Desenvolvimento de Sistemas

Projeto de *software*: conceito e etapas

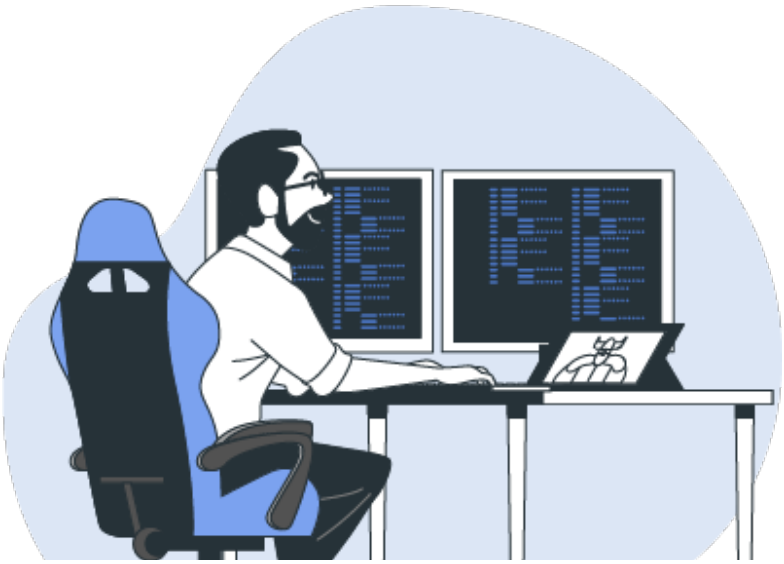
Imagine o seguinte contexto: você recebe uma visita do seu tio Sérgio, aquele senhor gentil que participa de todas as festas familiares de final de ano. Entre uma conversa e outra, ele finalmente descobre que você está estudando programação. Empolgado, ele fala para você que quer um sistema para controle de estoque da empresa de venda de peças automotivas que ele tem. Você também fica empolgado com a ideia de um novo projeto e aceita rapidamente. Mas e agora? Como você realizará esse desenvolvimento? O que o tio Sérgio espera do sistema? Será que ele vai saber utilizar um sistema muito tecnológico? Quais informações não podem faltar para que não haja problemas no estoque? Quais serão os seus primeiros passos para lidar com essa oportunidade?

Diversas atividades durante a vida requerem planejamento. A falta de organização e a obtenção de informações insuficientes podem acabar gerando despesas extras, atrasos ou até mesmo o esquecimento de detalhes importantes ao longo do processo. Se você organizar uma festa e não considerar a quantidade de pessoas na hora de comprar bebidas, por exemplo, isso pode gerar a insatisfação de seus convidados! Com o desenvolvimento de um *software* não é diferente. Sem as informações necessárias, você pode acabar não proporcionando um resultado satisfatório ao seu cliente.

Você consegue perceber importância da realização de um projeto?

A seguir, você aprenderá um pouco mais sobre projeto de *software* e saberá como proceder no caso do tio Sérgio!

Projeto de *software*: conceito



O que é um projeto de *software*?

Um projeto de *software* nada mais é do que um planejamento que é iniciado antes do desenvolvimento de um novo sistema. É por meio desse planejamento que se consegue elaborar uma sequência de eventos com início, meio e fim, definindo aspectos importantes para execução do projeto e controlando o processo de desenvolvimento. Um bom projeto de *software* garante que os objetivos sejam atingidos com a menor quantidade de trabalho possível, sem deixar pra trás nenhum detalhe. É assim que se deixa um cliente satisfeito, não é mesmo?

Os processos que compõem o gerenciamento do projeto podem ser observados no mapa mental a seguir:

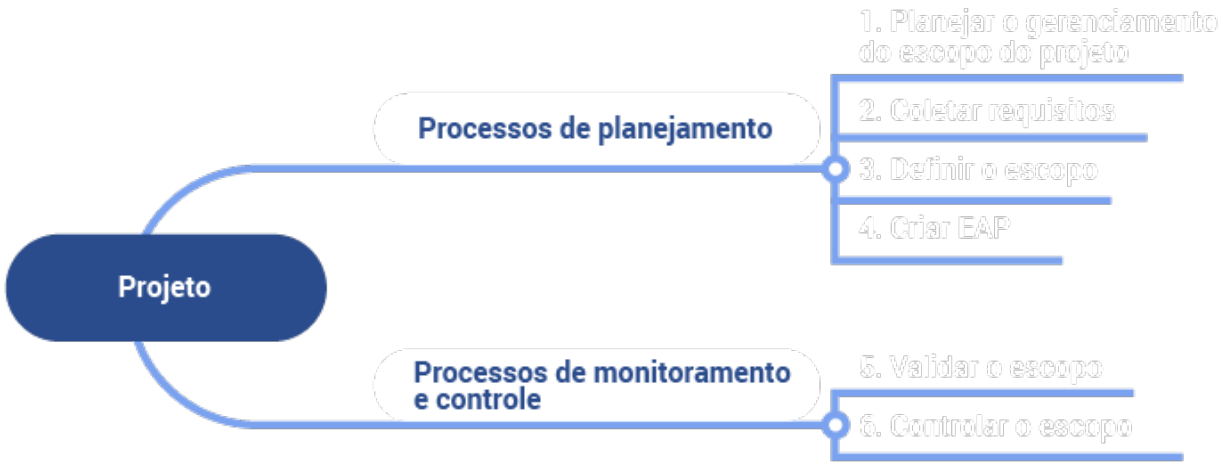


Figura 1 – Mapa mental planejamento de *software*

Fonte: adaptado de Vargas (2018)

Cada uma das etapas anteriores poderá gerar documentos úteis, que serão utilizados como guia durante sua jornada, gerando uma base de informações e ações a serem seguidas na hora de desenvolver o código. Lembre-se de que um projeto bem definido torna o desenvolvimento mais rápido, fácil e ainda auxilia a formalizar as decisões tomadas com o cliente.

Etapas de desenvolvimento

Agora, você explorará cada um dos processos encontrados em um projeto de *software* por meio de uma visão mais gerencial. Confira!

1. Planejar o gerenciamento do escopo do projeto Considerado como parte do planejamento, esse item é processo inicial do projeto. É nessa etapa que serão definidas informações mais amplas da ideia do projeto, o que o cliente deseja, de que modo será organizado, que tipo de ferramentas de controle serão utilizadas, quais serão os integrantes que pertencerão ao projeto, como o cliente validará o progresso do desenvolvimento, entre outros. Nessa fase, tem-se como resultado documentos como o Termo de abertura do projeto e o Plano de gerenciamento do projeto. Considerando o caso do tio Sérgio, é neste momento que você estaria questionando e registrando informações mais amplas sobre como funciona o estoque na empresa dele, enquanto já está pensando no tipo de tecnologia que será necessário, quanto tempo o desenvolvimento pode levar em média ou quantas pessoas serão necessárias considerando o contexto geral do projeto. 2. Coletar os requisitos Também considerada como parte do planejamento, nesta etapa você tratará do gerenciamento dos processos que ajudarão a documentar e determinar as necessidades e os requisitos do cliente. A coleta de requisitos trata-se da obtenção das características e funções que o cliente deseja para o *software*. Para um melhor planejamento do *software*, é importante definir os meios pelos quais o levantamento de requisitos será realizado, as datas e os participantes desse processo. Por meio dessa atividade será gerada a documentação de requisitos. Ainda utilizando o exemplo do tio Sérgio, nessa etapa você deveria questioná-lo mais detalhadamente sobre como funcionam atualmente os processos dentro do setor de estoque. Qualquer pessoa pode alterar o estoque? É necessário um registro das movimentações do estoque? Caso seja necessário, como essa movimentação é realizada? É necessário ter muita desenvoltura e boa percepção para obter informações relevantes para realizar o planejamento. Se você sentir necessidade, pode solicitar a presença de mais especialistas para ter uma melhor perspectiva das necessidades do cliente. Caso o seu cliente queira uma interface gráfica de usuário bem moderna, talvez seja interessante pedir auxílio de um *designer* para idealizar melhor o que está sendo solicitado. Analise a situação em que você se encontra para buscar formas de obter a melhor perspectiva possível. Esta etapa dará início à documentação de requisitos. As formas de coletar requisitos com eficácia serão apresentadas no conteúdo sobre as regras de negócio. 3. Definir o escopo Também considerada como parte do planejamento, a definição de escopo é como uma sequência direta do processo de coletar os requisitos, porém, de uma forma mais detalhada, delimitando com precisão o que será ou não feito no sistema. Neste momento, utilizando as informações coletadas anteriormente,

você desenvolverá um documento contendo com detalhes cada uma das funções, como serão feitas, as estimativas de entregas, premissas e restrições. É muito importante que essa parte seja bem elaborada, afinal, todo seu projeto será realizado utilizando essas definições como base. Certifique-se de apresentar uma cópia desse documento ao cliente como modo de confirmação e validação dos dados que foram definidos para o projeto. O projeto deverá começar somente após esse documento ser aprovado pelo cliente, afinal, desta forma você evita qualquer possível confusão após a entrega do produto, garantindo que todos os interessados estejam cientes do que foi combinado e que o *software* será desenvolvido de acordo com o contratado. É neste momento que você confirmaria com o tio Sérgio a estrutura do *software* que ele deseja. 4. Criar a EAP (estrutura analítica do projeto) Ainda dentro da parte de planejamento, nessa etapa é realizado o processo de subdividir as entregas e o trabalho do projeto em componentes menores, fazendo com que se torne mais fácil o gerenciamento de todo o desenvolvimento. Você utilizará como base os documentos desenvolvidos nas etapas anteriores, portanto, é imprescindível que esses documentos estejam bem completos para facilitar a subdivisão dentro do prazo de tempo previamente definido. Vale lembrar também que uma boa subdivisão de tarefas diminui a chance de alguma parte ser esquecida, portanto, fique de olho nessa etapa quando estiver gerenciando seu projeto! Comunicar-se com todos os desenvolvedores do projeto pode ajudar bastante a organizar as entregas de modo condizente e evitar atrasos. Delegar as atividades à equipe equilibradamente é importante para que as atividades sejam entregues em datas correspondentes aos prazos estipulados. Para iniciar a programação, documentos que auxiliem a definir o sistema são muito bem-vindos. Alguns exemplos desses documentos, os quais serão abordados ao longo do curso, são diagramas de casos de uso, diagramas de entidade e relacionamento e diagramas de classes e objetos. 5. Validar o escopo Quanto à parte de monitoramento e controle, esse processo busca formalizar a aceitação das entregas concluídas do projeto. Basicamente, trata-se de testar se as funcionalidades prontas estão funcionando corretamente. Caso não estejam funcionando tal como o solicitado pelo cliente ou apresentem algum *bug* no sistema, o código deverá ser corrigido até que atenda plenamente o que foi solicitado. Essa etapa funciona como uma inspeção das atividades realizadas, confirmando o sucesso no desenvolvimento e observando o desempenho de trabalho dos colaboradores. Qualquer alteração realizada nessa etapa deve gerar também a alteração nos documentos correspondentes ao trecho do projeto que foi alterado. 6. Controlar o escopo Ainda na parte de monitoramento e controle, essa etapa trata-se do monitoramento do andamento do projeto, do produto e das mudanças que aconteceram ao longo do desenvolvimento. Aqui, será verificado se a equipe está trabalhando de acordo com o cronograma definido. Se ela não estiver, será necessário analisar a razão das mudanças para tentar voltar ao planejamento base (ou alterar, se necessário). Esse processo seguirá até o fim do desenvolvimento, ou seja, até o momento em que o *software* esteja pronto, validado pelo cliente e entregue como concluído. Com a utilização de um bom controle de escopo, você não desperdiçará tempo ou recursos e seu projeto estará pronto dentro do prazo estipulado. Aposto que o tio Sérgio ficará satisfeito com o seu projeto!

Ter um projeto de *software* bem estruturado ajudará você a ter melhor noção sobre qual é o ciclo de vida do seu projeto. Mas o que é, afinal, um ciclo de vida de um *software*? Você já tem alguma ideia? Observe o conceito a seguir.

Agora que você já aprendeu as etapas, como lidaria com a história apresentada no início?

Faça uma lista dos processos que você faria, dados que tentaria coletar com seu cliente e como seria seu projeto de *software*. Simule a história como se fosse seu próprio cliente e idealize seu projeto!

Ciclo de vida do *software*

O ciclo de vida de um *software* é uma ordem de acontecimentos dos processos e das atividades que fazem parte do desenvolvimento. Observar o ciclo de vida auxilia a elaborar um melhor projeto de *software* e também contribui para a definição de um modelo de desenvolvimento a ser seguido (modelos e metodologias serão abordadas ao longo desta unidade curricular). Você pode resumir este ciclo em três etapas: definição, desenvolvimento e implementação. Confira uma análise detalhada de cada uma das fases que fazem parte de um ciclo de vida do *software*:

Fase de requisitos

É na fase de requisitos que você entrará em contato com o problema pela primeira vez. Como definido anteriormente por meio do projeto de *software*, aqui você analisará, com o levantamento de requisitos, se é possível atender à demanda do cliente e quais são as suas dificuldades. O levantamento de requisitos costuma ser feito em conjunto: com o gerente de projetos (ou analista do negócio) e com um engenheiro de sistema ou desenvolvedor, visando a obter mais perspectivas de um mesmo problema. Nessa etapa,

é possível definir quais serão: modelo do projeto, requisitos funcionais (funcionalidades do sistema) e não funcionais (características do sistema), regra de negócios (políticas e normas que devem ser aplicadas ao sistema). Essas definições podem ocorrer pelo uso de técnicas específicas de levantamento de requisitos, tais como entrevistas, questionários etc.

Modos de coletar requisitos com eficácia serão apresentadas no conteúdo sobre as regras de negócio.

Fase de projeto

Nessa fase, você pode utilizar as informações obtidas anteriormente para gerar protótipos do projeto (por meio do uso do Figma, por exemplo), definir especificamente tecnologias e padrões a serem utilizados e outras informações que você observou anteriormente nas etapas de projeto de *software*. É muito importante adequar seus conhecimentos às necessidades do cliente e buscar aprofundar-se sobre elas. A definição correta desses dados requer muita atenção, afinal, todo o andamento do projeto será desenvolvido a partir disso.

Em um projeto de *software* orientado a objetos, nos quais se tem várias estruturas com atributos e funções (que servem como modelo para criação de objetos), é comum que, nesta fase, um arquiteto de *software* faça o levantamento das classes de objetos utilizados nos sistemas, subsistemas e seus segmentos, avaliando os recursos disponíveis. É comum também que sejam modeladas as relações de cada parte do sistema para que o banco de dados seja projetado.

Fase da implementação

Essa é a fase mais longa do projeto e na qual se inicia objetivamente a parte de

codificação do sistema. Com base no que foi projetado nas etapas anteriores, é iniciado o desenvolvimento do sistema na linguagem escolhida. Agora finalmente será a hora de dividir os requisitos e desenvolvê-los na prática. Neste momento, as metodologias de desenvolvimento de *software* que foram definidas anteriormente começam a ser executadas. Existem diversas ferramentas que podem ser utilizadas para auxiliar no gerenciamento dessa parte do processo e elas facilitam muito a organização e o desempenho da equipe de desenvolvedores. Além disso, essas ferramentas também ajudam a gerenciar o projeto de *software* na etapa de controle de escopo, como mencionado anteriormente nas etapas do projeto de *software*. As metodologias e ferramentas que contribuem nessa etapa serão apresentadas nos conteúdos sobre as metodologias de desenvolvimento de *software* e sobre as ferramentas de apoio a projetos de *software*.

Fase de testes

Após todo o desenvolvimento, você chega na etapa de testes! Essa etapa será o momento para validar se todos os requisitos solicitados pelo cliente foram desenvolvidos e se estão funcionando corretamente. Esses testes podem ocorrer em duas etapas: caixa-branca e caixa-preta. O primeiro tipo de teste, caixa-branca, baseia-se no conhecimento da estrutura do programa, ou seja, tendo acesso ao código-fonte. Possíveis falhas no sistema são testadas via código, tal como fluxo de controle, fluxo dos dados, funções executadas etc. Já o teste caixa-preta baseia-se na visão do usuário, considerando a falta de acesso ao código-fonte do sistema. Dessa forma, os testes realizados são derivados das especificações definidas ao sistema. A aplicação desses testes deverá resultar em um relatório contendo uma listagem de sucessos e falhas para que, posteriormente, a manutenção do sistema possa ser realizada, tornando mais fácil identificar onde se encontram os problemas. A finalização dessa etapa acontece por meio da correção dos erros encontrados, resultando no produto final, o *software* funcional.

Fase de produção

Essa fase é marcada pela apresentação do resultado final do *software* ao cliente. Neste momento, todos os ajustes necessários serão realizados, tais como instalar o sistema no ambiente do usuário, importar dados do cliente, corrigir problemas ou até mesmo tirar dúvidas. Em alguns casos, pode ser necessário auxiliar o cliente com um treinamento de uso de sistema. Uma boa opção para prestar esse auxílio é a elaboração de um manual de uso de *software* para o usuário. O manual do usuário tem como objetivo apresentar o funcionamento do produto e o modo de utilização, auxiliando a identificar possíveis dificuldades que o usuário possa ter e realizando a “tradução” de conceitos técnicos para uma linguagem mais acessível a todos os interessados. Esse documento pode reduzir a necessidade de contato com o suporte para solucionar dúvidas básicas e também melhora a relação entre a empresa e o cliente. É importante que ele seja elaborado no conceito de *user friendly*, que pode ser entendido como “usabilidade amigável”, ou seja, facilitado para o usuário, baseado em princípios de fácil compreensão, estimulando o aprendizado. Após todos esses passos, você finalmente estará pronto para concluir este projeto, faltando apenas o termo de encerramento de projeto. Esse documento busca registrar a satisfação do cliente com o desenvolvimento do projeto, verificar as entregas e documentações e finalmente formalizar a aceitação das entregas.

Documentação para clientes

Após acompanhar os processos que acontecem durante um projeto de *software*, você deve ter percebido que diversas etapas acabam gerando documentação, não é mesmo?

Embora grande parte da documentação seja somente técnica, outras são apresentadas ao cliente. Para apresentar documentação, é muito importante que você tome alguns cuidados na hora de redigir.

Antes de elaborar uma proposta, pense no perfil do seu cliente. Existem clientes em potencial que têm conhecimentos mais técnicos, ou seja, que são adaptados ao mercado tecnológico. Em contrapartida, existem potenciais clientes que não entendem os termos técnicos utilizados durante o processo de desenvolvimento. É imprescindível que você saiba para quem você está oferecendo seu serviço. Uma proposta de trabalho, por mais completa que esteja, caso apresente muitos termos técnicos, poderá tornar inseguro um cliente que não tem tanto conhecimento na área, aumentando a necessidade de entrar em contato várias vezes para entender melhor o que está sendo desenvolvido, havendo chances de nem tudo ficar esclarecido. Já o contrato com algumas empresas pode acabar requerendo propostas mais rápidas de ler, que contenham informações mais básicas, como preço, prazo e garantias de um bom serviço. Mesmo que uma proposta longa e detalhada proporcione maior segurança, evitando o risco de ter que trabalhar mais que o combinado devido à falta de detalhes, se você não adaptar o conteúdo ao tipo de cliente, pode deixar a proposta muito rígida e acabar diminuindo o interesse do seu possível cliente, fazendo com que ele não feche a solicitação do projeto. É importante que você seja mais flexível e adapte-se aos problemas; que consiga expor suas condições de trabalho ao mesmo tempo que adapta seus documentos ao cliente.

Elabore propostas longas e detalhadas quando seu cliente tiver um nível técnico parecido com o seu. Assim, você poderá mostrar domínio na área e tomar os cuidados adequados, afinal, um *prospect* (potencial cliente que demonstrou interesse no seu serviço) mais técnico tende a ser mais crítico, exigente e detalhista. No caso do tio Sérgio, mencionado no início do conteúdo, que tipo de contrato se encaixaria melhor? Uma documentação com linguagem mais simplificada ou uma documentação com termos detalhados? Avalie sempre seus clientes!

Desenvolvedor *freelancer* e o projeto de *software*

Caso você, desenvolvedor, decida trabalhar de modo autônomo, toda a organização de documentos, levantamento de requisitos e projeto do *software* ficará em suas mãos. Isso é uma grande responsabilidade.

Você representará diversos papéis na hora de gerenciar o seu projeto, portanto, é muito importante dar atenção aos seguintes aspectos:

Priorizar itens de maior relevância

Você pode finalizar muitas tarefas que planejou, entretanto, seu foco maior deverá ser os aspectos que apresentam maior impacto no seu projeto. Gastar muito tempo caprichando naqueles detalhes de tarefas que já estão concluídas pode acabar atrasando e distraindo você de necessidades maiores. Organize-se para finalizar primeiramente a tarefa mais importante ou então finalizar rapidamente todas as tarefas que não são essenciais. Isso pode auxiliá-lo a deixar tempo para o que é mais importante. Essa escolha pode variar conforme sua necessidade e o projeto no qual você está envolvido e é necessário desenvolver pensamento analítico para observar quais são as maiores necessidades. Como você já viu anteriormente neste conteúdo, um bom projeto de *software* pode auxiliar bastante nesta organização.

Observar suas dificuldades e facilidades ao dividir tarefas

Sem ter uma equipe com você para trabalhar em um desenvolvimento, um grande projeto poderá se tornar intimidador. Isso pode acontecer tanto por você não estar familiarizado com o que foi solicitado quanto por ser um escopo muito grande. As razões podem ser diversas, portanto, dividir uma tarefa em partes menores auxilia na organização e na melhor observação de seu próprio progresso. Quando se está em uma equipe, é possível realizar as divisões de modo mais confortável, porém, se você não tiver uma equipe para auxiliá-lo no desenvolvimento, adapte com maior cuidado a ordem de tarefas, observando suas dificuldades e facilidades para que possa separar tempo suficiente para cada tarefa.

Gerenciar seu tempo

Mesmo que dividir um projeto em pequenas partes possa auxiliar na observação de progresso, é importante definir tempo-limite para as atividades, afinal, você estará controlando o próprio projeto e não terá ninguém para relembrá-lo sobre seus prazos de entrega. Existem diversas formas de gerenciar seu próprio cronograma. Muitas ferramentas e técnicas de gerenciamento que são utilizadas em empresas podem também auxiliar quem está trabalhando como *freelancer*. Você poderá encontrar algumas dessas ferramentas no conteúdo sobre ferramentas de apoio a projetos de *software*. Foque na conclusão da atividade mesmo que não esteja perfeito. Após concluir, você poderá aprimorar seu código conforme desejar, com o tempo que estiver sobrando.

Atualizar seus projetos constantemente

Para um melhor controle das suas atividades, atualize o progresso realizado, afinal, quem estará responsável pela execução das tarefas e da apresentação de resultados diretamente ao cliente será você mesmo. Atualize tanto as tarefas concluídas quanto as solicitações extras do cliente, a mudança de prazos etc. Todas essas informações

auxiliam você a ter um controle maior do que está fazendo e a se manter organizado em suas prioridades. Você pode realizar essas atualizações nas ferramentas de apoio a projeto de *software*.

Todos os aspectos mencionados requerem uma maior autodisciplina quando você se torna um *freelancer*. Corte distrações e tenha foco no projeto e no desenvolvimento de suas tarefas, adaptando as etapas padrão conforme suas necessidades como *freelancer*.

Riscos de projeto: principais erros e como evitá-los

Não seguir as etapas de um projeto de *software* pode levar a equipe a repetir erros e dificultar a definição de prazo de entrega e orçamento. Também pode causar retrabalho e dificuldade em implementar boas práticas e em prevenir problemas no sistema. Não é difícil imaginar os prejuízos que a falta de planejamento pode gerar, entretanto, você já refletiu sobre os possíveis riscos de falha enquanto se aplicam as etapas de um projeto de *software*?

Observe a seguir os possíveis riscos:

Falta de monitoramento sistemático do projeto

Embora existam diversas etapas a serem realizadas durante um projeto de *software*, não basta somente defini-las teoricamente. Sem monitorar as etapas e metodologias que deveriam estar sendo aplicadas, todo projeto se torna em vão. Projetar um *software* significa colocar em prática o que foi definido para alcançar os melhores resultados. O monitoramento durante todas as etapas é essencial. Fique sempre atento.

Má-comunicação entre a equipe e o gerente de projetos

Em conjunto ao monitoramento, é muito importante que o planejamento definido seja analisado por toda a equipe que participará do projeto. A equipe deve concordar com o que está sendo elaborado de modo que cada membro compreenda seu papel durante o projeto. Além disso, também é muito importante que o gerente de *software* escolhido seja uma pessoa que tenha o respeito da equipe, saiba coordenar as atividades e tenha conhecimento geral da empresa e dos processos aplicados. Seja você o gerente de *software* ou um membro da equipe de desenvolvimento, expresse seu ponto de vista para auxiliar no progresso da equipe como um todo. Comunicação é essencial.

Falta de gestão de riscos

Ainda que o projeto esteja bem estruturado, sempre existem riscos durante um desenvolvimento de *software*, os quais podem ser uma mudança inesperada na equipe, a falta de conhecimento da tecnologia por parte da equipe, resistência dos usuários ao uso do sistema, mudança de algum requisito ou até mesmo estimativa errada do cronograma do projeto. Os motivos são diversos. Analise os possíveis riscos do projeto para poder definir um tratamento de risco equivalente e não ter prejuízos ao longo do desenvolvimento por alguma situação prejudicial que tenha surgido.

Falta de comunicação com o cliente

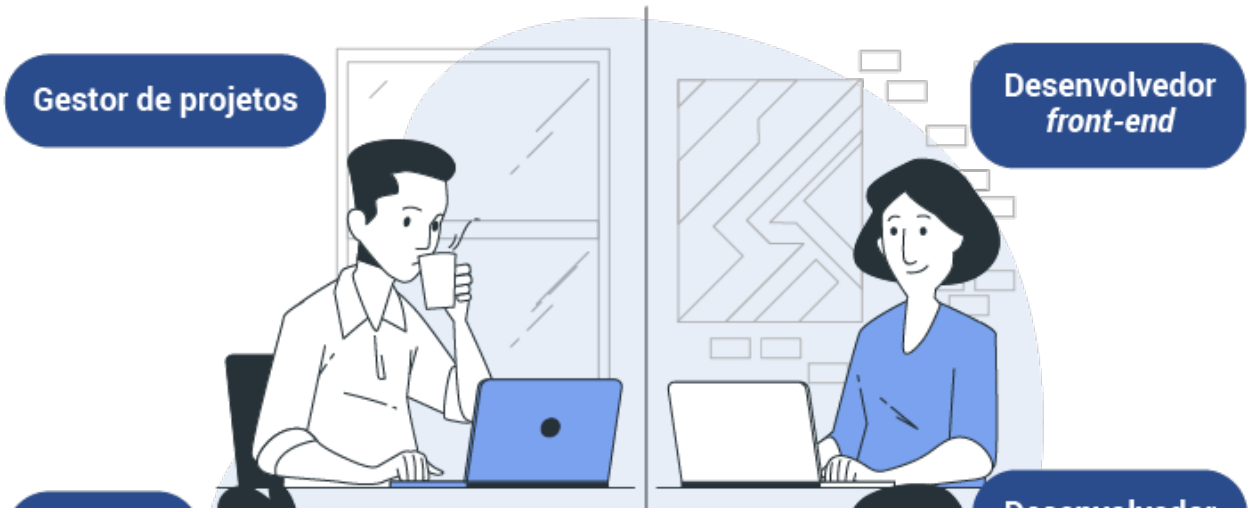
Embora todos os requisitos tenham sido definidos no início do projeto, é importante que o contato com o cliente permaneça durante o desenvolvimento para confirmar que o projeto está de acordo com o solicitado e para sanar qualquer inquietação que possa surgir. É necessário haver um equilíbrio entre informações repassadas ao cliente a fim de deixá-lo informado com atualizações do sistema e ao mesmo tempo não preocupá-lo com problemas banais do cotidiano.

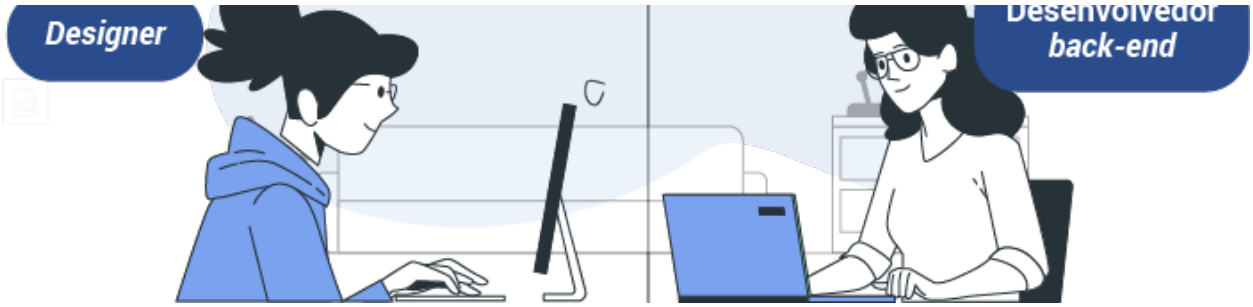
Esses são alguns exemplos de riscos que podem acontecer no seu ambiente de trabalho. Seja como parte da equipe de desenvolvimento, seja como gerenciador de projetos, observe bem os possíveis riscos. Você consegue imaginar melhor as possíveis dificuldades que pode encontrar no ambiente de trabalho? Uma visão analítica do contexto em que você se encontra pode tornar mais fácil preparar-se para qualquer problema! Atente-se aos riscos e ajude sua equipe em direção ao sucesso.

Equipe e cultura organizacional

Ao trabalhar em uma empresa, você fará parte de uma cultura organizacional. Cada organização tem seu próprio modo de lidar com problemas, dificuldades, sucessos e outras situações que possam surgir. Esses modos costumam ser passados entre funcionários, dos antigos aos mais novos, sendo criada assim uma cultura organizacional. É comum que cada empresa seja diferente, entretanto, é necessário que você seja capaz de se adaptar diante do seu novo ambiente. Acima de tudo, é muito importante que você saiba trabalhar em equipe. O que torna um grupo de pessoas uma equipe é o trabalho em conjunto para atingir um objetivo em comum, afinal, mesmo com tarefas bem definidas para cada membro da equipe, caso sejam executadas apenas com uma visão individual, muitos conflitos poderão ser gerados pela falta de sincronia.

Para auxiliar no processo de adaptação e integração, é importante que você conheça as funções dos possíveis membros de sua equipe. Observe a seguir:





Gestor de projetos

Como você deve ter observado ao longo deste conteúdo, o gestor de projetos é o responsável por acompanhar todas as etapas do projeto, monitorando se estão sendo realizadas de acordo com as definições. Esse profissional realiza grande parte da comunicação entre o cliente e o restante da equipe, atendendo a solicitações.

Designer

Esse profissional realizará o planejamento da interface gráfica do usuário (GUI – *graphical user interface*), definindo paleta de cores, ícones, espaçamento. Além disso, é possível que ele auxilie também no desenvolvimento de artes de publicidade para o *marketing* da empresa.

Desenvolvedor *front-end*

No momento de colocar em prática o que foi planejado pelo *designer*, o desenvolvedor *front-end* entra em ação. O profissional dessa área desenvolverá a parte do código direcionada ao usuário, aplicando seus conhecimentos sobre *user experience* (UX – experiência do usuário). Essa é a interface que proporcionará a interação entre o

usuário e o sistema.

Desenvolvedor *back-end*

Quando se trata de regras de negócio, armazenamento de dados e requisições do usuário, você encontrará o desenvolvedor *back-end* como responsável. Ele não participará da parte visual do sistema, porém estará envolvido na comunicação da interface com a realização das funções do sistema.

Essa divisão de funções é originalmente vinculada ao desenvolvimento de sistemas *web*, no qual ocorre a comunicação do seu computador (*front-end*) com o computador do servidor (*back-end*).

Analista de testes

Esse profissional verificará se o sistema foi desenvolvido corretamente. Ele será responsável pelo controle de qualidade e por apontar necessidades de melhoria.

Agora que você conhece um pouco mais sobre alguns dos possíveis membros de uma equipe, consegue entender melhor a estrutura à qual um dia pertencerá. Você poderá ver detalhes sobre esses e outros profissionais da área de TI no conteúdo sobre planejamento de carreira em análise de banco de dados.