projetos e para o crescimento da sua Software house.

Boa leitura e sucesso em seus projetos!

AS ETAPAS DO PROCESSO DE DESENVOLVIMENTO

O processo de desenvolvimento de software é uma sequência de atividades interligadas que visam criar um produto final que atenda às necessidades dos clientes e usuários. O processo é dividido em várias etapas, cada uma com suas próprias características e objetivos específicos.

A primeira etapa é o planejamento, onde a equipe de desenvolvimento se reúne para definir as funcionalidades e objetivos do software a ser desenvolvido. É importante definir metas realistas e estabelecer um cronograma para o projeto. Nesta etapa, é importante também definir os recursos necessários para o projeto e identificar potenciais riscos que possam impactar o desenvolvimento.

Após a etapa de planejamento, vem a etapa de análise. Nesta fase, a equipe de desenvolvimento analisa os requisitos do software e define a arquitetura e tecnologias a serem utilizadas. O objetivo é mapear as necessidades do cliente e identificar as funcionalidades que o software deve possuir.

Após a etapa de planejamento, é importante que a equipe de desenvolvimento realize a etapa de análise. Nesta etapa, a equipe deve analisar os requisitos do software e definir a arquitetura e tecnologias a serem utilizadas. É importante que a equipe tenha uma compreensão clara dos objetivos do projeto e das necessidades dos usuários finais para poder projetar o software eficazmente.

Uma vez que a análise foi concluída, a equipe de desenvolvimento pode passar para a etapa de design, na qual a interface do software é definida e o design de cada tela e funcionalidade é realizado. Nesta fase, a equipe deve garantir que o software seja fácil de usar e que atenda às necessidades dos usuários finais.

A etapa de design é a próxima, nesta fase a equipe de desenvolvimento define a interface do software e realiza o design de cada tela e funcionalidade. O objetivo é criar um software atraente e de fácil utilização pelos usuários.

Após a definição do design, vem a fase de implementação, onde a equipe de desenvolvimento começa a criar o software propriamente dito. Nesta fase, são definidas as linguagens de programação a serem utilizadas e o código é desenvolvido.

A fase de testes é a próxima etapa, onde o software é testado para garantir que ele está funcionando corretamente e atendendo aos requisitos definidos anteriormente. Nesta etapa, são identificados e corrigidos eventuais erros e problemas que prejudiquem a utilização do software.

Após testado, vem a fase de implantação, onde o software é entregue aos usuários finais. Nesta etapa, é importante garantir que os usuários recebam o treinamento necessário para utilizar o software corretamente e que o

suporte técnico esteja disponível para solucionar quaisquer problemas que possam surgir.

A fase de manutenção é a última etapa do processo de desenvolvimento de software. Nesta fase, a equipe de desenvolvimento realiza a manutenção do software, corrigindo eventuais problemas e realizando melhorias. O objetivo é garantir que o software continue atendendo às necessidades dos usuários e que possa ser atualizado de acordo com as mudanças no mercado e nas tecnologias.

Cada etapa do processo de desenvolvimento de software possui seus próprios indicadores e KPIs que ajudam a equipe a monitorar o progresso do projeto e garantir que ele está sendo entregue no prazo e orçamento previstos. Alguns exemplos de indicadores comuns incluem o número de bugs encontrados durante a fase de testes, a quantidade de tempo gasta em cada etapa do processo, a taxa de aceitação do software pelos usuários finais, entre outros.

Em resumo, o processo de desenvolvimento de software é composto por várias etapas interligadas que visam criar um produto final que atenda às necessidades dos usuários. Cada etapa possui seus próprios objetivos e indicadores de desempenho, e é importante que a equipe de desenvolvimento esteja alinhada e comprometida para garantir o sucesso do projeto. Além disso, a comunicação e colaboração entre os membros da equipe são fundamentais

para garantir que as metas e objetivos do projeto sejam alcançados.

É importante lembrar que o processo de desenvolvimento de software é dinâmico e pode exigir ajustes ao longo do caminho. Por isso, é importante que a equipe esteja aberta a feedbacks e adapte o processo sempre que necessário.

Alguns autores famosos já afirmaram a importância de um processo bem estruturado e organizado para o sucesso do desenvolvimento de software. O escritor Steve McConnell, por exemplo, afirmou que "o processo de desenvolvimento de software é uma atividade altamente intelectual que deve ser planejada e gerenciada com cuidado para os resultados serem previsíveis e confiáveis".

Outro autor famoso, Frederick Brooks, afirmou que "a complexidade de um software aumenta exponencialmente à medida que ele é desenvolvido". Por isso, é fundamental que a equipe de desenvolvimento esteja preparada para lidar com as complexidades do processo e tenha um planejamento sólido para garantir que o projeto seja entregue no prazo e orçamento previstos.

Em resumo, o sucesso do processo de desenvolvimento de software depende de uma equipe comprometida, um planejamento bem estruturado e uma execução cuidadosa de cada etapa do processo. Com uma

abordagem sistemática e organizada, é possível criar produtos finais de alta qualidade que atendam às necessidades dos usuários.

PLANEJAMENTO

A etapa de planejamento é uma das etapas mais importantes em uma equipe de desenvolvimento de software. Nesta etapa, a equipe se reúne para definir as funcionalidades e objetivos do software a ser desenvolvido. Para isso, é fundamental ter uma compreensão clara das necessidades dos usuários e do mercado onde o software será utilizado.

Para definir os objetivos do software, é importante ter em mente o que se espera alcançar com o software e como ele pode ajudar a Software house ou usuários finais a atingir seus objetivos. É importante que os objetivos sejam claros, mensuráveis e realistas, para garantir que a equipe possa trabalhar eficientemente e focada na entrega de valor.

OBJETIVOS DO PLANEJAMENTO

Na etapa de planejamento de software, os objetivos definidos devem estar alinhados com as necessidades dos usuários e com as expectativas da Software house. Abaixo, estão alguns exemplos de objetivos que podem ser traçados nesta etapa:

AUMENTAR A EFICIÊNCIA

O objetivo aqui é criar um software que possa automatizar tarefas manuais ou reduzir o tempo necessário para realizar uma determinada atividade, o que pode levar a uma maior eficiência e produtividade da equipe.

MELHORAR A EXPERIÊNCIA DO USUÁRIO

O objetivo aqui é criar um software que seja fácil de usar e que ofereça uma boa experiência para o usuário. Isso pode ajudar a aumentar a satisfação do cliente e a fidelidade à marca.

REDUZIR OS CUSTOS

O objetivo aqui é criar um software que ajude a reduzir os custos da Software house, seja automatizando tarefas ou melhorando a eficiência dos processos.

AUMENTAR A RECEITA

O objetivo aqui é criar um software que ajude a aumentar a receita da Software house, seja por meio da venda direta do software ou por meio do aumento da eficiência da equipe.

MELHORAR A SEGURANÇA

O objetivo aqui é criar um software seguro e que proteja os dados dos usuários e da empresa.

AUMENTAR A ESCALABILIDADE

O objetivo aqui é criar um software que possa ser facilmente escalável, para poder atender às necessidades da empresa à medida que ela cresce.

É importante lembrar que os objetivos devem ser específicos, mensuráveis e realistas, para a equipe poder trabalhar eficiente e focada na entrega de valor. Além disso, eles devem estar alinhados com as necessidades do usuário e da Software house, para garantir que o software entregue valor e gere resultados positivos.

CANVAS DE PROPOSTA DE VALOR

O Canvas de Proposta de Valor é uma ferramenta utilizada para criar e comunicar o valor que o software oferece aos clientes. Ele é dividido em seis etapas: Ganhos, Dores e Tarefas do Cliente, Produtos e Serviços, Criadores de Ganhos e Alivio das Dores. A seguir, explicaremos como utilizar o Canvas de Proposta de Valor em uma equipe de desenvolvimento de software na etapa do planejamento.

GANHOS

Nesta etapa, é importante identificar os benefícios e ganhos que o software pode trazer para os clientes. Isso inclui economia de tempo, aumento da produtividade, redução de custos, entre outros. A equipe de desenvolvimento deve identificar quais são os ganhos mais

importantes para os clientes e como o software pode atendê-los.

DORES E TAREFAS DO CLIENTE

Nesta etapa, é importante identificar as dores e problemas que os clientes enfrentam no seu dia a dia. Isso inclui tarefas que são difíceis ou demoradas, problemas que causam frustração ou insatisfação, entre outros. A equipe de desenvolvimento deve identificar quais são as dores mais importantes para os clientes e como o software pode aliviá-las.

PRODUTOS E SERVIÇOS

Nesta etapa, é importante identificar os produtos e serviços que serão oferecidos pelo software. A equipe de desenvolvimento deve identificar quais são as funcionalidades mais importantes e como elas podem atender às necessidades dos clientes.

CRIADORES DE GANHOS

Nesta etapa, é importante identificar quais são os fatores que podem contribuir para o sucesso do software. Isso inclui o valor da marca, a qualidade do software, a facilidade de uso, entre outros. A equipe de desenvolvimento deve identificar quais são os fatores mais importantes e como eles podem ser otimizados.

ALIVIO DAS DORES

Nesta etapa, é importante identificar como o software pode aliviar as dores e problemas dos clientes. Isso inclui a redução de custos, a melhoria da eficiência, a simplificação de tarefas, entre outros. A equipe de desenvolvimento deve identificar como o software pode aliviar as dores mais importantes para os clientes.

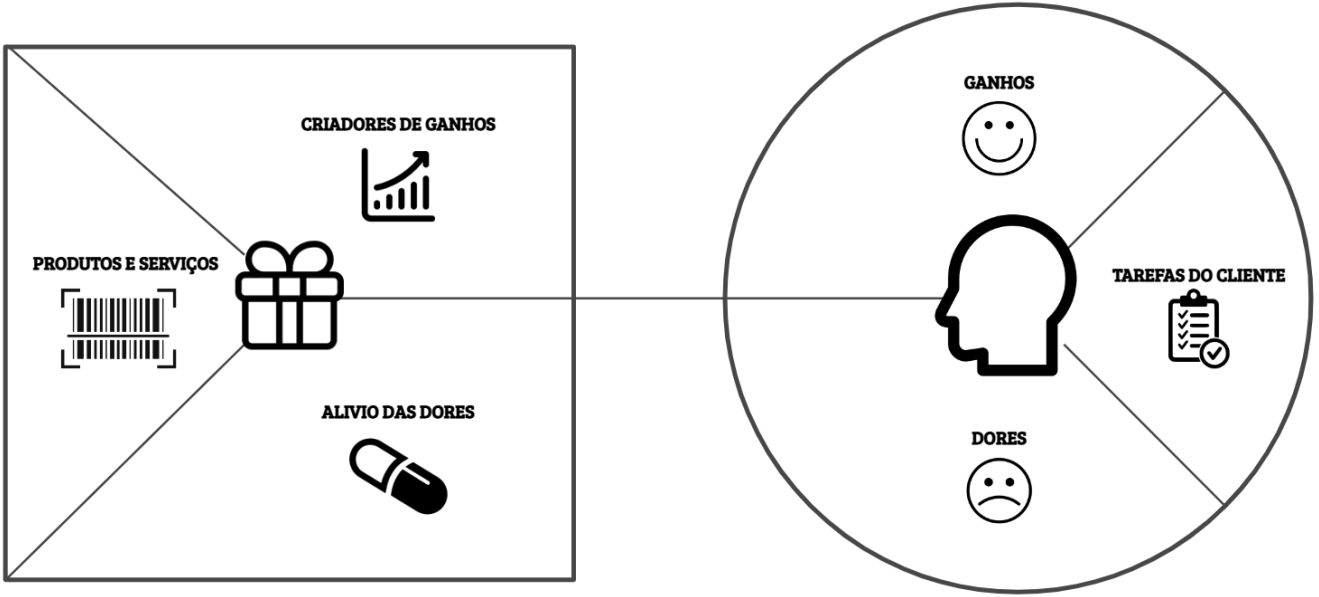
PROPOSTA DE VALOR

Nesta etapa, é importante sintetizar todas as informações coletadas nas etapas anteriores e criar uma proposta de valor clara e objetiva. A equipe de desenvolvimento deve comunicar claramente como o software pode atender às necessidades dos clientes, aliviar suas dores e gerar ganhos para eles.

Ao utilizar o Canvas de Proposta de Valor na etapa de planejamento, a equipe de desenvolvimento pode compreender melhor as necessidades dos clientes e criar soluções que atendam a essas necessidades. Além disso, a ferramenta pode ajudar a identificar oportunidades de melhoria e a desenvolver uma estratégia de negócios mais eficaz.

CANVAS DA PROPOSTA DE VALOR

EMPRESA: _____



Abaixo, segue um exemplo de um Canvas de Proposta de Valor preenchido em uma tabela para um software de gestão de projetos:

Etapas	Descrição
Ganhos	Economia de tempo e dinheiro, aumento da produtividade, maior transparência e visibilidade dos projetos, melhoria na comunicação e colaboração da equipe
Dores e Tarefas do Cliente	Dificuldade de acompanhar o andamento dos projetos, falta de visibilidade sobre o trabalho da equipe, problemas de comunicação entre os membros do projeto, dificuldade em gerenciar recursos e prazos
Produtos e Serviços	Dashboard com informações do projeto em tempo real, gerenciamento de recursos e tarefas, ferramentas de comunicação integradas, geração de relatórios e gráficos para análise
Criadores de Ganhos	Software fácil de usar, interface intuitiva, suporte técnico eficiente
Alivio das Dores	Simplificação do gerenciamento de projetos, melhor controle de recursos e prazos, melhoria na comunicação e colaboração da equipe
Proposta de Valor	Nosso software de gestão de projetos oferece uma solução completa e intuitiva para o gerenciamento eficiente de projetos, com ferramentas de comunicação, monitoramento de recursos, tarefas e prazos, gerando economia de tempo e dinheiro para sua empresa e melhorando a comunicação e colaboração da equipe.

Este é apenas um exemplo de como o Canvas de Proposta de Valor pode ser preenchido. É importante lembrar que cada software e equipe de desenvolvimento possuem suas próprias necessidades e objetivos, e o preenchimento do Canvas deve ser personalizado de acordo com essas particularidades.

A etapa de planejamento é uma das fases mais importantes no desenvolvimento de software, ao ser nela que a equipe define os objetivos, requisitos, recursos e prazos do projeto. É fundamental que o planejamento seja realizado cuidadosamente e detalhada, para que o desenvolvimento do software seja mais eficiente e eficaz.

Segundo o autor Tom DeMarco, "Você não pode controlar o que você não pode medir". Isso significa que, para que o desenvolvimento do software seja bem-sucedido, é necessário estabelecer métricas e indicadores que possam ser mensurados ao longo do projeto. Além disso, é importante que a equipe tenha um cronograma bem definido e realista, para ser possível cumprir as metas estabelecidas.

Outro autor renomado na área de desenvolvimento de software é Steve McConnell, que destaca a importância de um planejamento adequado para o sucesso do projeto: "O planejamento de software é uma corrida contra o caos e a dissipação. Você deve ter um plano de projeto e seguir esse

plano se quiser concluir o projeto no prazo e dentro do orçamento".

Em resumo, a etapa de planejamento é fundamental para o sucesso do desenvolvimento de software, ao permitir que a equipe defina objetivos claros, estabeleça um cronograma realista, estabeleça métricas e indicadores para o acompanhamento do projeto e evite o caos e a dissipação durante a execução do projeto. Como disse o famoso autor Stephen Covey, "Comece com o fim em mente". É importante que a equipe tenha uma visão clara dos objetivos do projeto desde o início, para poder trabalhar de forma mais eficiente e eficaz em direção a esses objetivos.

ANÁLISE

Na etapa de análise, a equipe de desenvolvimento de software planeja compreender os requisitos do projeto e definir a arquitetura e tecnologias a serem utilizadas. É uma fase importante do processo de desenvolvimento de software, ao permitir que a equipe compreenda as necessidades dos usuários e as possibilidades técnicas para a implementação das funcionalidades.

Para alcançar os objetivos da etapa de análise, a equipe pode definir algumas metas específicas, como:

METAS E OBJETIVOS DA ANÁLISE

Compreender as necessidades dos usuários: O objetivo principal da etapa de análise é compreender as necessidades dos usuários do software e identificar as funcionalidades que devem ser implementadas para atender a essas necessidades. A equipe deve ter como meta coletar informações suficientes sobre as necessidades dos usuários e transformá-las em requisitos claros e objetivos.

DEFINIR A ARQUITETURA DO SOFTWARE

A equipe de desenvolvimento deve definir a arquitetura do software, ou seja, a estrutura do sistema, as tecnologias e ferramentas que serão utilizadas e como os componentes do software interagirão entre si. O objetivo dessa meta é garantir que o software seja escalável, seguro, de fácil manutenção e eficiente.

VALIDAR OS REQUISITOS

É importante validar os requisitos do software com os usuários e outras partes interessadas, para garantir que o software atenderá às suas necessidades e expectativas. O objetivo dessa meta é garantir que o software seja desenvolvido conforme as necessidades reais dos usuários e evitar retrabalhos e desperdícios.

ESTABELECER AS PRIORIDADES

A equipe deve estabelecer as prioridades para a implementação das funcionalidades do software, conforme a sua importância e impacto no projeto. O objetivo dessa meta é garantir que as funcionalidades mais importantes sejam implementadas primeiro, maximizando o valor entregue ao usuário.

DEFINIR O ESCOPO DO SOFTWARE

A equipe deve definir o escopo do software, ou seja, o conjunto de funcionalidades que serão implementadas na primeira versão do software. O objetivo dessa meta é garantir que o software entregue valor ao usuário desde a primeira versão, simultaneamente, em que se mantém nos prazos e recursos definidos para o projeto.

Em resumo, as metas e objetivos da etapa de análise são garantir que a equipe compreenda as necessidades dos usuários, defina a arquitetura e tecnologias do software, valide os requisitos, estabeleça as prioridades e defina o escopo do software. Esses objetivos são fundamentais para o sucesso do projeto e devem ser definidos com clareza e objetividade para orientar a equipe ao longo do processo de desenvolvimento de software.

INDICADORES E KPIS

NÚMERO DE REQUISITOS IDENTIFICADOS

Este indicador pode ajudar a equipe a medir o progresso na compreensão das necessidades dos usuários e dos requisitos do software.

NÍVEL DE DETALHAMENTO DA ARQUITETURA DEFINIDA

Este indicador pode ajudar a equipe a medir o grau de detalhamento da arquitetura do software e garantir que ela seja adequada para a implementação das funcionalidades.

TAXA DE ADOÇÃO DAS TECNOLOGIAS SELECIONADAS

Este indicador pode ajudar a equipe a medir o sucesso na escolha das tecnologias a serem utilizadas no software,

garantindo que elas atendam aos requisitos de performance, segurança e escalabilidade.

Em resumo, a etapa de análise é fundamental para o sucesso do desenvolvimento de software, ao permitir que a equipe compreenda os requisitos do projeto e defina a arquitetura e tecnologias mais adequadas para a implementação das funcionalidades. O uso de metas, indicadores e KPIs pode ajudar a equipe a monitorar o progresso da etapa de análise e garantir que ela seja realizada de forma eficiente e eficaz.

CANVAS DE REQUISITOS

O Canvas de Requisitos é uma ferramenta visual que pode ajudar a equipe de desenvolvimento de software a mapear e organizar os requisitos do projeto claramente e objetiva. O objetivo do Canvas é permitir que a equipe compreenda as necessidades dos usuários e defina as funcionalidades que o software deve oferecer para atendê-las.

O Canvas de Requisitos é dividido em diferentes seções, cada uma representando um aspecto importante do projeto. As seções do Canvas podem variar conforme as necessidades da equipe e do projeto em questão, mas geralmente incluem:

PERSONA

Nesta seção, a equipe define quem são os usuários do software, quais são suas características e necessidades.

NECESSIDADES

Nesta seção, a equipe identifica as necessidades dos usuários e as funcionalidades que o software deve oferecer para atendê-las.

FUNCIONALIDADES

Nesta seção, a equipe define as funcionalidades que o software deve oferecer para atender às necessidades dos usuários.

PRIORIZAÇÃO

Nesta seção, a equipe define a importância e o impacto de cada funcionalidade, para definir a ordem de implementação das funcionalidades.

MVP

Nesta seção, a equipe define o que será incluído no MVP (Minimum Viable Product), ou seja, a versão mínima do software que pode ser lançada para atender às necessidades dos usuários.

Ao preencher o Canvas de Requisitos, a equipe pode ter uma visão clara e objetiva dos requisitos do software, permitindo que defina as funcionalidades mais importantes e estabelecer prioridades para a implementação. Além disso, o Canvas pode ser atualizado e revisado ao longo do projeto, permitindo que a equipe faça ajustes e adaptações conforme as necessidades dos usuários e do próprio desenvolvimento do software.

Em resumo, o Canvas de Requisitos é uma ferramenta visual e colaborativa que pode ajudar a equipe de desenvolvimento de software a mapear e organizar os requisitos do projeto de forma clara e objetiva, permitindo que defina as funcionalidades mais importantes e estabelecer prioridades para a implementação.

Abaixo um exemplo do canvas de requisito preenchido em uma tabela:

Seção	Descrição	Exemplo
Persona	Definição dos usuários do software	João, 30 anos, estudante de engenharia
Necessidades	Identificação das necessidades dos usuários	Gerenciamento de tarefas acadêmicas
Funcionalidades	Definição das funcionalidades do software	Criação de tarefas, definição de prazos, notificação de datas de

		entrega
Priorização	Definição da ordem de implementação das funcionalidades	Criação de tarefas > Definição de prazos > Notificação de datas de entrega
MVP	Definição das funcionalidades mínimas para o lançamento do software	Criação de tarefas, definição de prazos

DESIGN

Na etapa de design, a equipe de desenvolvimento de software define a interface do software e realizar o design de cada tela e funcionalidade. O objetivo é criar uma interface intuitiva e fácil de usar para os usuários do software.

Para atingir esses objetivos, a equipe de desenvolvimento deve seguir algumas metas importantes, como:

METAS E OBJETIVOS

DEFINIR O FLUXO DE NAVEGAÇÃO

O objetivo dessa meta é garantir que o usuário possa navegar intuitivamente pelo software, sem se perder ou ficar confuso com as funcionalidades. A equipe deve definir um fluxo de navegação lógico e coerente, que facilite a interação do usuário com o software.

CRIAR UM DESIGN RESPONSIVO

O objetivo dessa meta é garantir que o software possa ser acessado em diferentes dispositivos, como smartphones, tablets e desktops. A equipe deve criar um design responsivo, que se adapte automaticamente ao tamanho da tela do dispositivo do usuário, garantindo uma experiência de uso consistente.

DEFINIR UM ESTILO VISUAL COERENTE

O objetivo dessa meta é garantir que o software tenha uma identidade visual coerente, que reflita a marca ou Software house que o desenvolveu. A equipe deve definir um estilo visual consistente, que seja aplicado em todas as telas e funcionalidades do software.

VALIDAR O DESIGN COM USUÁRIOS

O objetivo dessa meta é validar o design do software com usuários reais, a fim de identificar problemas e oportunidades de melhoria. A equipe deve realizar testes de usabilidade com usuários, a fim de avaliar a experiência de uso e realizar ajustes no design, se necessário.

INDICADORES E KPIs

Os indicadores e KPIs são ferramentas importantes para medir o sucesso da etapa de design de um software. Abaixo, detalho melhor cada um dos indicadores e KPIs mencionados anteriormente:

TAXA DE CONVERSÃO DE USUÁRIOS

Esse indicador mede a porcentagem de usuários que realizam uma ação desejada no software, como, por exemplo, realizar uma compra ou preencher um formulário. Uma alta taxa de conversão indica que o design do software está

eficiente e intuitivo, enquanto uma baixa taxa de conversão pode indicar problemas de usabilidade ou falta de clareza nas funcionalidades.

TEMPO MÉDIO DE PERMANÊNCIA NO SOFTWARE

Esse indicador mede o tempo médio que os usuários permanecem no software. Um tempo médio de permanência elevado indica que os usuários estão engajados com o software, enquanto um tempo médio de permanência baixo pode indicar desinteresse ou dificuldades na interação com o software.

TAXA DE REJEIÇÃO DE USUÁRIOS

Esse indicador mede a porcentagem de usuários que abandonam o software após acessá-lo. Uma taxa de rejeição elevada pode indicar problemas de usabilidade ou falta de clareza nas funcionalidades.

NÍVEL DE SATISFAÇÃO DOS USUÁRIOS

Esse indicador mede o grau de satisfação dos usuários com o software. É possível medir a satisfação por meio de pesquisas de satisfação ou por meio da análise de comentários e feedbacks dos usuários. Um alto nível de satisfação indica que o software atende às expectativas dos usuários e oferece uma boa experiência de uso.

É importante destacar que os indicadores e KPIs podem variar conforme o software e as necessidades da equipe de desenvolvimento. Porém, é fundamental que a equipe defina indicadores que permitam medir o sucesso do software em relação ao design e usabilidade.

CANVAS DE JORNADA DO USUÁRIO

O Canvas de Jornada do Usuário é uma ferramenta que pode ser utilizada na etapa de design de um software para entender melhor as necessidades e expectativas dos usuários em relação ao produto. Ele ajuda a visualizar o caminho que o usuário percorre ao utilizar o software, identificando as suas dores, desafios e oportunidades de melhoria ao longo dessa jornada.

Esse canvas é dividido em várias etapas, que correspondem aos diferentes momentos da jornada do usuário, desde a descoberta do software até a utilização e fidelização. Algumas das etapas do Canvas de Jornada do Usuário incluem:

DESCOBERTA

Nessa etapa, o usuário toma conhecimento do software e começa a explorar suas funcionalidades. É importante entender como o usuário descobre o software, quais são as suas primeiras impressões e quais são as suas expectativas.

AQUISIÇÃO

Nessa etapa, o usuário realiza o cadastro ou aquisição do software. É importante entender quais são as barreiras ou dificuldades que podem impedir o usuário de concluir essa etapa, como a complexidade do processo de cadastro ou a falta de clareza sobre os benefícios do software.

ATIVAÇÃO

Nessa etapa, o usuário começa a utilizar o software pela primeira vez. É importante entender como o usuário utiliza o software nesse momento, quais são as suas primeiras experiências e quais são as funcionalidades mais relevantes para ele.

UTILIZAÇÃO

Nessa etapa, o usuário utiliza o software recorrentemente. É importante entender como o usuário utiliza o software no seu dia a dia, quais são as suas principais necessidades e quais são as oportunidades de melhoria.

FIDELIZAÇÃO

Nessa etapa, o usuário se torna um usuário fiel do software. É importante entender como o software contribui

para a fidelização do usuário, quais são as funcionalidades que mais impactam a sua decisão de continuar utilizando o software e quais são as oportunidades de melhoria para mantê-lo engajado.

Ao mapear a jornada do usuário por meio do Canvas de Jornada do Usuário, a equipe de desenvolvimento pode identificar oportunidades de melhoria e criar soluções mais eficientes e intuitivas para o usuário.

Por exemplo, é possível identificar quais são as funcionalidades mais importantes para o usuário e priorizá-las na etapa de desenvolvimento do software. Além disso, é possível identificar problemas de usabilidade ou fluxo de navegação que possam estar impedindo o usuário de ter uma boa experiência com o software.

Suponhamos que o cliente já contratou a solução e está solicitando uma nova feature, como um módulo de controle de estoque. Abaixo, você pode conferir um exemplo de como preencher o Canvas de Jornada do Usuário em uma tabela considerando essa situação:

Etapa	Descrição	Dores e Necessidades	Pontos de Fricção	Oportunidade de Melhoria
DESCOBERTA	O cliente já conhece o software e solicita a nova feature de controle de estoque.	Encontrar uma solução que atenda suas necessidades específicas.	Dificuldade de encontrar a nova feature na plataforma.	Disponibilizar informações claras e acessíveis sobre a nova feature.
AQUISIÇÃO	A nova feature é adicionada ao plano do cliente.	Compreender como funciona a nova feature e como utilizá-la.	Falta de clareza sobre a utilização da nova feature.	Disponibilizar materiais explicativos e agilizar o processo de contato com a equipe de suporte.
ATIVAÇÃO	O cliente realiza a configuração do novo módulo de controle de estoque.	Entender como funciona o novo módulo e como configurá-lo para a loja.	Dificuldade de configurar o software para o módulo de controle de estoque.	Investir em um tutorial intuitivo para a configuração do novo módulo.
UTILIZAÇÃO	Os funcionários da loja utilizam o módulo de controle de estoque.	Facilidade de uso e agilidade no controle do estoque.	Problemas de usabilidade ou lentidão do sistema.	Melhorar a interface e a velocidade do sistema para tornar a utilização mais fluida.
FIDELIZAÇÃO	O cliente utiliza o novo módulo para gerenciar o estoque e tomar decisões estratégicas.	Facilidade de análise e compreensão dos dados.	Dificuldade de visualizar e analisar os dados.	Adicionar recursos de visualização de dados e indicadores de performance para o cliente.

IMPLEMENTAÇÃO

A etapa de implementação é a fase em que a equipe de desenvolvimento de software começa a construir o produto propriamente dito, utilizando as especificações definidas nas fases anteriores.

Em resumo, a etapa de implementação é crucial para o sucesso do projeto de desenvolvimento de software, pois é o momento em que as ideias começam a se tornar realidade. Para garantir o sucesso nessa etapa, é importante manter um alto nível de qualidade, eficiência e comprometimento, e estar sempre em busca de melhorias e otimizações.

METAS E OBJETIVOS

A equipe de desenvolvimento de software tem como principal objetivo transformar as especificações definidas nas fases anteriores em um produto funcional que atenda às necessidades do cliente. Para isso, é importante definir metas e objetivos claros, a fim de garantir a eficiência e qualidade do desenvolvimento do software.

Entre os principais objetivos da etapa de implementação, podemos destacar:

SOFTWARE CONFORME ESPECIFICAÇÕES

A equipe desenvolverá um produto que atenda às especificações definidas nas fases anteriores, como o planejamento, análise e design. O objetivo é construir um

software que atenda às necessidades do cliente e cumpra com as expectativas definidas. É uma fase crucial para o sucesso do projeto, já que é nesse momento que as ideias começam a se tornar realidade. Para isso, é importante que a equipe se dedique a desenvolver um software de alta qualidade, que cumpra com as especificações e atenda às necessidades do cliente.

SEGUIR AS ESPECIFICAÇÕES

É importante utilizar as tecnologias e ferramentas definidas nas fases anteriores do processo de desenvolvimento de software, como no planejamento, análise e design. Isso significa que a equipe deve seguir as especificações definidas anteriormente, utilizando as melhores tecnologias e ferramentas disponíveis para construir o produto.

Por exemplo, se na fase de design foi definido que o software teria uma interface amigável para o usuário, a equipe de desenvolvimento deve utilizar tecnologias de interface gráfica de usuário (GUI) para construir essa interface. Ou ainda, se na fase de análise foi definido que o software deveria ter uma alta escalabilidade, a equipe deve utilizar tecnologias e técnicas de programação que permitam essa escalabilidade.

Dessa forma, utilizar as tecnologias e ferramentas definidas nas etapas anteriores garante que o produto final

seja construído coerentemente com as especificações definidas, atendendo às necessidades do cliente e garantindo a qualidade e eficiência do desenvolvimento do software.

QUALIDADE, EFICIÊNCIA E COMPROMETIMENTO

Quando falamos em manter um alto nível de qualidade e eficiência no desenvolvimento do software durante a etapa de implementação, estamos nos referindo a garantir que o produto final atenda às especificações definidas nas fases anteriores, de forma que seja útil e funcional para o cliente.

Para alcançar esse objetivo, é importante que a equipe de desenvolvimento utilize boas práticas de programação, faça testes automatizados para garantir a qualidade do código produzido, trabalhe colaborativamente e eficiente, e busque sempre por melhorias e otimizações no processo de desenvolvimento.

Assim, manter um alto nível de qualidade e eficiência durante a implementação do software é fundamental para garantir a satisfação do cliente e o sucesso do projeto, evitando retrabalhos e atrasos no cronograma.

ENTREGA NO PRAZO

Cumprir os prazos estabelecidos para a entrega do produto significa entregar o software no prazo determinado

previamente, sem atrasos. Isso é importante para garantir a satisfação do cliente, que espera receber o produto no tempo combinado, além de manter a equipe de desenvolvimento focada e organizada, evitando possíveis atrasos e retrabalhos.

Para cumprir os prazos, é importante que a equipe esteja bem organizada, seguindo um cronograma bem definido e utilizando ferramentas de gerenciamento de projetos para monitorar o progresso do desenvolvimento.

Também é importante estar preparado para lidar com eventuais imprevistos que possam surgir durante o processo, buscando sempre soluções ágeis e eficientes. Cumprir os prazos estabelecidos é um dos principais desafios enfrentados na etapa de implementação, mas é essencial para o sucesso do projeto de desenvolvimento de software.

INDICADORES E KPIs

Na etapa de implementação, alguns indicadores e KPIs podem ser utilizados para monitorar o progresso do desenvolvimento e garantir que as metas e objetivos definidos estejam sendo alcançados.

Entre os principais indicadores e KPIs da etapa de implementação, podemos destacar:

PROGRESSO NO DESENVOLVIMENTO

Para medir o progresso do desenvolvimento, pode-se utilizar ferramentas de gerenciamento de projetos, que permitem acompanhar a evolução do desenvolvimento e identificar possíveis atrasos.

QUALIDADE DO CÓDIGO

Para garantir a qualidade do código produzido, pode-se utilizar testes automatizados e análise estática do código.

SATISFAÇÃO DO CLIENTE

A satisfação do cliente pode ser medida por meio de pesquisas de satisfação e feedbacks recebidos.

CUMPRIMENTO DE PRAZOS

Para garantir o cumprimento dos prazos, pode-se utilizar ferramentas de gerenciamento de tempo e estabelecer marcos de entrega.

PRODUTIVIDADE DA EQUIPE

Para medir a produtividade da equipe, pode-se utilizar indicadores como horas trabalhadas, número de linhas de código produzidas, entre outros.

Ao utilizar indicadores e KPIs, é possível monitorar o progresso do desenvolvimento, identificar possíveis

problemas e buscar soluções eficientes para garantir a eficácia e qualidade do produto final.

PLANO DE AÇÃO

Embora não exista um canvas específico para a etapa de implementação, é importante definir um plano de ação claro e detalhado, estabelecendo metas, prazos e recursos necessários para garantir o sucesso da implementação. Assim, é possível manter a equipe focada e alinhada com os objetivos do projeto, garantindo a eficiência e qualidade do desenvolvimento do software.

ATIVIDADE	RESPONSÁVEL	PRAZO	RECURSOS	OBSERVAÇÕES
DESENVOLVIMENTO DAS FUNCIONALIDADES	Equipe de Desenvolvimento	8 semanas	Ambiente de desenvolvimento, bibliotecas e frameworks necessários.	Será realizado por meio de sprints semanais.
TESTES UNITÁRIOS	Equipe de Qualidade	Durante o desenvolvimento	Ferramentas de teste, ambiente de testes.	Será realizado em conjunto com o desenvolvimento das funcionalidades.
TESTES DE INTEGRAÇÃO	Equipe de Qualidade	9 semanas	Ambiente de testes, ferramentas de teste.	Será realizado após o desenvolvimento das funcionalidades.
HOMOLOGAÇÃO	Cliente	2 semanas	Ambiente de homologação.	Será realizado pelo cliente em conjunto com a equipe de qualidade.
CORREÇÃO DE PROBLEMAS	Equipe de Desenvolvimento	1 semana	Ambiente de desenvolvimento, ferramentas de correção.	Será realizado caso sejam identificados problemas durante a homologação.
IMPLANTAÇÃO	Equipe de Implantação	1 semana	Documentação, ambiente de produção.	Será realizado após a homologação e correção de problemas.
TREINAMENTO	Equipe de Implantação	1 semana	Material de treinamento, ambiente de treinamento.	Será realizado após a implantação.

TESTES

A etapa de teste é uma das mais importantes no processo de desenvolvimento de software, ao ser nela que são identificados e corrigidos possíveis problemas e falhas no sistema antes que ele seja implantado em um ambiente de produção.

Durante a etapa de teste, a equipe de desenvolvimento realiza diversos tipos de testes, como testes de unidade, testes de integração, testes de sistema e testes de aceitação. O objetivo é garantir que o software esteja funcionando adequadamente e que todas as funcionalidades desenvolvidas estejam disponíveis e funcionando corretamente.

Os testes de unidade, por exemplo, são realizados em partes individuais do código, verificando se elas funcionam corretamente. Já os testes de integração verificam se as diferentes partes do sistema se integram corretamente. Os testes de sistema, no que lhe concerne, verificam se o software funciona adequadamente na totalidade. E, por fim, os testes de aceitação são realizados para verificar se o software atende aos requisitos do cliente e se está pronto para ser implantado em um ambiente de produção.

Alguns dos principais objetivos da etapa de teste incluem garantir a qualidade do software, evitar possíveis problemas que afetem a utilização pelos usuários finais e garantir a satisfação do cliente. Por isso, é importante que a equipe de desenvolvimento dedique tempo e recursos para

realizar testes adequados e identificar possíveis problemas antes que o software seja implantado em um ambiente de produção.

OBJETIVOS E METAS

Traçar metas e objetivos é uma etapa fundamental em qualquer processo de desenvolvimento de software, especialmente na etapa de teste. Isso porque, sem objetivos claros, fica difícil avaliar se o software está funcionando adequadamente e atendendo às expectativas dos usuários.

Ao traçar metas e objetivos para a etapa de teste, a equipe de desenvolvimento define os principais pontos que precisam ser avaliados no software, como a funcionalidade, usabilidade, performance e segurança, por exemplo. Além disso, essas metas e objetivos devem estar alinhados com as expectativas dos usuários finais e com as necessidades do negócio.

Ao ter objetivos claros, é possível avaliar se o software está funcionando adequadamente e identificar possíveis falhas e problemas a serem corrigidos. Isso permite que a equipe de desenvolvimento realize ajustes e melhorias antes da implantação, garantindo que o software atenda às expectativas dos usuários e do negócio.

Alguns exemplos de metas e objetivos para esta etapa:

TESTAR TODAS FUNCIONALIDADES

É importante que a equipe de desenvolvimento teste todas as funcionalidades desenvolvidas, para garantir que elas estejam funcionando corretamente e atendendo aos requisitos definidos.

IDENTIFICAR E CORRIGIR PROBLEMAS

Durante os testes, é possível identificar problemas e erros no software, que devem ser corrigidos pela equipe de desenvolvimento para garantir que o software esteja funcionando corretamente.

GARANTIR A QUALIDADE DO SOFTWARE

A etapa de teste é fundamental para garantir a qualidade do software, ao ser nessa etapa que são realizados os testes de desempenho, segurança, usabilidade, entre outros.

PREPARAR PARA IMPLANTAÇÃO

Após os testes serem realizados e os problemas corrigidos, o software estará pronto para ser implantado no ambiente de produção.

INDICADORES E KPIs

Ter indicadores e KPIs (Key Performance Indicators) é fundamental para a etapa de teste no processo de desenvolvimento de software. Esses indicadores permitem avaliar o desempenho do software durante a etapa de testes e identificar possíveis problemas que possam prejudicar sua qualidade e usabilidade.

Os KPIs são métricas utilizadas para medir o desempenho do software e avaliar seu sucesso durante a etapa de testes. Eles ajudam a equipe de desenvolvimento a identificar problemas e oportunidades de melhoria, a fim de garantir que o software esteja funcionando adequadamente e atenda às necessidades dos usuários finais.

Ter indicadores e KPIs é essencial para garantir que o software esteja funcionando adequadamente e atenda às necessidades dos usuários finais. Eles permitem identificar problemas e oportunidades de melhoria, e auxiliam a equipe de desenvolvimento a tomar decisões mais assertivas, visando garantir a qualidade do software e a satisfação do cliente.

TAXA DE BUGS ENCONTRADOS

Mede a quantidade de bugs encontrados durante os testes, indicando a qualidade do software.

TAXA DE BUGS CORRIGIDOS

Mede a quantidade de bugs corrigidos pela equipe de desenvolvimento, indicando a eficiência da equipe na correção de problemas identificados nos testes.

TAXA DE TESTES REALIZADOS

Mede a quantidade de testes realizados pela equipe de desenvolvimento, indicando a cobertura dos testes em relação às funcionalidades desenvolvidas.

TEMPO MÉDIO DE CORREÇÃO DE BUGS

Mede o tempo que a equipe de desenvolvimento leva para corrigir bugs identificados durante os testes, indicando a eficiência da equipe na correção de problemas.

TAXA DE ACEITAÇÃO DO SOFTWARE

Mede a porcentagem de testes realizados com sucesso, ou seja, sem erros ou problemas.

Ao monitorar esses indicadores e KPIs constantemente, é possível garantir que o software esteja funcionando adequadamente e atendendo aos requisitos definidos, o que é fundamental para o sucesso do projeto na totalidade.

TEST CANVAS

O Test Canvas é uma metodologia que ajuda a equipe de desenvolvimento a planejar e executar os testes de software de forma mais eficiente e organizada.

O Test Canvas é dividido em 6 seções: contexto, requisitos, cenários, casos de teste, execução e resultados. Cada seção tem um objetivo específico e auxilia a equipe de desenvolvimento a planejar e executar os testes de forma mais estruturada e eficiente.

Na seção de contexto, são definidos o objetivo do teste e as informações sobre o ambiente de teste. Na seção de requisitos, são listados os requisitos que serão testados. Na seção de cenários, são criados cenários de teste para cada requisito. Na seção de casos de teste, são definidos os casos de teste utilizados para testar cada cenário. Na seção de execução, são detalhados os procedimentos para a execução dos casos de teste. Na seção de resultados, são registrados os resultados obtidos durante os testes.

O Test Canvas ajuda a equipe de desenvolvimento a ter uma visão mais clara dos testes a serem realizados e a monitorar o progresso e resultados dos testes de forma mais organizada e eficiente. Além disso, essa metodologia auxilia a identificar problemas mais rapidamente e a tomar decisões mais assertivas, visando garantir a qualidade do software e a satisfação do cliente.

SEÇÃO	PERGUNTAS	DORES E NECESSIDADES
CONTEXTO	Qual é o objetivo do teste?	Testar a funcionalidade de pagamento com cartão de crédito.
AMBIENTE	Qual é o ambiente de teste?	Ambiente de teste: servidor de testes.
REQUISITOS	Quais são os requisitos que serão testados?	Pagamento com cartão de crédito.
CENÁRIO	Quais são os cenários de teste para cada requisito?	Cenário 1: Inserir informações do cartão de crédito. Cenário 2: Verificar limite do cartão. Cenário 3: Confirmar pagamento.
CASO DE TESTE	Quais são os casos de teste utilizados para testar cada cenário?	CEN01-CT01 — Inserir informações do cartão de crédito inválido. CEN01-CT02 — Inserir informações do cartão de crédito válido. CEN02-CT03 — Verificar limite do cartão disponível. CEN02-CT04 — Verificar limite do cartão excedido. CEN03-CT05 — Confirmar pagamento com sucesso. CEN03-CT06 — Falha ao confirmar pagamento.
EXECUÇÃO	Quais são os procedimentos para a execução dos casos de teste?	Passo 1: Inserir informações do cartão de crédito e selecionar a opção de pagamento. Passo 2: Verificar o limite do cartão. Passo 3: Confirmar o pagamento.
RESULTADO	Quais foram os resultados obtidos durante os testes?	CEN01-CT01 — Falha ao inserir informações do cartão de crédito inválido. CEN01-CT02 — Sucesso ao inserir informações do cartão de crédito válido. CEN02-CT03 — Limite do cartão disponível. CEN02-CT04 — Limite do cartão excedido. CEN03-CT05 — Sucesso ao confirmar pagamento. CEN03-CT06 — Falha ao confirmar pagamento.

IMPLANTAÇÃO

Na etapa de implantação, o software desenvolvido é finalmente implantado em um ambiente de produção, pronto para ser utilizado pelos usuários finais. A meta principal é garantir que a transição do ambiente de desenvolvimento para o ambiente de produção seja realizada com sucesso e que o software esteja funcionando adequadamente para os usuários finais.

METAS E OBJETIVOS

As metas e objetivos da etapa de implementação no processo de desenvolvimento de software incluem:

INSTALAÇÃO SEM ERROS

É importante que todo o processo de instalação seja realizado adequadamente, para evitar possíveis problemas que afetem a utilização do software pelos usuários finais.

FUNCIONALIDADE SEM FALHAS

É necessário verificar se todas as funcionalidades desenvolvidas estão disponíveis e funcionando corretamente no ambiente de produção, para os usuários poderem utilizá-las sem problemas.

TREINAR OS USUÁRIOS

A equipe de desenvolvimento deve preparar materiais explicativos e realizar treinamentos para os usuários finais, para que estes possam utilizar o software adequadamente e aproveitar todas as suas funcionalidades.

TRANSIÇÃO SEM INTERRUPÇÕES

É importante que a transição seja feita transparentemente e que não ocorram perdas de dados durante o processo.

GARANTIR A SATISFAÇÃO DO CLIENTE

O sucesso da etapa de implementação depende também da satisfação do cliente com o produto final entregue.

Ao atingir essas metas e objetivos, a equipe de desenvolvimento garante que o software esteja pronto para ser utilizado pelos usuários finais, com todas as funcionalidades disponíveis e funcionando adequadamente, garantindo o sucesso do projeto na totalidade.

INDICADORES E KPIs

A etapa de implementação do processo de desenvolvimento de software possui alguns indicadores e KPIs importantes, que ajudam a medir a eficiência e o

sucesso da equipe na implantação do software no ambiente de produção. Alguns desses indicadores e KPIs são:

TAXA DE INSTALAÇÃO SEM ERROS

Mede a porcentagem de instalações realizadas sem erros ou problemas, indicando a eficiência da equipe de desenvolvimento na implantação do software no ambiente de produção.

TEMPO DE RESPOSTA DO SISTEMA

Mede o tempo de resposta do sistema após a implantação, ou seja, o tempo que o software leva para processar as solicitações dos usuários. Esse indicador é importante, pois um tempo de resposta muito longo pode indicar problemas de desempenho no software.

TAXA DE ADOÇÃO PELOS USUÁRIOS FINAIS

Mede a porcentagem de usuários finais que estão utilizando o software após a implantação. Esse indicador é importante, ao indicar se o software está sendo adotado pelos usuários finais e se está atendendo às necessidades e expectativas deles.

SATISFAÇÃO DO CLIENTE

Mede a satisfação do cliente com o software após a implantação. Esse indicador é importante, ao indicar se o software atendeu às expectativas do cliente e se está sendo útil para ele.

TAXA DE ERROS OU FALHAS NO SISTEMA

Mede a porcentagem de erros ou falhas no sistema após a implantação. Esse indicador é importante, ao indicar se o software está funcionando corretamente e se está livre de problemas que afetem a sua utilização pelos usuários finais.

Esses indicadores e KPIs ajudam a equipe de desenvolvimento a medir o sucesso da etapa de implantação e a identificar possíveis problemas que afetem a utilização do software pelos usuários finais. Ao monitorar esses indicadores e KPIs constantemente, é possível garantir que o software esteja funcionando adequadamente e atendendo às necessidades dos usuários finais.

Com essas medidas, é possível garantir que o processo de implantação do software seja bem-sucedido, atendendo às necessidades e expectativas dos usuários finais e garantindo a qualidade e a efetividade do produto final.

MANUTENÇÃO

A etapa de manutenção em uma equipe de desenvolvimento de software é essencial para garantir o bom funcionamento do software após a sua implantação. Nessa fase, a equipe de desenvolvimento é responsável por corrigir eventuais problemas, realizar melhorias e garantir que o software atenda às necessidades do usuário.

METAS E OBJETIVOS

As metas e objetivos da etapa de manutenção em uma equipe de desenvolvimento de software incluem:

MANTER A QUALIDADE DO SOFTWARE

A equipe de manutenção deve trabalhar para garantir que o software continue funcionando adequadamente e atendendo às necessidades do usuário. Para isso, é importante corrigir eventuais problemas e realizar melhorias no software quando necessário.

GARANTIR A SATISFAÇÃO DO USUÁRIO

A equipe de manutenção deve estar sempre atenta às necessidades do usuário e trabalhar para garantir a sua satisfação. Para isso, é importante responder prontamente às solicitações dos usuários e trabalhar para melhorar continuamente o software.

MELHORAR A PERFORMANCE DO SOFTWARE

A equipe de manutenção deve trabalhar para garantir que o software esteja sempre funcionando de forma eficiente e rápida. Para isso, é importante identificar e corrigir eventuais problemas de performance e implementar melhorias no software quando necessário.

Essas metas e objetivos são importantes para garantir o sucesso do software após a sua implantação. A equipe de manutenção deve trabalhar para manter a qualidade do software, garantir a satisfação do usuário e melhorar continuamente a performance do software. Ao cumprir essas metas, a equipe pode garantir que o software atenda às necessidades do usuário e seja uma ferramenta eficiente e confiável para o negócio.

INDICADORES E KPIs

Os indicadores e KPIs (Key Performance Indicators) utilizados na etapa de manutenção em uma equipe de desenvolvimento de software são importantes para medir o sucesso da equipe e garantir que o software esteja funcionando adequadamente. Alguns exemplos de indicadores e KPIs que podem ser utilizados incluem:

QUANTIDADE DE BUGS CORRIGIDOS

Esse indicador mede a quantidade de problemas encontrados no software e corrigidos pela equipe de manutenção. Um alto número de bugs corrigidos pode indicar que o software apresenta muitos problemas e precisa de melhorias.

VELOCIDADE DE RESPOSTA ÀS SOLICITAÇÕES DOS USUÁRIOS

Esse indicador mede o tempo que a equipe de manutenção leva para responder às solicitações dos usuários. Um tempo de resposta rápido pode indicar que a equipe está comprometida em garantir a satisfação do usuário.

TAXA DE SATISFAÇÃO DO USUÁRIO

Esse indicador mede o nível de satisfação do usuário com o software. Uma alta taxa de satisfação pode indicar que o software atende às necessidades do usuário e é uma ferramenta eficiente para o negócio.

TEMPO MÉDIO DE RESOLUÇÃO DE PROBLEMAS

Esse indicador mede o tempo que a equipe de manutenção leva para resolver os problemas encontrados no software. Um tempo médio de resolução rápido pode indicar que a equipe está comprometida em garantir a qualidade do software.

Esses são apenas alguns exemplos de indicadores e KPIs que podem ser utilizados na etapa de manutenção em uma equipe de desenvolvimento de software. É importante escolher os indicadores e KPIs que sejam relevantes para o negócio e para as metas da equipe de manutenção, e monitorá-los regularmente para garantir que o software esteja sempre funcionando adequadamente e atendendo às necessidades do usuário.

CANVAS DE MANUTENÇÃO

O Canvas de Manutenção é uma ferramenta que pode ser utilizada para organizar as atividades da equipe de manutenção em uma equipe de desenvolvimento de software. Ele é composto por nove elementos que ajudam a identificar os principais problemas do software, definir as atividades necessárias para corrigir esses problemas e monitorar o progresso da equipe.

Os nove elementos do Canvas de Manutenção são:

PROBLEMAS CONHECIDOS

Essa seção é utilizada para identificar os principais problemas conhecidos do software. Ela pode incluir problemas identificados pelos usuários ou pela equipe de manutenção.

PRIORIZAÇÃO DOS PROBLEMAS

Essa seção é utilizada para priorizar os problemas identificados. A equipe de manutenção pode utilizar critérios como impacto no usuário e tempo necessário para corrigir o problema para definir a priorização.

ATIVIDADES NECESSÁRIAS

Essa seção é utilizada para definir as atividades necessárias para corrigir os problemas identificados. Ela pode incluir atividades como revisão de código, correção de bugs e implementação de melhorias no software.

CRONOGRAMA

Essa seção é utilizada para definir o cronograma das atividades necessárias para corrigir os problemas identificados. Ela ajuda a garantir que a equipe de manutenção esteja sempre trabalhando nas atividades mais importantes e que o software esteja sempre funcionando adequadamente.

ORÇAMENTO

Essa seção é utilizada para definir o orçamento disponível para a equipe de manutenção. Ela ajuda a garantir que as atividades necessárias para corrigir os problemas identificados possam ser realizadas no orçamento disponível.

RECURSOS NECESSÁRIOS

Essa seção é utilizada para definir os recursos necessários para realizar as atividades necessárias para corrigir os problemas identificados. Ela pode incluir recursos como pessoal, hardware e software.

RESPONSABILIDADES

Essa seção é utilizada para definir as responsabilidades de cada membro da equipe de manutenção. Ela ajuda a garantir que todos os membros da equipe estejam cientes das suas responsabilidades e trabalhem de forma eficiente.

INDICADORES DE DESEMPENHO

Essa seção é utilizada para definir os indicadores de desempenho utilizados para monitorar o progresso da equipe de manutenção. Isso pode incluir indicadores como quantidade de bugs corrigidos e tempo médio de resolução de problemas.

RESULTADOS ESPERADOS

Essa seção é utilizada para definir os resultados esperados da equipe de manutenção. Ela ajuda a garantir que a equipe esteja trabalhando para atender às necessidades do

usuário e garantir que o software esteja sempre funcionando adequadamente.

O Canvas de Manutenção é uma ferramenta bastante útil para organizar as atividades da equipe de manutenção em uma equipe de desenvolvimento de software. Ele ajuda a identificar os principais problemas do software, definir as atividades necessárias para corrigir esses problemas e monitorar o progresso da equipe. Com isso, é possível garantir que o software esteja sempre funcionando adequadamente e atendendo às necessidades do usuário.

ELEMENTO	DESCRIÇÃO
PROBLEMAS CONHECIDOS	Lentidão ao acessar relatórios financeiros.
PRIORIZAÇÃO DOS PROBLEMAS	Alto impacto no usuário, médio tempo necessário para corrigir.
ATIVIDADES NECESSÁRIAS	Otimização do banco de dados, revisão do código dos relatórios financeiros.
CRONOGRAMA	Otimização do banco de dados: duas semanas. Revisão do código dos relatórios financeiros: quatro semanas.
ORÇAMENTO	R$10.000,00 disponíveis para realizar as atividades necessárias.
RECURSOS NECESSÁRIOS	Dois desenvolvedores, acesso ao ambiente de produção.
RESPONSABILIDADES	Desenvolvedor 1: otimização do banco de dados. Desenvolvedor 2: revisão do código dos relatórios financeiros.
INDICADORES DE DESEMPENHO	Quantidade de relatórios financeiros otimizados, tempo médio de resposta do usuário.
RESULTADOS ESPERADOS	Acesso mais rápido aos relatórios financeiros, aumento da satisfação do usuário.

Em resumo, a etapa de manutenção em uma equipe de desenvolvimento de software é crucial para garantir o sucesso do software após a sua implantação. A equipe deve trabalhar para manter a qualidade do software, corrigir eventuais problemas, realizar melhorias e garantir a satisfação do usuário. Para isso, é importante utilizar indicadores e KPIs para medir o sucesso da equipe e trabalhar continuamente para melhorar o software e atender às necessidades do usuário.

PLANO DE AÇÃO DE 30 DIAS

Aqui está um plano de ação de 30 dias que pode ajudar a organizar o time de desenvolvimento da sua Software house:

SEMANA 1

LEVANTAMENTO DE INFORMAÇÕES E DEFINIÇÃO DE OBJETIVOS

- Realizar reuniões com os membros da equipe de desenvolvimento para entender as principais dificuldades e problemas enfrentados pela equipe.

- Definir objetivos claros e mensuráveis para a equipe de desenvolvimento, alinhados com os objetivos estratégicos da Software house.

SEMANA 2

ANÁLISE E DEFINIÇÃO DE PROCESSOS

- Analisar os processos atuais da equipe de desenvolvimento e identificar oportunidades de melhoria.

- Definir novos processos para a equipe de desenvolvimento, visando aumentar a eficiência e qualidade do trabalho.

SEMANA 3

IMPLEMENTAÇÃO DE FERRAMENTAS DE GESTÃO DE PROJETOS

- Selecionar e implementar uma ferramenta de gestão de projetos para a equipe de desenvolvimento.

- Capacitar a equipe de desenvolvimento na utilização da ferramenta selecionada.

SEMANA 4

DEFINIÇÃO DE INDICADORES DE DESEMPENHO E MONITORAMENTO

- Definir indicadores de desempenho para a equipe de desenvolvimento, como tempo médio de resolução de problemas e quantidade de bugs corrigidos.

- Implementar um sistema de monitoramento para acompanhar o desempenho da equipe de desenvolvimento em relação aos indicadores definidos.

Ao final dessas quatro semanas, a sua equipe de desenvolvimento estará mais organizada e alinhada com os objetivos estratégicos da Software house. A implementação de ferramentas de gestão de projetos e a definição de indicadores de desempenho ajudarão a aumentar a eficiência e qualidade do trabalho da equipe. Lembre-se de que a

organização é um processo contínuo, e é importante continuar acompanhando e ajustando os processos e indicadores ao longo do tempo.

REFERÊNCIAS
BIBLIOGRAFIA E FONTES CONSULTADAS:

"Engenharia de Software: uma abordagem profissional": https://www.amazon.com.br/Engenharia-Software-Roger-S-Pressman/dp/0073375977

"The Mythical Man-Month: Essays on Software Engineering": https://www.amazon.com.br/Mythical-Man-Month-Software-Engineering-Anniversary/dp/0201835959

"Agile Software Development, Principles, Patterns, and Practices": https://www.amazon.com.br/Agile-Software-Development-Principles-Practices/dp/0135974445

"Clean Code: A Handbook of Agile Software Craftsmanship": https://www.amazon.com.br/Clean-Code-Handbook-Software-Craftsmanship/dp/0132350882

"Software Engineering: A Practitioner's Approach": https://www.amazon.com.br/Software-Engineering-Practitioners-Approach-Pressman/dp/1259872995

"Effective DevOps: Building a Culture of Collaboration, Affinity, and Tooling at Scale": https://www.amazon.com.br/Effective-DevOps-Building-Collaboration-Affinity/dp/1491926309

"Scrum: A Arte de Fazer o Dobro do Trabalho na Metade do Tempo":
https://www.amazon.com.br/Scrum-Arte-Dobro-Trabalho-Metade/dp/8576086476

"Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation":
https://www.amazon.com.br/Continuous-Delivery-Deployment-Automation-Addison-Wesley/dp/0321601912

"Managing the Unmanageable: Rules, Tools, and Insights for Managing Software People and Teams":
https://www.amazon.com.br/Managing-Unmanageable-Insights-Software-People/dp/032182203X

"The Phoenix Project: A Novel About IT, DevOps, and Helping Your Business Win":
https://www.amazon.com.br/Phoenix-Project-DevOps-Helping-Business/dp/1942788290

"The DevOps Handbook: How to Create World-Class Agility, Reliability, and Security in Technology Organizations":
https://www.amazon.com.br/DevOps-Handbook-World-Class-Reliability-Organizations/dp/1942788002

"User Stories Applied: For Agile Software Development":

https://www.amazon.com.br/User-Stories-Applied-Software-Development/dp/0321205685

"Domain-Driven Design: Tackling Complexity in the Heart of Software":
https://www.amazon.com.br/Domain-Driven-Design-Tackling-Complexity-Software/dp/0321125215

"Patterns of Enterprise Application Architecture": https://www.amazon.com.br/Patterns-Enterprise-Application-Architecture-Martin/dp/0321127420

"Refactoring: Improving the Design of Existing Code": https://www.amazon.com.br/Refactoring-Improving-Existing-Addison-Wesley-Technology/dp/0134757599

"Software Requirements": https://www.amazon.com.br/Software-Requirements-Karl-E-Wiegers/dp/0735679665

"Code Complete: A Practical Handbook of Software Construction":
https://www.amazon.com.br/Code-Complete-Practical-Handbook-Construction/dp/0735619670